

# **Virtuoso Spectre Circuit Simulator RF Analysis User Guide**

**Product Version 6.2  
June 2007**

© 1994–2007 Cadence Design Systems, Inc. All rights reserved.

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.

MMSIM contains technology licensed from, and copyrighted by: C. L. Lawson, R. J. Hanson, D. Kincaid, and F. T. Krogh © 1979, J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson © 1988, J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling © 1990; University of Tennessee, Knoxville, TN and Oak Ridge National Laboratory, Oak Ridge, TN © 1992-1996; Brian Paul © 1999-2003; M. G. Johnson, Brisbane, Queensland, Australia © 1994; Kenneth S. Kundert and the University of California, 1111 Franklin St., Oakland, CA 94607-5200 © 1985-1988; Hewlett-Packard Company, 3000 Hanover Street, Palo Alto, CA 94304-1185 USA © 1994, Silicon Graphics Computer Systems, Inc., 1140 E. Arques Ave., Sunnyvale, CA 94085 © 1996-1997, Moscow Center for SPARC Technology, Moscow, Russia © 1997; Regents of the University of California, 1111 Franklin St., Oakland, CA 94607-5200 © 1990-1994, Sun Microsystems, Inc., 4150 Network Circle Santa Clara, CA 95054 USA © 1994-2000, Scriptics Corporation, and other parties © 1998-1999; Aladdin Enterprises, 35 Eyal St., Kiryat Arye, Petach Tikva, Israel 49511 © 1999 and Jean-loup Gailly and Mark Adler © 1995-2005; RSA Security, Inc., 174 Middlesex Turnpike Bedford, MA 01730 © 2005.

All rights reserved. Associated third party license terms may be found at <install\_dir>/doc/OpenSource/\*

**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.

**Patents:** Cadence Product Virtuoso Spectre Circuit Simulator RF Analysis, described in this document, is protected by U.S. Patents 5,610,847; 5,790,436; 5,812,431; 5,859,785; 5,949,992; 5,987,238; 6,088,523; 6,101,323; 6,151,698; 6,181,754; 6,260,176; 6,278,964; 6,349,272; 6,374,390; 6,493,849; 6,504,885; 6,618,837; 6,636,839; 6,778,025; 6,832,358; 6,851,097; 7,035,782; 7,085,700.

**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.

**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

---

<u>Preface</u> .....	21
<u>Features in Spectre L and Spectre XL</u> .....	21
<u>Related Documents</u> .....	22
<u>1</u>	
<u>Spectre RF Analyses</u> .....	23
<u>Periodic Analyses</u> .....	23
<u>Quasi-Periodic Analyses</u> .....	24
<u>Envelope Analysis</u> .....	25
<u>The Flexible Balance and Shooting Method Simulation Engines</u> .....	25
<u>Flexible Balance Method</u> .....	25
<u>Shooting Method</u> .....	25
<u>Large vs. Small Signal Analysis</u> .....	26
<u>2</u>	
<u>Spectre RF Simulation Form Reference</u> .....	29
<u>Choosing Analyses Form</u> .....	30
<u>Field Descriptions for the Choosing Analyses Form</u> .....	31
<u>Accuracy Defaults (errpreset) (PSS, QPSS, and ENVLP)</u> .....	33
<u>Additional Time for Stabilization (tstab) (PSS and QPSS)</u> .....	34
<u>Analysis</u> .....	34
<u>Beat Frequency, Beat Period, and Auto Calculate (PSS)</u> .....	35
<u>Clock Name and Select Clock Name Button (ENVLP)</u> .....	36
<u>Do Noise (PSP and QPSP)</u> .....	37
<u>Enabled</u> .....	37
<u>Engine (ENVLP, PSS, QPSS)</u> .....	37
<u>Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)</u> .....	38
<u>Fund Frequency (ENVLP)</u> .....	42
<u>Fundamental Tones (PSS and QPSS)</u> .....	42
<u>Harmonics (QPSS)</u> .....	46

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Input Source and Reference Side-Band (Pnoise)</u>	48
<u>Input Source and Reference Side-Band (QPnoise)</u>	52
<u>Modulated Analysis (PAC, PXF) — One of the Specialized Analyses</u>	56
<u>Noise Type (Pnoise)</u>	59
<u>Options</u>	62
<u>Oscillator (PSS)</u>	63
<u>Output (PXF and QPXF)</u>	64
<u>Output (Pnoise and QPnoise)</u>	65
<u>Output Harmonics (PSS and ENVLP)</u>	67
<u>Period (ENVLP)</u>	70
<u>Periodic Stab Analysis Notification (PSTB)</u>	70
<u>Probe Instance (PSTB)</u>	72
<u>PSS Beat Frequency (PAC, PSTB, Pnoise, and PXF)</u>	72
<u>Save Initial Transient Results (PSS and QPSS)</u>	73
<u>Select Ports (PSP and QPSP)</u>	73
<u>Sidebands (PAC, Pnoise, and PXF)</u>	78
<u>Sidebands (QPAC, QPnoise, and QPXF)</u>	82
<u>Specialized Analyses (PAC)</u>	85
<u>Rapid IP3 Specialized PAC Analysis</u>	89
<u>Start ACPR Wizard (ENVLP)</u>	91
<u>Stop Time (ENVLP)</u>	92
<u>Sweep (PSS and QPSS)</u>	92
<u>Sweep Range, Sweep Type, and Add Specific Points (PSS and QPSS)</u>	95
<u>Sweep Type (PSTB)</u>	98
<u>Sweeptype (Pnoise)</u>	99
<u>Sweeptype (PAC and PXF)</u>	101
<u>Sweeptype (PSP and QPSP)</u>	103
<u>Options Forms</u>	104
<u>Field Descriptions for the Options Forms</u>	106
<u>Accuracy Parameters (PSS, QPSS, and ENVLP)</u>	106
<u>Additional Parameters (All)</u>	110
<u>Annotation Parameters (All)</u>	110
<u>Convergence Parameters (All)</u>	110
<u>Initial Condition Parameters (PSS, QPSS, and ENVLP)</u>	111
<u>Integration Method Parameters (PSS, QPSS, and ENVLP)</u>	112
<u>Multitone Stabilization Parameter (QPSS)</u>	113

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Newton Parameters (PSS, QPSS, and ENVLP)</u>	113
<u>Output Parameters (All)</u>	113
<u>Simulation Bandwidth Parameters (ENVLP)</u>	116
<u>Simulation Interval Parameters (ENVLP, PSS and QPSS)</u>	116
<u>State File Parameters (ENVLP, PSS and QPSS)</u>	116
<u>Time Step Parameters (ENVLP, PSS, QPSS)</u>	117
<u>Direct Plot Form</u>	118
<u>Opening the Direct Plot Form</u>	118
<u>Defining Measurements in a Plot Form</u>	120
<u>Plotting Data for Swept Simulations</u>	120
<u>Selecting Sidebands and Harmonics</u>	122
<u>Generating a Spectral Plot</u>	122
<u>Saving a Displayed Output and Displaying Saved Outputs.</u>	124
<u>Changing the Noise Floor of a Spectral Plot</u>	125
<u>Generating a Time Waveform</u>	125
<u>Saving a Displayed Output and Displaying Saved Outputs.</u>	126
<u>Plotting Complex Impedance</u>	127
<u>Field Descriptions for the Direct Plot Form</u>	128
<u>1st Order Harmonic</u>	128
<u>Add To Outputs</u>	128
<u>Analysis</u>	129
<u>Circuit Input Power (QPSS, PAC)</u>	129
<u>Close Contours (PSS and ENVLP)</u>	130
<u>Extrapolation Point (PSS and QPSS)</u>	130
<u>First-Order Harmonic (PSS)</u>	130
<u>First Order Harmonic</u>	131
<u>First Order Sideband (PAC)</u>	131
<u>Freq. Multiplier (Pnoise)</u>	132
<u>Function</u>	132
<u>Gain Compression (PSS and QPSS)</u>	133
<u>Harmonic Frequency (PSS)</u>	134
<u>Harmonic Number (ENVLP)</u>	134
<u>Input Harmonic (PSS)</u>	135
<u>Input Harmonic (QPSS)</u>	135
<u>Input Power Value (dBm) (PSS, QPSS, and PAC)</u>	136
<u>Input or Output Referred 1dB Compression (PSS and QPSS)</u>	136

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Input or Output Referred IPN and Order (PSS, QPSS, and PAC)</u>	137
<u>Maximum Reflection Magnitude (ENVLP)</u>	138
<u>Min Reflection Mag</u>	138
<u>Modifier</u>	139
<u>Modulated Input/Output (PAC and PXF)</u>	139
<u>Noise Type</u>	140
<u>Number of Contours</u>	140
<u>Nth Order Harmonic (QPSS)</u>	141
<u>Nth Order Sideband (PAC)</u>	141
<u>Order</u>	142
<u>Output Harmonic (PSS)</u>	142
<u>Output Harmonic (For QPSS)</u>	143
<u>Output Sideband (PAC and PXF)</u>	143
<u>Plot and Replot</u>	144
<u>Plot Mode</u>	144
<u>Power Spectral Density Parameters (ENVLP)</u>	145
<u>Reference Resistance (ENVLP)</u>	146
<u>Resistance</u>	146
<u>Select</u>	147
<u>Signal Level (PSS, QPSS)</u>	148
<u>Sweep (PSS, PXF, and ENVLP)</u>	149
<u>Variable Value (PSS, QPSS, and PAC)</u>	149
<u>ACPR Wizard</u>	150
<u>Clock Name</u>	152
<u>How to Measure</u>	152
<u>Channel Definitions</u>	153
<u>Simulation Control</u>	154
<u>OK and Apply</u>	155
<u>Large Signal S-Parameter Wizard</u>	156
<u>Define Input/Output</u>	158
<u>Sweep</u>	159
<u>Sweep Range</u>	159
<u>Sweep Type</u>	160
<u>OK and Apply</u>	161
<u>The Spectre RF Simulation Forms Quick Reference</u>	162
<u>Choosing Analyses Form</u>	162



---

<u>Option Forms</u> .....	172
<u>Direct Plot Form</u> .....	174
<u>ACPR Wizard</u> .....	174

## 3

### Setting Up for the Examples .....

<u>Setting Up Environment Variables and the Path Statement</u> .....	177
<u>Using Spectre RF from the MMSIM Hierarchy</u> .....	177
<u>Accessing the Most Current Spectre RF Documentation</u> .....	178
<u>Creating a Local Editable Copy of the rfExamples Library</u> .....	178
<u>Setting Up the Cadence Libraries</u> .....	179
<u>Using the Library Path Editor</u> .....	179
<u>Using a UNIX Shell Window</u> .....	181
<u>Setting Up For Simulation</u> .....	181
<u>Opening a Circuit in the Schematic Window</u> .....	181
<u>Choosing Simulator Options</u> .....	184
<u>Specifying Outputs to Save</u> .....	185
<u>Setting Up Model Libraries</u> .....	186
<u>Editing Design Variable Values</u> .....	187

## 4

### Modeling Transmission Lines .....

<u>The Transmission Line Models: tline3, mline and mtline</u> .....	191
<u>The tline3 Model</u> .....	191
<u>The mline Model</u> .....	192
<u>The mtline Model</u> .....	192
<u>LMG Use Models</u> .....	192
<u>Modeling Transmission Lines Using the LMG GUI</u> .....	193
<u>Modeling Transmission Lines Without Using the LMG GUI</u> .....	194
<u>Default Values and the initlmg File</u> .....	194
<u>Tline3 Transmission Line Modeling Examples</u> .....	195
<u>Creating a tline3 Macromodel in the LMG GUI</u> .....	195
<u>Using an Existing tline3 Macromodel in the Schematic Flow</u> .....	205
<u>Resolving A Possible Error Message for a tline3 Model File</u> .....	212
<u>Creating a tline3 Macromodel in the Schematic Flow</u> .....	213

<u>Mline Transmission Line Modeling Examples</u> .....	218
<u>Creating an mline Transmission Line Model Starting LMG From UNIX</u> .....	218
<u>Using an mline Macromodel in the Schematic Flow</u> .....	223
<u>Resolving a Possible Error Message for an mline Model File</u> .....	228
<u>Creating an mline Transmission Line in the Schematic Flow</u> .....	230
<u>Looking at mtline in the Analog Design Environment</u> .....	234
<u>Coplanar Waveguide Modeling and Analysis</u> .....	238
<u>Using LMG to Obtain Subcircuit Macromodel and LRCG Files</u> .....	238
<u>Resolving a Possible Error Message in the mtline Model File</u> .....	244
<u>General Theory of Coplanar Waveguide Analysis</u> .....	245
<u>Simulating Coplanar Waveguides with the Generated LRCG File</u> .....	247
<u>Simulating Coplanar Waveguides with the Generated Subcircuit Macromodel File</u> ..	273
<u>The LMG GUI Reference</u> .....	286
<u>Menus for the LMG GUI</u> .....	288
<u>Function Buttons for the LMG GUI</u> .....	296

## 5

<u>Creating and Using Receiver K-Models</u> .....	299
<u>Procedures for Simulating k mod extraction example</u> .....	300
<u>Analysis Setup</u> .....	306
<u>Setting Up the PSS Analyses</u> .....	309
<u>Running the Simulation</u> .....	314
<u>Creating the K-Model</u> .....	317
<u>More About the K-Model</u> .....	321
<u>K-model data files</u> .....	327

## 6

<u>Creating and Using Transmitter J-Models</u> .....	331
<u>Procedures for Simulating j mod extraction example</u> .....	332
<u>Optional: Setting Up the Input and Output Jigs</u> .....	334
<u>Analysis Setup</u> .....	337
<u>Running the Simulation</u> .....	343
<u>Creating the J-Model</u> .....	343
<u>Using the J-model in a Circuit</u> .....	345
<u>More About the J-model</u> .....	356

## 7

<b>Modeling Transmitters</b> .....	369
<b>Envelope Analysis</b> .....	370
<u>Opening the EF example Circuit in the Schematic Window</u> .....	370
<u>Opening the Simulation Window</u> .....	372
<u>Setting Up the Model Libraries</u> .....	373
<u>Setting Up an Envelope Analysis</u> .....	376
<u>Following the Baseband Signal Changes Through an Ideal Circuit</u> .....	382
<u>Following the Baseband Signal Changes Through a Non-Ideal Circuit</u> .....	389
<u>Plotting the Complete Baseband Signal</u> .....	393
<u>Plotting the Baseband Trajectory</u> .....	397
<b>Measuring ACPR and PSD</b> .....	408
<u>Measuring ACPR</u> .....	411
<u>Estimating PSD From the Direct Plot Form</u> .....	422
<u>Reference Information for ACPR and PSD Calculations</u> .....	428
<b>Measuring Load-Pull Contours and Load Reflection Coefficients</b> .....	439
<u>Creating and Setting Up the Modified Circuit (EF LoadPull)</u> .....	439
<u>Setting Up and Running the PSS and Parametric Analyses</u> .....	450
<u>Displaying Load Contours</u> .....	454
<u>Moving to Differential Mode</u> .....	469
<u>Setting Up the EF example Schematic for the First Simulation</u> .....	470
<u>Adding Components to the Schematic</u> .....	474
<u>Setting Up the sparamfirst Schematic</u> .....	477
<u>Setting Up and Running the Second sp Simulation</u> .....	479
<u>Using the S-Parameter Input File with a Spectre RF Envlp Analysis</u> .....	487
<b>Measuring AM and PM Conversion with Modulated PAC, AC and PXF Analyses</b> .....	497
<u>The Modulated Analysis Settings</u> .....	497
<u>Creating the EF AMP Circuit</u> .....	498
<u>Setting Up the EF AMP Circuit</u> .....	506
<u>Running the Simulations</u> .....	523
<u>Plotting and Calculating PAC Modulated Results</u> .....	523
<u>Plotting and Calculating PXF Modulated Results</u> .....	528
<b>Measuring Jitter with PSS and Pnoise Analyses</b> .....	532
<u>Setting Up the EF AMP Circuit</u> .....	532
<u>Opening the EF AMP Circuit in the Schematic Window</u> .....	533

<u>Opening the Simulation Window for the EF AMP Circuit</u> .....	534
<u>Setting Up and Running the PSS and Pnoise Analyses</u> .....	535
<u>Running the Simulations</u> .....	541
<u>Plotting the Jitter Measurement</u> .....	542

## 8

### Methods for Top-Down RF System Design .....

<u>Methods for Top Down RF System Design</u> .....	553
<u>Top-Down Design of RF Systems</u> .....	554
<u>Use Model for Top Down Design</u> .....	555
<u>Baseband Modeling</u> .....	558
<u>Example Comparing Baseband and Passband Models</u> .....	560
<u>rfLib Library Overview</u> .....	572
<u>Use Model and Design Example</u> .....	576
<u>Opening a New Schematic Window</u> .....	577
<u>Opening the Analog Environment</u> .....	578
<u>Constructing the Baseband Model for the Receiver</u> .....	579
<u>Setting Variable Values for the Receiver Schematic</u> .....	616
<u>Setting Up and Running a Transient Analysis</u> .....	619
<u>Examining the Results: Eye Diagram, Histogram, and Scatter Plot</u> .....	621
<u>Generating the Scatter Plot</u> .....	625
<u>Summarizing the Design Procedure</u> .....	654
<u>Creating a Passband View of the Architectural Model</u> .....	654
<u>Comparing Baseband and Passband Models</u> .....	658
<u>Relationship Between Baseband and Passband Noise</u> .....	661
<u>Introduction to Analysis</u> .....	662
<u>Prep Steps for Analyses</u> .....	663

## 9

### Cosimulation with MATLAB and Simulink .....

<u>Introduction to Cosimulation with MATLAB</u> .....	672
<u>Software Requirements</u> .....	672
<u>Setting Up and Running a Cosimulation</u> .....	672
<u>Connecting the Coupler Block Into the System-Level Simulink Schematic</u> .....	673
<u>Determining How You Want to Start and Run the Cosimulation</u> .....	677

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Generating a Netlist for the Lower-Level Block</u> .....	677
<u>Preparing the Netlist When Using ADE</u> .....	677
<u>Preparing the Netlist Without Using a Graphical User Interface</u> .....	683
<u>Running the Cosimulation</u> .....	685
<u>Starting the Two Applications Separately</u> .....	686
<u>Starting Spectre RF Manually and MATLAB Automatically</u> .....	686
<u>Starting MATLAB Manually and Spectre RF Automatically</u> .....	687

### A

<u>Oscillator Noise Analysis</u> .....	689
<u>Models for Phase Noise</u> .....	693
<u>Linear Time-Invariant (LTI) Models</u> .....	694
<u>Linear Time-Varying (LTV) Models</u> .....	695
<u>Amplitude Noise and Phase Noise in the Linear Model</u> .....	699
<u>Details of the Spectre RF Calculation</u> .....	700
<u>Calculating Phase Noise</u> .....	703
<u>Setting Simulator Options</u> .....	704
<u>Troubleshooting Phase Noise Calculations</u> .....	706
<u>Known Limitations of the Simulator</u> .....	706
<u>What Can Go Wrong</u> .....	707
<u>Phase Noise Error Messages</u> .....	709
<u>The tstab Parameter</u> .....	710
<u>Frequently Asked Questions</u> .....	711
<u>Further Reading</u> .....	716
<u>References</u> .....	716

### B

<u>Using PSS Analysis Effectively</u> .....	719
<u>General Convergence Aids</u> .....	719
<u>Additional Convergence Aids</u> .....	720
<u>Convergence Aids for Oscillators</u> .....	721
<u>Running PSS Analysis Hierarchically</u> .....	722

## C

<b>Using the psin Component</b> .....	725
<u>Independent Resistive Source (psin)</u> .....	725
<u>Terminating the psin</u> .....	726
<u>Parameter Types for the psin Component</u> .....	727
<u>Name Parameters</u> .....	730
<u>psin Instance Parameter</u> .....	730
<u>General Waveform Parameters</u> .....	730
<u>Sinusoidal Waveform Parameters</u> .....	731
<u>Amplitude Modulation Parameters</u> .....	732
<u>FM Modulation Parameters</u> .....	734
<u>Noise Parameters</u> .....	737
<u>Temperature Effect Parameters</u> .....	738
<u>Small-Signal Parameters</u> .....	739
<u>Additional Notes</u> .....	741

## D

<b>The RF Library</b> .....	743
<u>The Contents of the rLib</u> .....	743
<u>Elements for Transistor-Level RF Circuit Design</u> .....	743
<u>Elements for Top-Down System-Level RF Design</u> .....	744
<u>Elements for Use With Both Design Methodologies</u> .....	744
<u>Elements for Bottom Up Transmitter Design</u> .....	745
<u>Models for Transistor-Level RF Circuit Design</u> .....	745
<u>Balun</u> .....	746
<u>Balun_com</u> .....	748
<u>Filters</u> .....	749
<u>Low-Noise Amplifier</u> .....	752
<u>Mixer</u> .....	755
<u>Oscillator</u> .....	762
<u>Quadrature Signal Generator</u> .....	765
<u>Measurement Elements</u> .....	768
<u>CDMA Signal Source (CDMA_reverse_xmit)</u> .....	769
<u>GSM Signal Source (GSM_xmtr)</u> .....	773

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Pi/4-DQPSK Signal Source (pi_over4_dqpsk)</u>	776
<u>Modifying the Baseband Signal Generators Using the Modelwriter</u>	782
<u>Testbenches Elements</u>	786
<u>Uncategorized Elements</u>	787
<u>Bottom-Up Design Elements</u>	787
<u>Models for Top-Down RF System Design</u>	788
<u>Baseband and Passband Models</u>	789
<u>Assumptions About Behavioral Models</u>	792
<u>Inputs and Outputs for Baseband Models</u>	792
<u>Some Common Model Parameters</u>	793
<u>Library Description</u>	814
<u>Notes on Models Involving Frequency Translation</u>	815
<u>Top Down Baseband and Passband Models: top_dwnBB and top_dwn PB</u>	817
<u>Power Amplifier Model</u>	817
<u>Low Noise Amplifier Baseband Model</u>	819
<u>IQ Modulator Models</u>	821
<u>IQ Demodulator</u>	828
<u>RF-to-IF and IF-to-RF Mixers</u>	836
<u>Passive Devices</u>	842
<u>Linear Time Invariant Filters</u>	850
<u>BB Loss</u>	860
<u>Phase Shifter Splitter</u>	861
<u>Phase Shifter Combiner</u>	865
<u>Variable Gain Amplifier Model</u>	872
<u>Comparison of Baseband and Passband Models</u>	873
<u>Measurement Blocks</u>	875
<u>Ideal Transformer (BB_xfmr)</u>	875
<u>Rectangular-to-Polar Transformation (rect_polar)</u>	876
<u>Polar-to-Rectangular Transformation (polar_rect)</u>	877
<u>Instrumentation and Terminating Blocks</u>	878
<u>Baseband Drive Signals (BB_driver, CDMA_reverse_xmit and GSM_xmtr)</u>	881
<u>References</u>	888
<u>WCDMA Blocks</u>	888
<u>Introduction</u>	888
<u>QPSK Modulation/Mapping</u>	889
<u>DL Common Channel Generator</u>	890

<u>Spreading</u> .....	891
<u>Scrambling/Scrambling Code Generator</u> .....	892
<u>OCNS Generator</u> .....	892
<u>SCH Generator/Multiplexer</u> .....	893
<u>Power Adjustment</u> .....	894
<u>Square-Root Raised Cosine Filtering</u> .....	894
<u>GMSK Block</u> .....	895

## E

<u>Plotting Spectre S-Parameter Simulation Data</u> .....	897
<u>Network Parameters</u> .....	897
<u>Equations for Network Parameters</u> .....	898
<u>Two-Port Scalar Quantities</u> .....	900
<u>Equations for Two-Port Scalar Quantities</u> .....	901
<u>Two-Port Gain Quantities</u> .....	903
<u>Equations for Two-Port Gain Calculations</u> .....	904
<u>Two-Port Network Circles</u> .....	906
<u>Equations for Two-Port Network Circle</u> .....	907
<u>Equation for VSWR (Voltage Standing Wave Ratio)</u> .....	910
<u>Equation for GD (group delay)</u> .....	911

## F

<u>Using QPSS Analysis Effectively</u> .....	913
<u>When Should You Use QPSS Analysis</u> .....	914
<u>Essentials of the MFT Method</u> .....	915
<u>QPSS and PSS Analyses Compared</u> .....	920
<u>QPSS and PSS/PAC Analyses Compared</u> .....	922
<u>QPSS Analysis Parameters</u> .....	922
<u>Switched Capacitor Filter Example</u> .....	923
<u>High-Performance Receiver Example</u> .....	925
<u>Running a QPSS Analysis</u> .....	926
<u>Picking the Large Fundamental</u> .....	926
<u>Setting Up Sources</u> .....	927
<u>Sweeping a QPSS Analysis</u> .....	928
<u>Convergence Aids</u> .....	929



<u>Memory Management</u> .....	931
<u>Understanding the Narration from the QPSS Analysis</u> .....	932
<u>References</u> .....	937

### G

<u>Introduction to the PLL library</u> .....	939
<u>Models in the PLL library</u> .....	940
<u>Introduction to the PLL Library Documentation</u> .....	941
<u>Phase-Domain Model of a Simple PLL</u> .....	941
<u>Example 1: Dynamic Test for Capture Range and Lock Range</u> .....	946
<u>Example 2: Loop Gain Measurement</u> .....	947
<u>Example 3: PM Input</u> .....	965
<u>Modeling a PFD-Based PLL</u> .....	967
<u>VCO</u> .....	968
<u>Charge Pump</u> .....	969
<u>Loop Filter</u> .....	969
<u>State-Space Averaged PFD (Phase-Domain Phase-Frequency Detector Model)</u> ..	970
<u>Lock Indicator</u> .....	973
<u>Example 5: Comparison With a Voltage-Domain Model</u> .....	975
<u>How the PFD Model Works</u> .....	977
<u>How the PDF/CP Pump Works</u> .....	977
<u>References</u> .....	982

### H

<u>Using the Port Component</u> .....	985
<u>Capabilities of the port Component</u> .....	985
<u>Terminating the port</u> .....	986
<u>Parameters for the Port Component</u> .....	986
<u>Port parameters</u> .....	987
<u>General waveform parameters</u> .....	987
<u>DC Waveform parameters</u> .....	988
<u>Pulse waveform parameters</u> .....	988
<u>PWL waveform parameters</u> .....	988
<u>Sinusoidal waveform parameters</u> .....	988
<u>Amplitude and Frequency modulation parameters</u> .....	989

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Exponential waveform parameters</u>	989
<u>Small-signal parameters</u>	990
<u>Temperature effect parameters</u>	990
<u>Noise parameters</u>	990
<u>Port Parameters</u>	990
<u>General Waveform Parameters</u>	991
<u>DC Waveform Parameters</u>	992
<u>DC voltage</u>	992
<u>Pulse Waveform Parameters</u>	993
<u>PWL Waveform Parameters</u>	995
<u>Waveform Entry Method</u>	996
<u>Sinusoidal Waveform Parameters</u>	998
<u>Modulation Parameters</u>	1000
<u>Effect of Amplitude Modulation (Background Information)</u>	1002
<u>Display second sinusoid</u>	1004
<u>Display multi sinusoid</u>	1006
<u>Number of FM Files</u>	1007
<u>Exponential Waveform Parameters</u>	1008
<u>Noise Parameters</u>	1009
<u>Small-Signal Parameters</u>	1011
<u>Temperature Effect Parameters</u>	1013
<u>How Temperature Parameters Affect the Voltage Level (Background Information)</u>	1014
<u>Additional Notes</u>	1014

### I

<u>Analyzing Time-Varying Noise</u>	1017
<u>Calculating Noise Correlation Coefficients</u>	1023
<u>Reference Information on Time-Varying Noise</u>	1032
<u>Thermal Noise</u>	1033
<u>Linear Systems and Noise</u>	1039
<u>Time-Varying Systems and the Autocorrelation Function</u>	1043
<u>Time-Varying Systems and Frequency Correlations</u>	1047
<u>Summary</u>	1052

## J

<u>Using Tabulated S-parameters</u> .....	1055
<u>Using the nport Components</u> .....	1056
<u>Controlling Model Accuracy</u> .....	1058
<u>Using relerr and abserr</u> .....	1058
<u>Using the ratorder Parameter</u> .....	1061
<u>Troubleshooting</u> .....	1061
<u>Assessing the Quality of the Rational Interpolation</u> .....	1061
<u>Model Reuse</u> .....	1062
<u>The S-Parameter File Format Translator (SPTR)</u> .....	1063
<u>References</u> .....	1063

## K

<u>Measuring AM, PM, and FM Conversion</u> .....	1065
<u>Derivation</u> .....	1065
<u>Positive Frequencies</u> .....	1069
<u>FM Modulation</u> .....	1070
<u>Simulation</u> .....	1071
<u>Results</u> .....	1073
<u>Conclusion</u> .....	1076
<u>References</u> .....	1076

## L

<u>Using PSP and Pnoise Analyses</u> .....	1077
<u>Overview of PSP and Pnoise Analyses</u> .....	1077
<u>Periodic S-parameters</u> .....	1078
<u>Linear Time-Invariant S-Parameters</u> .....	1078
<u>Frequency Translating S-Parameters</u> .....	1079
<u>Calculating Noise in Linear Time-Invariant (DC Bias) Circuits</u> .....	1084
<u>Calculating Noise in Time-Varying (Periodic Bias) Circuits</u> .....	1085
<u>Noise Correlation Matrices and Equivalent Noise Sources</u> .....	1086
<u>Noise Figure</u> .....	1089
<u>Performing Noise Figure Computations</u> .....	1089
<u>Noise Figure From Noise and SP Analyses</u> .....	1090

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

<u>Pnoise (SSB) Noise Figure</u> .....	1091
<u>DSB Noise Figure</u> .....	1092
<u>IEEE Noise Figure</u> .....	1093
<u>Noise Computation Example</u> .....	1096
<u>Input Referred Noise</u> .....	1097
<u>Using Input Referred Noise</u> .....	1098
<u>How IRN is Calculated</u> .....	1099
<u>Relation to Gain</u> .....	1101
<u>Referring Noise to Ports</u> .....	1101
<u>Gain Calculations</u> .....	1102
<u>Definitions of Gain</u> .....	1102
<u>Gain Calculations in Pnoise</u> .....	1106
<u>Phase Noise</u> .....	1108
<u>Frequently Asked Questions</u> .....	1109
<u>Known Problems and Limitations</u> .....	1112
<u>Dubious AC-Noise Analysis Features</u> .....	1112
<u>Gain in Pnoise and PSP Analyses Inconsistent</u> .....	1113
<u>Harmonics and Sidebands in PSP, PAC, PXF, and Pnoise Analyses</u> .....	1113
<u>Index</u> .....	1117

## Preface

---

Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) provides functionality designed for the needs of RF designers. This manual contains information about the RF functionality but for a complete description of Spectre RF functionality you also need to refer to the Virtuoso Spectre Circuit Simulator documents.

This manual assumes that you are familiar with RF circuit design and that you have some familiarity with SPICE simulation. It contains information about using Spectre RF within the Virtuoso<sup>®</sup> analog design environment (ADE).

Spectre RF analyses support the efficient calculation of the operating point, transfer function, noise, and distortion of common RF and communication circuits, such as mixers, oscillators, sample and holds, and switched capacitor filters.

Spectre RF supports a multi-technology simulation (MTS) mode that enables the simulation of a system consisting of blocks designed with different processes. For more information, see chapter 5 in *Virtuoso Spectre Circuit Simulator User Guide*.

## Features in Spectre L and Spectre XL

Spectre and Spectre RF features are available in different tiers. The L tier offers basic design creation and implementation capabilities. The XL tier introduces new technologies and advancements in automation. The following table provides an overview of the features supported by each tier of products.

### Features Supported in Spectre L and Spectre XL Tiers

Features	L Tier	XL Tier
DC, AC, and transient analysis	X	X
Noise, transfer function, and sensitivity analysis	X	X
Transient noise analysis	X	X
Monte Carlo and parametric statistical support	X	X
Built-in measurement description language	X	X
Parametric sweep	X	X

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Preface

### Features Supported in Spectre L and Spectre XL Tiers, *continued*

Features	L Tier	XL Tier
Periodic and quasi-periodic steady state analysis (PSS and QPSS) based on harmonic balance and shooting Newton		X
Periodic and quasi-periodic noise analysis (PNoise, QPNoise)		X
Periodic and quasi-periodic small signal analysis (PAC, PXF, PSP, QPAC, QPSF, QPSP)		X
Periodic stability analysis (PSTB)		x
Time-domain and frequency-domain envelope analysis		X
Perturbation-based rapid IP2 and IP3		X
Co-simulation with Simulink <sup>®</sup> from The MathWorks		X
MMSIM Toolbox for MATLAB <sup>®</sup> from The MathWorks		X

## Related Documents

The following documents can give you more information about Spectre RF and related products.

- To learn more about the Analog Circuit Design Environment, consult the *Virtuoso<sup>®</sup> Analog Design Environment User Guide*.
- To learn more about Spectre RF, see the reference information and theoretical concepts in *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

---

# Spectre RF Analyses

---

The Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) analyses add capabilities to the Virtuoso Spectre circuit simulator, such as direct, efficient computation of steady-state solutions and simulation of circuits that translate frequency. You use the Spectre RF analyses in combination with the Fourier analysis capability of the Spectre circuit simulator and with the Verilog®-A behavioral modeling language.

## Periodic Analyses

Spectre RF adds periodic large and small-signal analyses to Spectre simulation.

- Periodic Steady-State Analysis, PSS (Large-Signal)
- Periodic AC Analysis, PAC (Small-Signal)
- Periodic S-Parameter Analysis, PSP (Small-Signal)
- Periodic Transfer Function Analysis, PXF (Small-Signal)
- Periodic Noise Analysis, Pnoise (Small-Signal)

For details on the periodic analyses, see *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

Periodic Steady-State (PSS) analysis is a large-signal analysis that directly computes the periodic steady-state response of a circuit. With PSS, simulation times are independent of the time constants of the circuit, so PSS can quickly compute the steady-state response of circuits with long time constants, such as high-Q filters and oscillators. You can also sweep frequency or other variables using PSS.

After completing a PSS analysis, the Spectre RF simulator can model frequency conversion effects by performing one or more of the periodic small-signal analyses, Periodic AC analysis (PAC), Periodic S-Parameter analysis (PSP), Periodic Transfer Function analysis (PXF) and Periodic Noise analysis (Pnoise). The periodic small-signal analyses are similar to the Spectre L AC, SP, XF, and Noise analyses, but you can apply the periodic small-signal analyses to periodically driven circuits that exhibit frequency conversion. Examples of

important frequency conversion effects include conversion gain in mixers, noise in oscillators, and filtering using switched-capacitors.

Therefore, with periodic small-signal analyses you apply a small signal at a frequency that may be noncommensurate (not harmonically related) to the small signal fundamental. This small signal is assumed to be small enough so that it is not distorted by the circuit.

## Quasi-Periodic Analyses

Spectre RF adds quasi-periodic large and small-signal analyses to Spectre L simulation.

Quasi-Periodic Steady-State Analysis, QPSS (Large-Signal)

- Quasi-Periodic AC Analysis, QPAC (Small-Signal)
- Quasi-Periodic S-Parameter Analysis, QPSP (Small-Signal)
- Quasi-Periodic Transfer Function Analysis, QPXF (Small-Signal)
- Quasi-Periodic Noise Analysis, QPnoise (Small-Signal)

For details on the quasi-periodic analyses, see *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

Quasi-Periodic Steady-State (QPSS) analysis, a large-signal analysis, is used for circuits with multiple large tones. With QPSS, you can model periodic distortion and include harmonic effects. (Periodic small-signal analyses assume the small signal you specify generates no harmonics). QPSS computes both a large signal, the periodic steady-state response of the circuit, and also the distortion effects of a specified number of moderate signals, including the distortion effects of the number of harmonics that you choose.

With QPSS, you can apply one or more additional signals at frequencies not harmonically related to the large signal, and these signals can be large enough to create distortion. In the past, this analysis was called Pdisto analysis.

Quasi-Periodic Noise (QPnoise) analysis is similar to a transient noise analysis, except that it includes frequency conversion and intermodulation effects. QPnoise analysis is useful for predicting the noise behavior of mixers, switched-capacitor filters and other periodically or quasi-periodically driven circuits. QPnoise analysis linearizes the circuit about the quasi-periodic operating point computed in the prerequisite QPSS analysis. It is the quasi-periodically time-varying nature of the linearized circuit that accounts for the frequency conversion and intermodulation.



The Quasi-Periodic AC (QPAC), Quasi-Periodic S-Parameter (QPSP) and Quasi-Periodic Transfer Function (QPXF) analyses all work in a similar way as the Spectre L AC, SP and XF analyses.

## Envelope Analysis

Envelope analysis allows RF circuit designers to efficiently and accurately predict the envelope transient response of the RF circuits used in communication systems.

For details on the Envelope analysis, see *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

## The Flexible Balance and Shooting Method Simulation Engines

Spectre RF provides a choice of simulation engines between the traditional *shooting method* and the new *flexible balance method* (FB) with most analyses. The flexible balance engine complements the capabilities of the shooting method.

The combination of a PSS or QPSS analysis using the shooting method with a time-varying small-signal analyses is efficient for circuits that respond in a strongly nonlinear manner to the LO or the clock. Consequently, you can use the Spectre RF simulations with the shooting method to simulate strongly nonlinear circuits, such as switched-capacitor filters, switching mixers, chopper-stabilized amplifiers, PLL-based frequency multipliers, sample-and-holds, and samplers. You can use the Spectre RF simulations with the flexible balance method to simulate weakly nonlinear circuits as well.

### Flexible Balance Method

The flexible balance engine supports frequency domain harmonic balance analyses. It provides efficient and robust simulation for linear and weakly nonlinear circuits. The flexible balance engine is supported on the Solaris, Linux, HP and IBM platforms for both 32 and 64 bit architectures. see *Virtuoso Spectre Circuit Simulator RF Analysis Theory* for more information on the flexible balance engine.

### Shooting Method

Spectre RF has traditionally used an engine known as the *shooting method* [kundert90] to implement periodic and quasi-periodic analyses and the envelope analysis. The shooting

method is a time domain method and it is used in most descriptions and examples in this manual.

## Large vs. Small Signal Analysis

Spectre RF provides a variety of time-varying small signal analysis for both periodic and quasi-periodic circuits. These small-signal analyses accurately model the frequency translation effects of time-varying circuits. Rather than using traditional small-signal analyses for circuits that exhibit frequency translation, such as amplifiers and filters, you can simulate these circuits using time-varying small-signal analyses.

Circuits designed to translate from one frequency to another include mixers, detectors, samplers, frequency multipliers, phase-locked loops and parametric oscillators. Such circuits are commonly found in wireless communication systems.

Other circuits that translate energy between frequencies as a side effect include oscillators, switched-capacitor and switched-current filters, chopper-stabilized and parametric amplifiers, and sample-and-hold circuits. These circuits are found in both analog and RF circuits.

The quasi-periodic small-signal analyses accurately model the small signal characteristics of circuits with a quasi-periodic operating point, such as mixers with multiple LO frequencies or large RF inputs. The periodic small-signal analyses are more useful for circuits with a single fundamental frequency.

Applying a time-varying small-signal analysis is a two-step process.

First, the simulator ignores the small input or noise signals while performing a PSS or QPSS analysis to compute the steady-state response to the remaining large-signals, such as the LO or the clock. The initial PSS or QPSS analysis, linearizes the circuit about the time-varying large-signal operating point.

For each subsequent small-signal analysis, the simulator uses the time-varying operating point computed by the PSS or QPSS analysis to predict the circuit response to a small sinusoid at an arbitrary frequency. You can perform any number of small-signal analyses after calculating the time-varying large-signal operating point.

The input signals for the small-signal analyses must be sufficiently small that the circuit does not respond to them in a significantly nonlinear fashion. You should use input signals that are at least 10 dB smaller than the 1 dB compression point. This restriction does not apply to the signals you apply in the large-signal analysis.

This two-step process is widely applicable because most circuits that translate frequency react in a strongly nonlinear manner to one stimulus, usually either the LO or the clock, while

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Analyses

---

they react in a weakly nonlinear manner to other stimuli such as the inputs. A mixer is a typical example. Its noise and conversion characteristics improve if it is discontinuously switched between two states by the LO, yet it must respond linearly to the input signal over a wide dynamic range.

To analyze a mixer with a small RF input and a single LO, you should use a PSS large-signal analysis followed by one or more of the PAC, PNoise, PSP or PXF small-signal analyses.

If the mixer has a small RF input and a large blocker as well as the LO, then a QPSS analysis would be the more appropriate large-signal analysis. Follow the QPSS analysis with one or more of the QPAC, QPNoise, QPSP or QPXF small-signal analyses for the RF input.

Some circuits, such as frequency dividers, generate subharmonics. PSS can simulate the large-signal behavior of such circuits if you specify the period  $T$  to be that of the subharmonic. For other circuits, such as delta-sigma modulators, the periodically driven circuits respond chaotically, and you must use transient analysis rather than the PSS or QPSS analyses.

With the time-varying small-signal analyses such as QPAC or PXF, unlike traditional small-signal analyses such as AC or XF, there are many transfer functions between any single input and output due to harmonics. Usually, however, only one or two harmonics provide useful information. For example, when you analyze the down-conversion mixers found in receivers, you want to know about the transfer function that maps the input signal at the RF to the output signal at the IF, which is usually the LO minus the RF.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Analyses

---

---

## Spectre RF Simulation Form Reference

---

The Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) simulation forms include the Choosing Analyses form, the Options forms, and the Results forms. The simulation forms change to display only the fields relevant for the currently selected analysis.

The field description topics are presented in the following sections.

- [“Choosing Analyses Form”](#) on page 30
- [“Field Descriptions for the Choosing Analyses Form”](#) on page 31
- [“Options Forms”](#) on page 104
- [“Field Descriptions for the Options Forms”](#) on page 106
- [“Direct Plot Form”](#) on page 118
- [“Field Descriptions for the Direct Plot Form”](#) on page 128
- [“ACPR Wizard”](#) on page 150
- [“Large Signal S-Parameter Wizard”](#) on page 156

Within each section, the form field descriptions are arranged alphabetically according to the top-level headings on the forms. The top-level headings are usually found at the leftmost margin of the form.

## Choosing Analyses Form

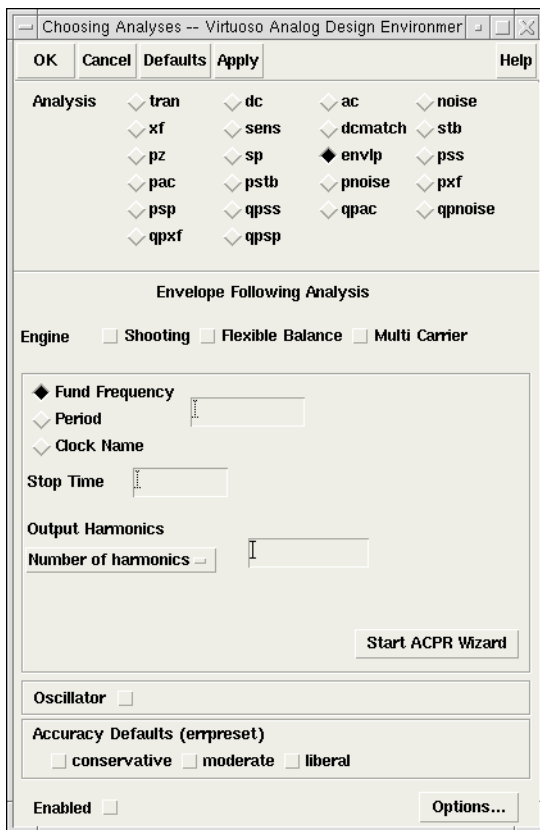
Use the Choosing Analyses form in the Analog Circuit Design Environment (the Simulation window) to select and set up RF simulations.

To open the Choosing Analyses form,

- In the Simulation window, select the *Analyses – Choose* command to open the Choosing Analyses form.

Figure 2-1 on page 30 shows the Choosing Analyses form for the Envelope analysis. The content of the Choosing Analyses form changes depending on the analysis selected.

Figure 2-1 Choosing Analyses Form



The Analysis section at the top of the Choosing Analyses form displays the available analyses, including the Spectre RF analyses (ENVLP, PSS, PAC, PSTB, Pnoise, PXF, PSP, QPSS, QPAC, QPnoise, QPXF, and QPSP). When you highlight an analysis in this section, the form changes to display the title of the analysis (directly below the analysis section), and below the title, relevant parameters for the selected simulation.

At the bottom of the form, highlight *Enabled* to select and run the analysis with the next simulation. Click *Options* to display the Options form for the selected analysis. Each Options form displays only the parameters relevant for that particular analysis.

The Spectre RF analyses are

- The periodic large-signal PSS, periodic steady state analysis
- The periodic large-signal QPSS, quasi-periodic steady state analysis
- The periodic small-signal analyses: PAC, PSTB, PSP, Pnoise, and PXF
- The quasi-periodic small-signal analyses: QPAC, QPSP, QPnoise, and QPXF
- The envelope ENVLP analysis

When you highlight an analysis type, the Choosing Analyses form changes to allow you to specify information for that simulation.

When your simulation requires that two analyses be run (for example, a PSS large-signal analysis followed by a Pnoise small-signal analysis), the Choosing Analyses form maintains the simulation set-up data for the two simulations interactively. For example, when you highlight *pnoise*, the values displayed in the Choosing Analyses form reflect the information you entered for the Pnoise analysis. When you highlight *pss*, the values displayed in the Choosing Analyses form reflect the information you entered for the PSS analysis.

Run the periodic small-signal PAC, PSP, Pnoise and PXF analyses after a large-signal PSS analysis. Run the quasi-periodic small-signal QPAC, QPSP, QPnoise and QPXF analyses after a large-signal QPSS analysis.

## Field Descriptions for the Choosing Analyses Form

The following sections describe the panes and fields that ever appear on the Choosing Analyses form, independently of the analysis that is selected. The descriptions are arranged alphabetically, according to the labels that are usually found along the left side of the form. If you are looking for descriptions of the Choosing Analyses form as they appear for a particular analysis, see “[Choosing Analyses Form](#)” on page 162.

The following sections are:

- “[Accuracy Defaults \(errpreset\) \(PSS, QPSS, and ENVLP\)](#)” on page 33
- “[Additional Time for Stabilization \(tstab\) \(PSS and QPSS\)](#)” on page 34
- “[Analysis](#)” on page 34

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

---

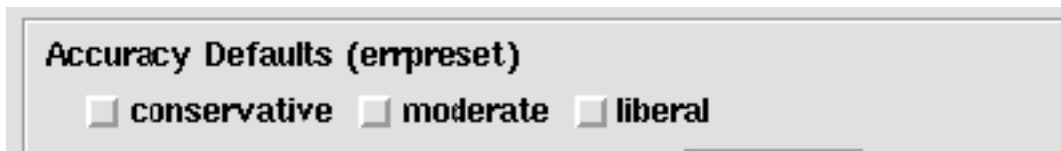
- [“Beat Frequency, Beat Period, and Auto Calculate \(PSS\)”](#) on page 35
- [“Clock Name and Select Clock Name Button \(ENVLP\)”](#) on page 36
- [“Do Noise \(PSP and QPSP\)”](#) on page 37
- [“Enabled”](#) on page 37
- [“Engine \(ENVLP, PSS, QPSS\)”](#) on page 37
- [“Frequency Sweep Range, Sweep Type, Add Specific Points \(Small-Signal\)”](#) on page 38
- [“Fund Frequency \(ENVLP\)”](#) on page 42
- [“Fundamental Tones \(PSS and QPSS\)”](#) on page 42
- [“Harmonics \(QPSS\)”](#) on page 46
- [“Input Source and Reference Side-Band \(Pnoise\)”](#) on page 48
- [“Input Source and Reference Side-Band \(QPnoise\)”](#) on page 52
- [“Modulated Analysis \(PAC, PXF\) — One of the Specialized Analyses”](#) on page 56
- [“Noise Type \(Pnoise\)”](#) on page 59
- [“Options”](#) on page 62
- [“Oscillator \(PSS\)”](#) on page 63
- [“Output \(PXF and QPXF\)”](#) on page 64
- [“Output \(Pnoise and QPnoise\)”](#) on page 65
- [“Output Harmonics \(PSS and ENVLP\)”](#) on page 67
- [“Period \(ENVLP\)”](#) on page 70
- [“Periodic Stab Analysis Notification \(PSTB\)”](#) on page 70
- [“Probe Instance \(PSTB\)”](#) on page 72
- [“PSS Beat Frequency \(PAC, PSTB, Pnoise, and PXF\)”](#) on page 72
- [“Save Initial Transient Results \(PSS and QPSS\)”](#) on page 73
- [“Select Ports \(PSP and QPSP\)”](#) on page 73
- [“Choose Harmonic Pop Up”](#) on page 75
- [“Sidebands \(PAC, Pnoise, and PXF\)”](#) on page 78
- [“Sidebands \(QPAC, QPnoise, and QPXF\)”](#) on page 82



- [“Specialized Analyses \(PAC\)”](#) on page 85
- [“Choose Harmonic Pop Up”](#) on page 87
- [“Rapid IP3 Specialized PAC Analysis”](#) on page 89
- [“Start ACPR Wizard \(ENVLP\)”](#) on page 91
- [“Stop Time \(ENVLP\)”](#) on page 92
- [“Sweep \(PSS and QPSS\)”](#) on page 92
- [“Sweep Range, Sweep Type, and Add Specific Points \(PSS and QPSS\)”](#) on page 95
- [“Sweep Type \(PSTB\)”](#) on page 98
- [“Sweep Type \(Pnoise\)”](#) on page 99
- [“Sweep Type \(PAC and PXF\)”](#) on page 101
- [“Sweep Type \(PSP and QPSP\)”](#) on page 103

## Accuracy Defaults (`errpreset`) (PSS, QPSS, and ENVLP)

Quickly adjusts the simulator parameters.



The `errpreset` parameter quickly adjusts the simulator accuracy parameters to fit your needs. In most cases, `errpreset` should be the only parameter you need to adjust.

For a fast simulation with reasonable accuracy, set `errpreset` to *liberal*.

For greater accuracy, set `errpreset` to *moderate*.

For maximum accuracy, set `errpreset` to *conservative*.

The effect of `errpreset` on other parameters varies depending on the type of analysis to which you are applying it.

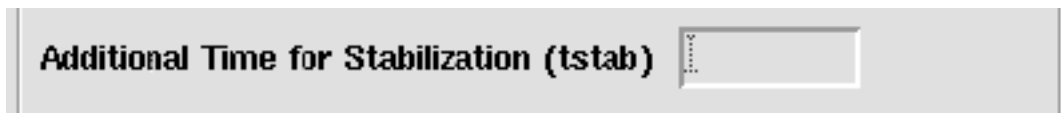
For details see the following sections in *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

- [The `errpreset` Parameter in PSS Analysis](#)

- [The errpreset Parameter in QPSS Analysis](#)
- [The errpreset Parameter in Envlp Analysis](#)

### Additional Time for Stabilization (tstab) (PSS and QPSS)

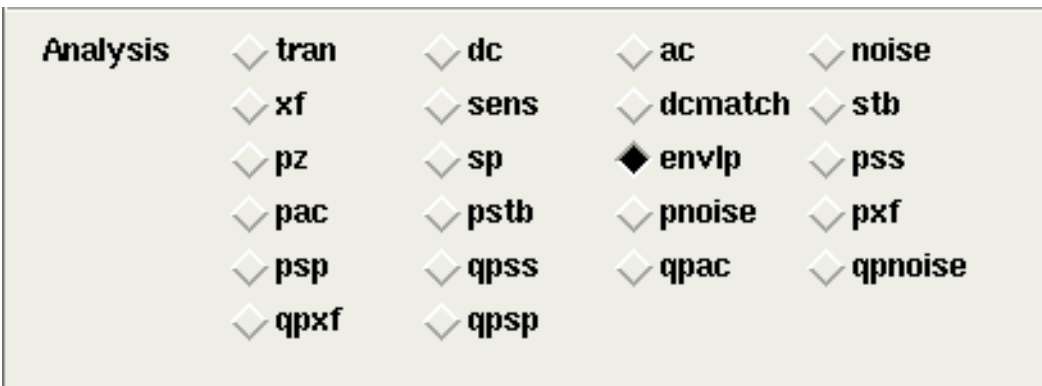
Specifies an amount of additional time to allow for the circuit to settle.



Use `tstab` if the circuit exhibits more than one periodic solution and you want only one. A long `tstab` can also improve convergence.

### Analysis

Selects the Spectre RF analysis type.



The RF analyses are

- The periodic large-signal PSS analysis
- The quasi-periodic large-signal QPSS analysis
- The periodic small-signal analyses: PAC, PSTB, PSP, Pnoise, and PXF
- The quasi-periodic small-signal analyses: QPAC, QPSP, QPnoise, and QPXF
- The Envelope analysis, ENVLP

When you highlight an analysis type, the Choosing Analyses form changes to allow you to specify information for that simulation. Below the Analysis buttons, the analysis title changes to the name of the analysis you select.

When your simulation requires that two analyses be run (for example, a PSS analysis followed by a Pnoise small-signal analysis), the Choosing Analyses form maintains the simulation set-up data for both simulations. You can edit the data for both simulations interactively. For example, when you highlight *pnoise*, the values displayed in the Pnoise Choosing Analyses form reflect the information you entered for the Pnoise analysis.

Run the periodic small-signal analyses PAC, PSTB, PSP, Pnoise, and PXF, after a large-signal PSS analysis. Run the quasi-periodic small-signal QPnoise analysis after a QPSS analysis.

### Beat Frequency, Beat Period, and Auto Calculate (PSS)

Determines whether the PSS analysis uses beat frequency or beat period and supplies an initial value.



The screenshot shows a control panel with two radio buttons: **◆ Beat Frequency** (selected) and **◇ Beat Period**. To the right of the selected radio button is a text input field containing the value **100M**. Further to the right is a checkbox labeled **Auto Calculate** which is checked.

- Highlight either *Beat Frequency* or *Beat Period*.
- The *Beat Frequency* (or *Beat Period*) field is initially empty.
- Enter a *Beat Frequency* value in one of two ways:
  - Type a frequency value for which all the tone frequencies are integer multiples of the value.
  - Select *Auto Calculate* to automatically calculate a value. The field disappears.  
  
The *Beat Frequency* value is calculated based on the tones present in the Fundamental Tones list box. It is the greatest common multiple of all the tone frequencies that are not small-signals.
- Enter a *Beat Period* value in one of two ways:
  - Type a period value for which all the tone frequencies are integer multiples of the inverse of the value.

- ❑ Click the *Auto Calculate* button to automatically calculate a value. The field disappears.

The *Beat Period* value is calculated based on the tones present in the Fundamental Tones list box. The *Beat Period* value is set to the inverse of the frequency.

The *Beat Frequency* (or *Period*) value is displayed in a read-only field at the top of the periodic small-signal analysis forms.

### Clock Name and Select Clock Name Button (ENVLP)

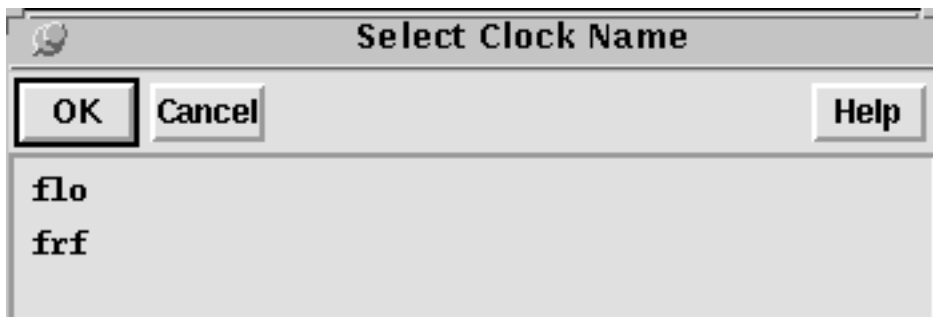
Identifies the clock signal for an ENVLP analysis. The simulator automatically determines the clock period by looking through all the sources with the specified name.



The screenshot shows a portion of the simulation form. On the left, there are three radio button options: 'Fund Frequency' (unselected), 'Period' (unselected), and 'Clock Name' (selected). To the right of these options is a text input field. Further to the right is a button labeled 'Select Clock Name'.

Enter a clock signal name in the *Clock Name* field in one of two ways:

- By typing the clock signal name in the *Clock Name* field.
- By clicking the *Select Clock Name* button to display a list of clock signals.



- Click to highlight a clock signal from the list.
- Click *OK* to select the signal and display it in the *Clock Name* field.

See [“Stop Time \(ENVLP\)”](#) on page 92 for related information.

## Do Noise (PSP and QPSP)

Performs noise measurements during the PSP or QPSP analysis.



Do Noise

yes

no

Maximum Sideband

Highlight *yes* for *Do Noise* to compute noise figure, equivalent noise sources, and noise parameters during the analysis.

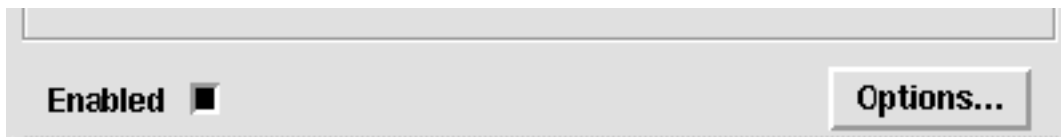
To include relevant noise folding effects, specify a maximum sideband value in the *MaximumSideband* field. A sideband array of the form

```
[-max. sideband . . . 0 . . . + max. sideband]
```

is automatically generated.

## Enabled

Includes the analysis in the next simulation.



Enabled

Options...

Select *Enabled* to perform the analysis in the next simulation. Enabled analyses are listed in the Simulation window.

Click *Options* to display the Options Form for that analysis.

## Engine (ENVLP, PSS, QPSS)

Specifies the method to be used for the simulation.



Engine  Shooting  Flexible Balance  Multi Carrier

**Shooting** engine combined with a time-varying small-signal analyses is efficient for circuits that respond in a strongly nonlinear manner to the LO or the clock. Consequently, you can use the Spectre RF simulations with the shooting engine to simulate strongly nonlinear circuits, such as switched-capacitor filters, switching mixers, chopper-stabilized amplifiers, PLL-based frequency multipliers, sample-and-holds, and samplers.

**Flexible Balance** engine supports frequency domain harmonic balance analyses. It provides efficient and robust simulation for linear and weakly nonlinear circuits.

**Multi Carrier** engine extends single-carrier HB envelope analysis to better handle cases such as down-conversion mixers and circuits with multi-carriers. This choice appears only for the ENVLP analysis.

## Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)

Defines the analysis sweep range, sweep type, and any additional sweep points for a small-signal analysis. The PAC, Pnoise, PXF, PSP, and PSTB periodic small-signal analyses follow a PSS large-signal analysis. The QPAC, QPnoise, QPXF, and QPSP quasi-periodic small-signal analyses follow a QPSS analysis.

### Frequency Sweep Range (Hz)

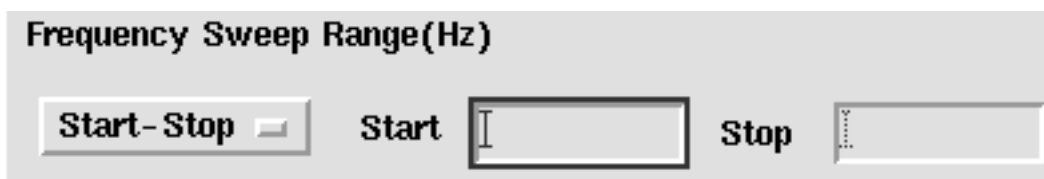
Defines the bounds for the small-signal analysis. Choices are: *Start-Stop*, *Center-Span*, and *Single-Point*.

For small-signal analyses following a swept PSS or QPSS analysis, *Single-Point* and *Freq* are the only *Frequency Sweep Range (Hz)* options for the small-signal analyses.

When you make a selection from the *Frequency Sweep Range (Hz)* cyclical field, the form fields change to let you specify appropriate data.

### **Start - Stop**

Defines the beginning and ending points for the sweep.



The screenshot shows a form titled "Frequency Sweep Range(Hz)". On the left, there is a dropdown menu with "Start-Stop" selected. To the right of the dropdown are the labels "Start" and "Stop", each followed by a rectangular input field. The "Start" input field is currently empty, and the "Stop" input field is also empty.

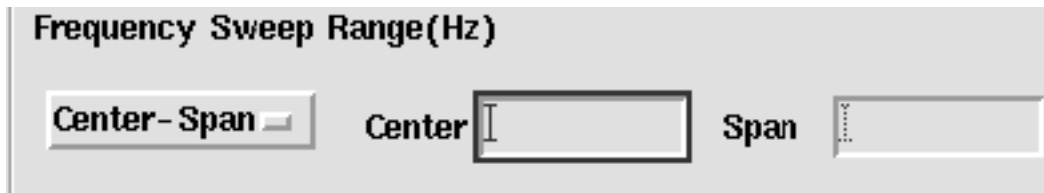
1. Highlight *Start-Stop*.

The form changes to let you type the start and stop points.

2. Type the initial point for the sweep in the *Start* field.
3. Type the final point in the *Stop* field.

### ***Center - Span***

Defines the center point for the sweep and its span.



The screenshot shows a dialog box titled "Frequency Sweep Range (Hz)". On the left, there is a dropdown menu with "Center-Span" selected. To the right of the dropdown are two input fields: "Center" and "Span". Both input fields are highlighted with a black border, indicating they are the active focus for data entry.

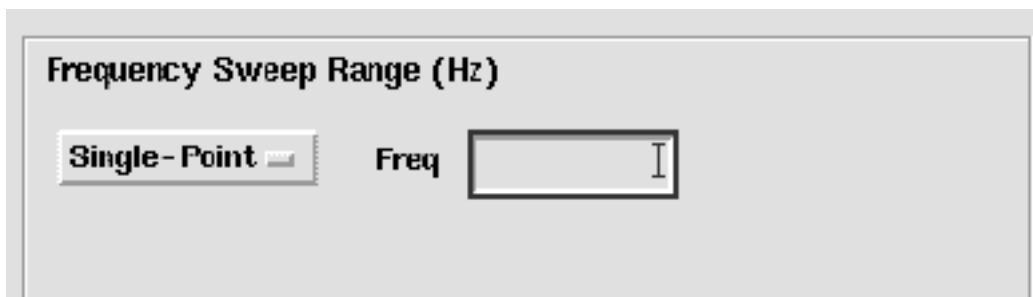
1. Highlight *Center-Span*.

The form changes to let you type the center point and span.

2. Type the midpoint for the sweep in the *Center* field.
3. Type the span in the *Span* field.

### ***Single - Point and Freq***

Defines the frequency range as a single point and prompts you for the point value.



The screenshot shows a dialog box titled "Frequency Sweep Range (Hz)". On the left, there is a dropdown menu with "Single-Point" selected. To the right of the dropdown is a single input field labeled "Freq". This input field is highlighted with a black border, indicating it is the active focus for data entry.

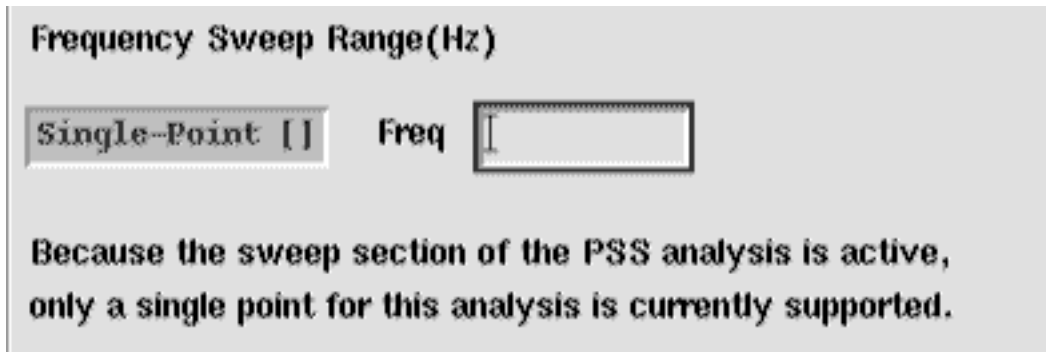
1. Highlight *Single-Point*.

The form changes to let you type the frequency.

2. Type the specific frequency for the small-signal analysis.

### **Single - Point and Freq Following Swept PSS or Swept QPSS Analysis**

When small-signal analyses follows a swept PSS or QPSS analysis, the only *Frequency Sweep Range (Hz)* option for the small-signal analyses is *Single-Point* and *Freq*.



Frequency Sweep Range(Hz)

Single-Point [ ]    Freq [ ]

Because the sweep section of the PSS analysis is active,  
only a single point for this analysis is currently supported.

- In the *Freq* field, type the specific frequency for the small-signal analysis that follows each pass of a PSS or QPSS sweep.

### **Sweep Type**

Specifies whether the small-signal sweep is linear, logarithmic, or automatic.

#### **Linear**

Specifies a linear sweep.



Sweep Type

Linear [ ]     Step Size    [ ]

Number of Steps

1. Select *Linear*.

The form changes to let you type either the step size or the total number of points (steps).

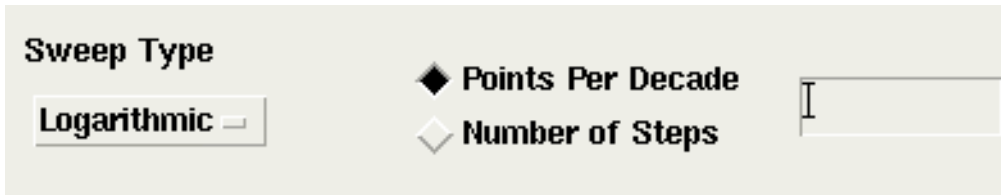
2. Do one of the following:

- Highlight *Step Size* and type the size of each step in the field.
- Highlight *Number of Steps* and type the number of steps in the field.



### **Logarithmic**

Specifies a logarithmic sweep.



1. Select *Logarithmic*.

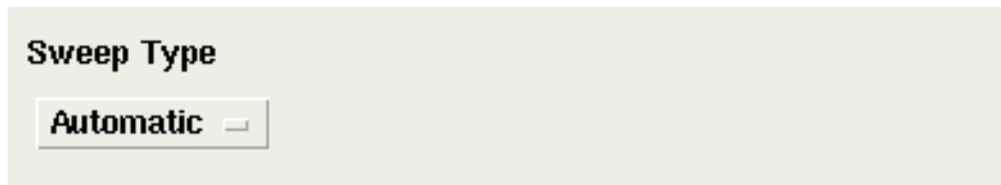
The form changes to let you type either the number of points per decade or the total number of points (steps).

2. Do one of the following:

- Highlight *Points per Decade* and type the number of points per decade in the field.
- Highlight *Number of Steps* and type the number of steps in the field.

### **Automatic**

Lets the simulator determine whether the *Sweep Type* is *Linear* or *Logarithmic*.



- Select *Automatic*.

The *Sweep Type* is *Linear* if the ratio of *Start* to *Stop* values is less than 10 or *Logarithmic* if the ratio of *Start* to *Stop* values is 10 or higher.

### **Add Specific Points**

Specifies additional sweep points for the small-signal analysis.



Highlight *Add Specific Points* and type the additional sweep point values into the field. Separate them with spaces.

## Fund Frequency (ENVLP)

Specifies the frequency of the clock fundamental.

Type the frequency into the field.

## Fundamental Tones (PSS and QPSS)

The *Fundamental Tones* fields include the following list box, data entry fields, and data entry buttons. In addition, the PSS analysis includes the related *Beat Frequency*, *Beat Period*, and *Auto Calculate* buttons.

#	Name	Expr	Value	Signal	SrcId
1	fff	1G	1G	Large	PORT3
2	frf	1G	1G	Large	

### Fundamental Tones List Box

The *Fundamental Tones* list box displays information about every top-level tone in the circuit that has both a non-zero frequency or period value, and a non-zero amplitude value (absolute). The tones in the list box are arranged alphabetically by name.

For QPSS analyses, you can edit

- The *Signal* level designation
- The *Harms* (Harmonic Range) value

To edit values for a tone, highlight the tone in the list box then edit in the data entry fields.

For tones that are not at the top level of the schematic, you can manually create a tone entry by typing the pertinent information in the data entry fields.

For non-small-signal tones, each tone name and its correlated frequency value are used in the *Select from range* and *Array of coefficients* fields in the *Output Harmonics* and *Sidebands* sections of the PSS, PAC, Pnoise, PXF, and QPnoise small-signal analysis forms.

### Fundamental Tones List Box Terms and Data Entry Fields

- **Name** – Displays the name assigned to the tone. This tone name must be entered into the pertinent Component Description Format (CDF) fields of each source in the schematic with a tone. The current CDF name field prompts are “First frequency name”, “Second frequency name”, “Frequency name”, and “Frequency name for 1/period”.
- **Expr** – Displays the value or expression representing the frequency of a particular tone. The expression can also be a user variable or it can contain user variables. If the frequency for the tone is specified as a variable, the *Expr* field displays the name of the variable. Otherwise, the field displays the numerical value of the frequency.
- **Value** – Displays the evaluated value of the *Expr* field using the current values of the user variables.
- **Signal** – This cyclic field displays one of two values: *Large* or *Moderate*.

For QPSS analysis, you must select one *Large* tone. This is the only time you use the *Large* specification. Specify *Moderate* for all additional tones you want to include in the simulation.

For PSS analysis, *Moderate* appears for all tones in the simulation.

- **SrcId** – Displays the instance name of the source in the schematic where the tone is declared.
- **Harms** – Used *only* with QPSS analysis.

Specifies the range of harmonics, which in turn determines the maximum number of harmonics of the tone to be used during the simulation.

The *Harms* value must be 1 or higher for the tone to be included in the simulation. The default value is 1.

Cannot deal with AHDL sources unless they are done with inlined Spectre sources.

## Setting Up Tones for a QPSS Analysis

When you set up tones for a QPSS analysis,

- Designate one signal as the *Large* tone. Designate all other tones as *Moderate*.

By choosing *Large* as the *Signal* value for a tone, you specify that tone to be the *Large* or clock signal. Each QPSS analysis must have one tone set to be the *Large* signal.

Select your *Large* signal to be the signal that

- Causes the largest response in the circuit.
  - Is the least sinusoidal signal in the circuit.
  - Causes the most nonlinearity in the circuit.
- Designate a non-zero harmonic range (*Harms*) value for each tone.

Never set the harmonic range value to zero. If you do, the simulation does not run properly and you might get incorrect results.

Choose at least one harmonic for each signal that you want to include in a QPSS analysis. A signal with an harmonic range (*Harms*) value of 0 is ignored by the simulation. In general, when selecting a *Harms* value:

For the *Large* signal, in most cases, set the *Harms* value to be equal to or greater than 5. An harmonic range of 5 gives 11 harmonics (-5, ..., 0, ... +5) for the *Large* tone.

### *Important*

Be aware of the following information about the *Harms* value.

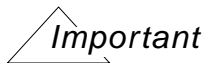
- The *Harms* value you use for the *Large* signal varies with the circuit you are analyzing. For example, for a down converting mixer, a *Harms* value of 1 is sufficient. Entering a large *Harms* value for the large tone does not affect the simulation run time.
- For *Moderate* tones, set the *Harms* value to be approximately 2 or 3. Setting the *Harms* value to 2 gives you up to the 3rd order intermodulation terms. Setting the *Harms* value to 3 gives you up to the 5th order intermodulation terms.

To obtain higher order intermodulation terms, you can increase the *Harms* value accordingly. However, for *Moderate* tones, increasing the *Harms* value increases the simulation run time. For example, when you specify *Harms* values of 5 for the *Large* signal and 2 for the *Moderate* signals, you get `maxharms = [ 5, 2 ]` which gives you 11 harmonics for the *Large* tone and 5 harmonics (-2, -1, 0, 1, 2) for the *Moderate* tone. As a result, you get noise from  $11 \times 5 = 55$  frequency sidebands.

- The *Harms* value you select depends on the degree of nonlinearity the signal causes. If the signal is not nonlinear, then a *Harms* value of 1 is sufficient. For a moderate signal, a few harmonics should be sufficient to accurately capture the nonlinearity. However, it is hard to determine before running the analysis what is sufficient. Generally, for a given *Harms* value, if increasing it does not significantly change the spectrum results, then the *Harms* value is high enough. In some situations, you could use a *Harms* value as high as 9. For a high *Harms* value, the algorithm still works, but not as efficiently.

### Fundamental Tones Data Entry Buttons

- **Clear/Add** – Clears the data entry fields for the purpose of manually adding a new tone to the list box. This button also resets the list box so that no line is currently selected.
- **Delete** – Deletes a tone selected in the *Fundamental Tones* list box.
- You cannot use the *Delete* button to delete tones that are specified in the schematic. Such tones are deleted by setting the CDF frequency or period field value to zero or blank and CDF amplitude value(s) to zero (absolute) or blank.
- **Update From Schematic** – Updates the values in the *Fundamental Tones* list box from the schematic.



Before you can update from the schematic, you must have performed a *Check and Save* on the design in the Schematic window.

### Using the Fundamental Tones Data Entry Fields and Buttons

Use the data entry fields below the list box to edit the information in the *Fundamental Tones* list box or to add new tones.

To specify a new tone,

1. Make sure there are no selected tones in the list box. If there is a selected tone, click the *Clear/Add* button.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

2. Type a value in any of the data entry fields (typically starting with the *Name* field).
3. As you advance to a second data entry field, a new tone line is added to the list box and is selected.
4. If required, select a value from the *Signal* cyclic field.
5. Continue editing each data entry field until all the pertinent information is complete.
6. The last value in the data entry fields is recorded in the list box when the next operation is performed in the analysis form or when you click the *Clear/Add* button. (Other operations that terminate the editing operation include moving the cursor off the Choosing Analyses form, clicking either the *OK* or *Apply* button, or changing a value in a non-related field.)

To edit an existing tone,

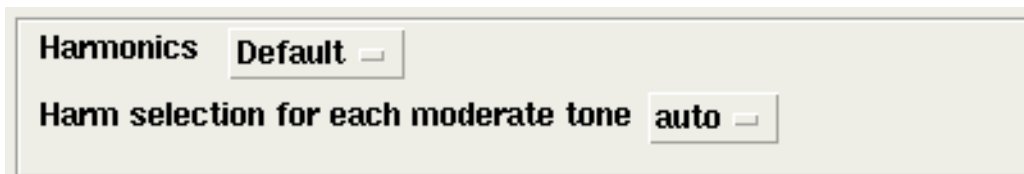
1. Select the tone in the list box.
2. The tone data appears in the data entry fields where you can edit it.
3. Modify a value in any one of the data entry fields. Values you cannot edit in the data entry fields (such as values specified in the schematic), are grayed out. Values originally specified in the schematic must be edited in the schematic.
4. The last modified value in the data entry fields is recorded in the list box when the next operation is performed in the analysis form or when you click the *Clear/Add* button.

## Harmonics (QPSS)

The *Harmonics* area enables you to select the important moderate tone harmonics for QPSS multitone simulation. This makes the QPSS analysis more efficient.

Choices for *Harmonics* are: *Default*, *View All* and *Select*.

### Default



The screenshot shows a control panel for the Harmonics section. It contains two dropdown menus. The first dropdown menu is labeled "Harmonics" and has "Default" selected. The second dropdown menu is labeled "Harm selection for each moderate tone" and has "auto" selected.

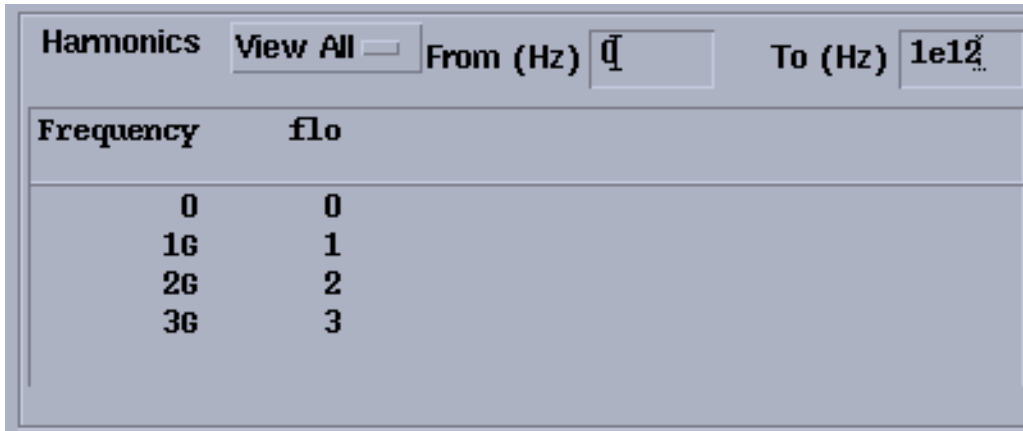
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

Select harmonics by choosing *auto*, *even*, *odd*, or *all* in the *Harm selection* for each *moderate tone* field. The *Harm selection* for each *moderate tone* field only appears when *Engine* is set to *Shooting*.

#### View All



Frequency	flo
0	0
1G	1
2G	2
3G	3

Lets you first enter a frequency range and then select harmonics from within this range of harmonic values.

1. In the *From (Hz)* and *To (Hz)* type-in fields, type the lower and upper values for the frequency range.

The form changes to display all input frequencies, their harmonics, and the intermodulations of the frequencies and harmonics for named, top-level, large and moderate input signals in the circuit whose frequencies fall within the specified frequency range. The first column is the frequency, and the remaining columns specify tone coefficients for each fundamental tone that contributed to the listed harmonic.

2. Select from the listed harmonics.

Click a harmonic in the list box to select it. Select adjacent harmonics by clicking and dragging with the mouse over the harmonics to select. Select harmonics that are not adjacent by holding the `Control` key down while you click the individual harmonics. (Deselect a harmonic by clicking a selected harmonic while you hold the `Control` key down.)

## Select

The screenshot shows a dialog box titled "Harmonics" with a "Select" button. Under "Method", there are four radio button options: "diamond" (selected), "funnel", "axis", and "arbitrary". To the right of these options is a "Boundary (int)" text field. Below the radio buttons is a "Harm selection for each moderate tone" section with an "auto" button.

**Method** specifies how the harmonics are selected: *diamond*, *funnel*, *axis*, or *arbitrary*.

- The *diamond* method has order  $P$ .
- The *axis* method selects harmonics along an axis; that is, there are no intermodulation harmonics.
- The *funnel* method is a combination of the *diamond* method and *axis* methods. It selects harmonics along an axis and intermodulation harmonics of order  $P$  or less. (The `selectharm` parameter determines the type of cut.)
- The *arbitrary* method allows you to specify indices of interest for each of the moderate fundamental tones. Guidance on the format to use appears in a message when you select this method.

**Boundary (int)** specifies the harmonic selection boundary.

### Harm selection for each moderate tone

When *Engine* is set to *Flexible Balance*, the *Harm selection for each moderate tone* and *Boundary (int)* options are replaced by an option called *MaxImOrder*.

**MaxImOrder** specifies the maximum intermodulation order. This value interacts with the chosen method to determine which harmonics are selected.

## Input Source and Reference Side-Band (Pnoise)

Identifies the noise generator and the reference sidebands to use for the Pnoise simulation.

The Reference Side-Band field specifies which conversion gain to use when the Spectre RF simulation computes the input-referred noise, noise factor, and noise figure.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

The designated reference sidebands, as well as the sideband zero, are included in the pool of sidebands used in noise calculations. For example, for the *ne600* test schematic,

- If `refsideband=-1` and `sidebands=[-2]`, then `Pnoise` computes contributions from sidebands -2, -1, and 0.
- If `refsideband=-2` and `sidebands=[-2]`, then Spectre RF computes contributions from sidebands -2 and 0.
- See “[Sidebands \(PAC, Pnoise, and PXF\)](#)” on page 78 for information on selecting sidebands.

The output total noise is different for the two simulation setups. The input-referred noise, noise factor, and noise figure are also different. Pnoise analysis internally includes the `refsideband` as a contribution to the total noise. This inclusion is not reflected in the netlist.

### Input Source

Choices for Input Source are: *voltage*, *current*, *port*, or *none*.

#### Voltage



The screenshot shows a dialog box titled "Input Source". On the left, there is a dropdown menu currently showing "voltage". To the right of the dropdown is the text "Input Voltage Source" followed by an empty text input field. To the right of the input field is a button labeled "Select".

1. Select *voltage* from the *Input Source* cyclic field.  
The *Reference side-band* fields appear.
2. Either type the name of the noise voltage generator in the *Input Voltage Source* field or click *Select* and then click the generator in the schematic.

#### Current



The screenshot shows a dialog box titled "Input Source". On the left, there is a dropdown menu currently showing "current". To the right of the dropdown is the text "Input Current Source" followed by an empty text input field. To the right of the input field is a button labeled "Select".

1. Select *current* from the *Input Source* cyclic field.

The *Reference side-band* fields appear.

2. Either type the name of a noise current generator in the *Input Current Source* field or click *Select* and then click the generator in the schematic.

### Port



1. Select *port* from the *Input Source* cyclic field.

The *Reference side-band* fields appear.

2. Either type the name of a port in the *Input Port Source* field or click *Select* and then click the port in the schematic.

When you select *port*, the analysis computes the noise voltage across the port and subtracts the contribution of this port in noise figure calculations.

### None



- Select *none* from the *Input Source* cyclic field.

When you select none, there are certain calculations you cannot perform. For example, input referred noise.

### Reference Side-Band

Reference Side-Band specifies which conversion gain to use when the Spectre RF simulation computes the input-referred noise, noise factor, and noise figure.

Choices are *Enter in field* or *Select from list*.

**Enter in Field**

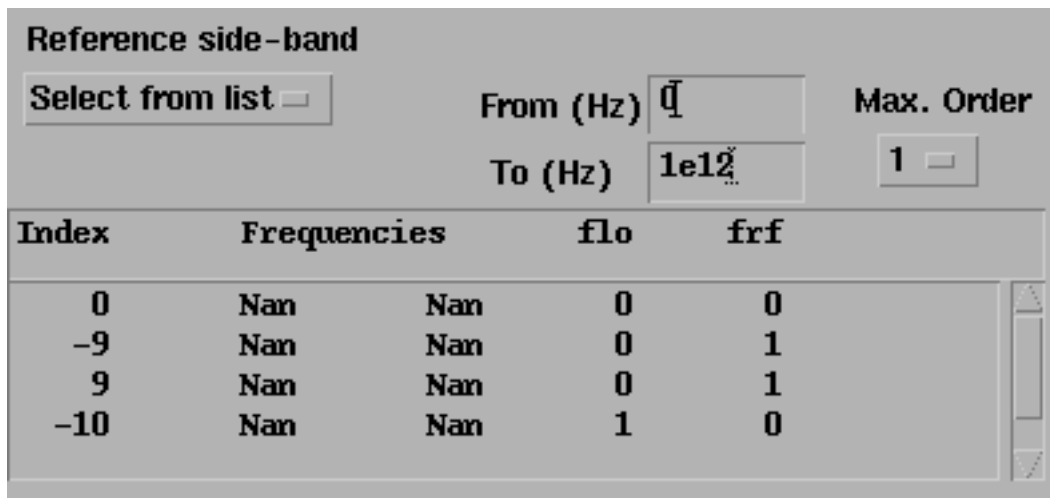


- Select *Enter in field* from the *Reference side-band* cyclic field.
- Type an integer value in the field.
  - 0 for amplifiers and filters
  - $-K$  for down converters
  - $+K$  for up converters

where  $K$  is the mixing harmonic.

**Select From List**

Lets you first enter a frequency range and then select sidebands from within this range of sideband values.



Depending on the selection in the **Sweep type** cyclic field, both the values in the list box and the values you specify are either *absolute* values or they are *relative* to the fundamental.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

To specify the listed sidebands,

1. In the *From (Hz)* and *To (Hz)* type-in fields, type the lower and upper values for the frequency range.

The sideband frequencies displayed in the list box are within this range of frequencies.

2. From the *Max. Order* cyclic field select the maximum order of harmonics that contribute to the sidebands.

If, for example, you select 3 as the *Max. Order* value, the sum of the *absolute* values of the tone coefficients contributing to the sidebands in the list box must be less than or equal to three.

In the list box, the first column is the index of a sideband, the second and third columns list the frequency range of the sideband. The remaining columns list tone coefficients for each fundamental tone that contributed to the listed sideband.

To select from the listed sidebands,

- Click a sideband in the list box to select it.

Select adjacent sidebands by clicking and dragging with the mouse over the sidebands to select. Select sidebands that are not adjacent by holding the `Control` key down while you click the individual sidebands. (Deselect a sideband by clicking a selected sideband while you hold the `Control` key down.)

## Input Source and Reference Side-Band (QPnoise)

Identifies the noise generator and the reference sidebands vector for the QPnoise simulation.

### Input Source

Choices for *Input Source* are: *voltage*, *current*, *port*, or *none*.

### Voltage



The screenshot shows a dialog box titled "Input Source". On the left, there is a dropdown menu with "voltage" selected. To the right of the dropdown is a text input field labeled "Input Voltage Source" and a "Select" button.

1. Select *voltage* from the *Input Source* cyclic field.

2. Either type the name of the noise voltage generator in the *Input Voltage Source* field or click *Select* and then click the generator in the schematic.

### Current



Input Source  
current Input Current Source [ ] Select

1. Select *current* from the *Input Source* cyclic field.
2. Either type the name of a noise current generator in the *Input Current Source* field or click *Select* and then click the generator in the schematic.

### Port



Input Source  
port Input Port Source [ ] Select

1. Select *port* from the *Input Source* cyclic field.
2. Either type the name of a port in the *Input Port Source* field or click *Select* and then click the port in the schematic.

### None



Input Source  
none

- Select *none* from the *Input Source* cyclic field.

When you select none, there are certain calculations you cannot perform. For example, input referred noise. *Reference Side-Band* is not available when you select *none* for *Input Source*.

## Reference Side-Band

*Reference side-band* specifies which conversion gain to use when the Spectre RF simulation computes the input-referred noise, noise factor, and noise figure. *Reference side-band* is available with all *Input Source* choices except *none*.

For QPnoise analysis, the *Reference Side-Band* field value is a vector. For example, 1 0 0.

Choices are *Enter in field* or *Select from list*.

### *Enter in Field*



Select *Enter in field* from the *Reference side-band* cyclic field and type a vector value into the field.

When the input and output are at the same frequency, use the zero vector

[0 0 ...]

- When you do not use the zero vector, the single sideband noise figure is computed.

### Select From List

Lets you first enter a frequency range and then select sidebands from within this range of sideband values.

Index	Frequencies	flo	frf
0	Nan	Nan	0
-9	Nan	Nan	1
9	Nan	Nan	1
-10	Nan	Nan	0

Depending on the selection in the *Sweep* cyclic field, both the values in the list box and the values you specify are either *absolute* values or they are *relative* to the fundamental.

To specify the listed sidebands,

1. First, in the *From (Hz)* and *To (Hz)* type-in fields, type the lower and upper values for the frequency range.

The sideband frequencies displayed in the list box are within this range of frequencies.

2. Then, from the *Max. Order* cyclic field select the maximum order of harmonics that contribute to the sidebands.

If, for example, you select 3 as the *Max. Order* value, the sum of the *absolute* values of the tone coefficients contributing to the sidebands in the list box must be less than or equal to three.

In the list box, the first column is the index of a sideband, the second and third columns list the frequency range of the sideband. The remaining columns list tone coefficients for each fundamental tone that contributed to the listed sideband.

To select from the listed sidebands,

- Click a sideband in the list box to select it.

Select adjacent sidebands by clicking and dragging with the mouse over the sidebands to select. Select sidebands that are not adjacent by holding the `Control` key down while

you click the individual sidebands. (Deselect a sideband by clicking a selected sideband while you hold the `Control` key down.)

## Modulated Analysis (PAC, PXF) — One of the Specialized Analyses

Measures AM and PM small-signal effects for the PAC and PXF analyses. When you select *Modulated* in the *Specialized Analyses* field, the Choosing Analyses form changes to let you enter more information.

The *Modulated Analysis* fields for the PXF analysis include the following.

The screenshot shows a form titled "Specialized Analyses". It contains the following fields and controls:

- A dropdown menu labeled "Modulated" with a downward arrow.
- A dropdown menu labeled "Output Type" with the value "SSB/AM/PM" and a downward arrow.
- A text input field labeled "Input Modulated Harmonic List" containing the value "1", followed by a "Choose" button.
- A text input field labeled "Output Modulated Harmonic" containing the value "1", followed by a "Choose" button.

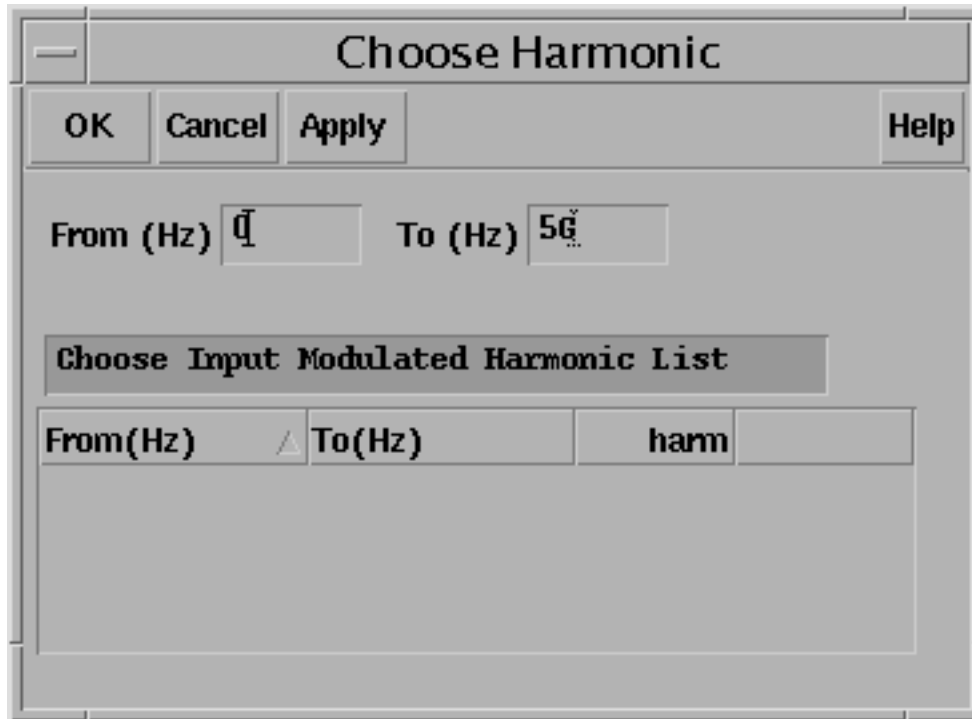
### Modulated Analysis for PXF Terms and Data Entry Fields

- **Output Type** – This cyclic field displays one of two values: *SSB* and *SSB/AM/PM*.
- **Input Modulated Harmonic List** – Lists the harmonic indexes.
- **Input Upper Sideband** – (Displays when you choose *SSB* for *Output Type*.) Click *Choose* to display the Choose Harmonic pop up.
- **Output Modulated Harmonic** – (Displays when you choose the *Output Type SSB/AM/PM*.) Click *Choose* to display the Choose Harmonic pop up.

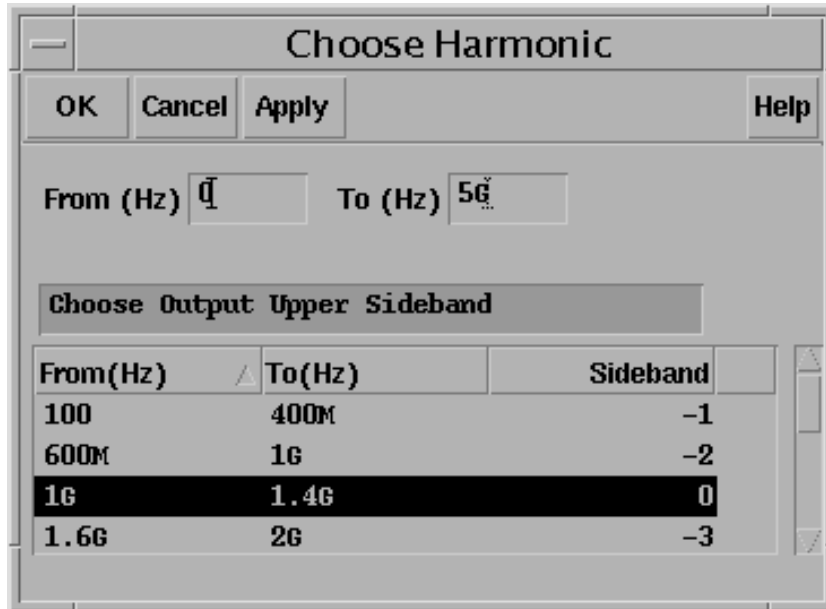


### Choose Harmonic Pop Up

Selects harmonics or sidebands for the analysis. Display the Choose Harmonic pop up from *Modulated Analysis* with the *Choose* button.



or



### **Choose Output Modulated Harmonic List Box and Data Entry Fields**

The *Choose Output Modulated Harmonic* list box displays the harmonic indexes and associated frequency ranges to select from. Changing the values in the *From (Hz)* and *To (Hz)* fields, changes the harmonic indexes and frequency ranges displayed in the list box.

**From (Hz) and To (Hz) fields** - The upper and lower bounds for the frequency range.

**Harm** – Displays the harmonic index, the integer which is multiplied by the fundamental to calculate the harmonic frequency.

For PSS analysis of port1 named RF with an harmonic index of 1, given the PSS fundamental of 900 MHz, port1 is analyzed from 901 MHz to 1000 MHz. For QPSP analysis, the computation is more complicated because there are more fundamentals and the harmonic specification is a vector of indexes.

**From (Hz)** - The lower bound for the frequency range associated with the harmonic.

**To (Hz)** - The upper bound for the frequency range associated with the harmonic.

### **Choose Input Upper Sideband List Box and Data Entry Fields**

The *Choose Input Upper Sideband* list box displays the harmonic indexes and associated frequency ranges to select from. Changing the values in the *From (Hz)* and *To (Hz)* fields, changes the harmonic indexes and frequency ranges displayed in the list box.

**From (Hz) and To (Hz) fields** - The upper and lower bounds for the frequency range.

**Sideband** – Displays the sidebands, the integers which are the periodic small-signal output frequencies of interest.

**From (Hz)** - The lower bound for the frequency range associated with the sideband.

**To (Hz)** - The upper bound for the frequency range associated with the sideband.

### **Noise Type (Pnoise)**

Specifies the type of noise to compute. Choices are: *jitter*, *modulated*, *sources*, and *timedomain*.

#### **Jitter (for a driven circuit)**

Measures PM jitter at the output for a driven circuit.

The screenshot shows a dialog box titled "Noise Type" with a dropdown menu set to "jitter". Below the dropdown, the text "jitter: jitter measurement at the output" is displayed. A text field contains "PM jitter for driven circuit". Below this, there are three columns: "Signal", "Threshold Value", and "Crossing Direction". The "Signal" field is empty, the "Threshold Value" field contains "1", and the "Crossing Direction" dropdown is set to "all".

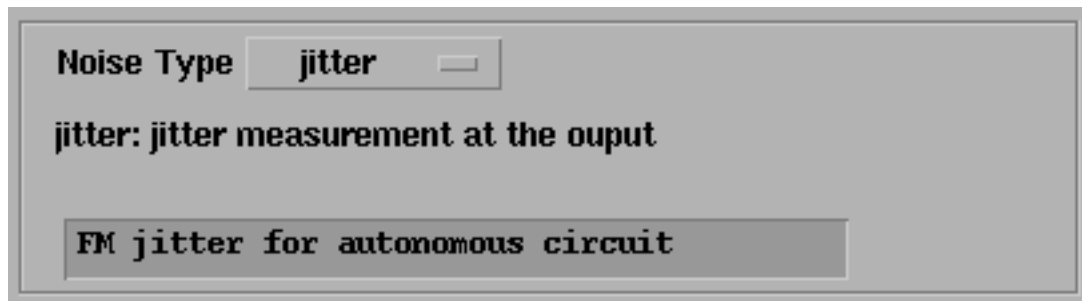
1. Select *jitter* from the *Noise Type* cyclic field.

The *Signal* field displays the signal to measure.

2. In the *Threshold Value* field, type the value where PM jitter is measured when the signal crosses this value.
3. From the *Crossing Direction* cyclic field, select the transition where jitter is measured.

### Jitter (for an autonomous circuit)

Measures FM jitter at the output for an autonomous circuit.



### Modulated

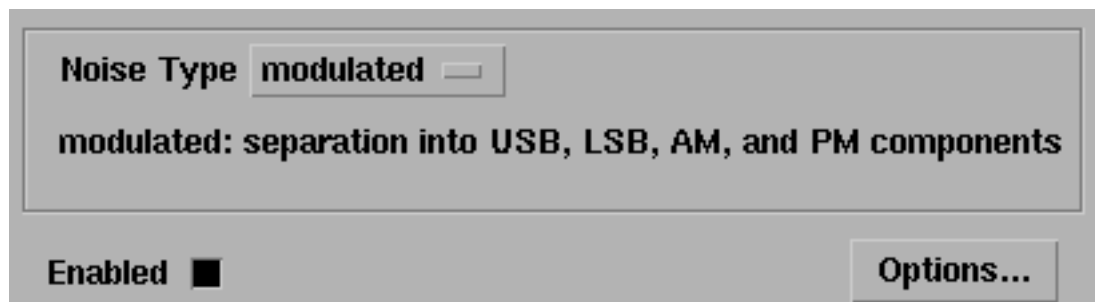
Separates noise into AM and PM components. It also calculates USB and LSB noise.

1. Select *modulated* from the *Noise Type* cyclic field.

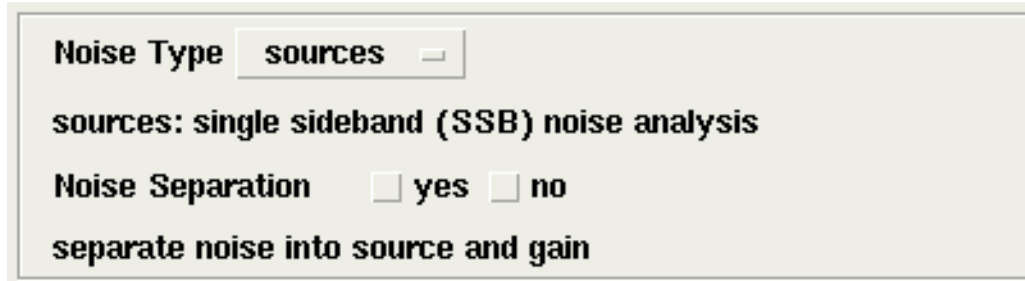
The following message displays:

When Noise Type is "modulated," Sweeptype must be "relative."

2. Choose *Close* to dismiss the message box and select relative for Sweeptype. See "[Sweeptype \(Pnoise\)](#)" on page 99 for more information.



## Sources



The screenshot shows a dialog box with the following content:

Noise Type

**sources: single sideband (SSB) noise analysis**

Noise Separation  yes  no

**separate noise into source and gain**

1. Select *sources* from the *Noise Type* cyclic field.

When you select *sources*, the simulation computes the total time-average noise at an output over a given frequency range. It computes the contribution of each noise source to the total noise at each frequency.

2. Choose whether *Noise Separation* runs or not.

This option determines whether the noise separation feature runs during the simulation. The possible values are *yes* and *no*.

*yes*                      Noise separation runs and the results are saved.

*no*                        Noise separation does not run.

## Timedomain

Computes the time-varying instantaneous noise power in a circuit with periodically driven components.

Noise Type **timedomain**

**timedomain: strobed noise analysis**

Noise Skip Count

Number of Points

Add Specific Points

Enabled  **Options...**

1. Select *timedomain* from the *Noise Type* cyclic field.
2. Type the noise skip count in the *Noise Skip Count* field.
3. (Optional) Highlight *Number of Points* and type the number of points in the *Number of points* field.
4. (Optional) Highlight *Add Specific Points* and type one or more points.

## Options

Opens the Options form.

Enabled  **Options...**

- Click *Options* to open the Options form for the selected analysis.

See [“Option Forms”](#) on page 172 for information on the Options forms.

## Oscillator (PSS)

Specifies that the circuit is an oscillator. When you highlight *Oscillator*, the form changes to let you enter additional information.

<b>Oscillator</b> <input checked="" type="checkbox"/>	<b>Oscillator node</b>	<input type="text"/>	<b>Select</b>
	<b>Reference node</b>	<input type="text"/>	<b>Select</b>
<b>Osc initial condition</b>	<input checked="" type="checkbox"/> <b>default</b>	<input type="checkbox"/> <b>linear</b>	
<b>Save osc PPV</b>	<input type="checkbox"/> <b>yes</b>	<input type="checkbox"/> <b>no</b>	
<b>Osc Newton method</b>	<input type="checkbox"/> <b>onetier</b>	<input type="checkbox"/> <b>twotier</b>	

- Either
  - Type the *Oscillator node* and *Reference node* names.
  - Click the corresponding *Select* and then click the node in the schematic.

If you leave the *Reference node* field empty, the name defaults to `gnd`.

The *OSC initial condition* option determines how the starting values for the oscillator are determined. The possible values are *default* and *linear*.

*default* This is the default value. A transient analysis controlled by the `tstab` setting is used to generate the initial guess of the solution.

*linear* A linear analysis at the DC solution is used to estimate the oscillation frequency and the amplitude. Then a transient analysis controlled by the `tstab` setting is performed using the estimated linear solution as a starting point.

With this approach, the large `tstab` value that would otherwise be required for high-Q oscillators can be reduced or eliminated.

The *linear* value is suited for linear oscillators such as LC and crystal oscillators.

The *Save osc PPV* option lets you save a perturbation projection vector (PPV) file. A PPV can be thought of as representing the oscillator's phase sensitivity to perturbations in the voltage or current at the nodes of the oscillator. The saved PPV file is used in phase noise calculations, such as when you prepare to generate a VCO macromodel.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

The PPV file is saved into the raw directory after simulation. For example, after running an autonomous circuit named `vco.ckt` with the *Save osc PPV* option set to *yes*, the PPV file is placed in the `vco.raw` directory. The PPV file is named `analysisID.td.ppv.pss`.

If the *Save osc PPV* option is set to *yes* and the PNOISE analysis *use ppv for osc* option is set to *yes* also, only the PPV file for the PNOISE analysis is saved.

The *Osc Newton method* option specifies that only the *onetier* or only the *twotier* method, but not both are to be used.

- In the **onetier** method, the frequency and voltage spectrum are solved simultaneously in one single set of nonlinear equations.
- In the **twotier** method, the nonlinear equations are split into two sets: the inner set of nonlinear equations solves the spectrum of node voltage equations; the outer set of nonlinear equations solves the oscillation frequency. The *twotier* method has a larger convergence zone because its convergence is slightly more robust. This method is however computationally more expensive than the *onetier* method.

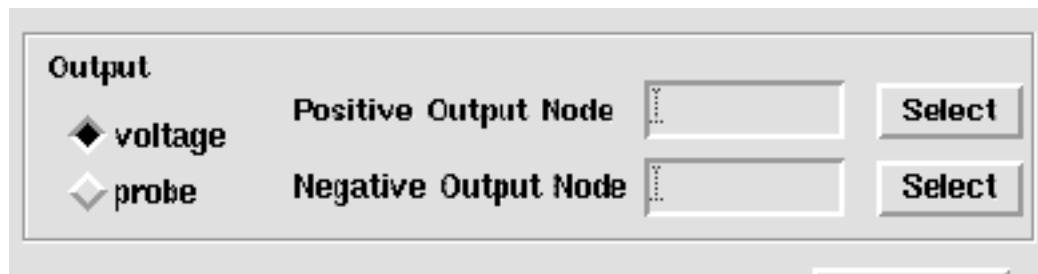
By default, Spectre RF runs *onetier* first for *n* iterations; if necessary, it runs *twotier* for the next *n* iterations (*n* is set by *maxperiods*).

### Output (PXF and QPXF)

Specifies the output transfer function for the PXF and QPXF analyses. Choices are *voltage* or *probe*.

#### Voltage

In PXF analysis, the voltage across the two nodes that you specify is the output for each transfer function.



1. Highlight *voltage*.



The form changes to let you specify a *Positive Output Node* and a *Negative Output Node*.

2. Specify the node names by either

- Typing the node name into the *Positive Output Node* or *Negative Output Node* fields.
- Clicking the adjacent *Select* button and then clicking the node in the schematic.

If you leave the *Negative Output Node* field empty, it defaults to `gnd`.

### Probe

In PXF analysis, the current through the point that you select is the output of each transfer function.



1. Highlight *Probe*.

The form changes to let you specify an output voltage source.

2. Specify the output voltage source by either

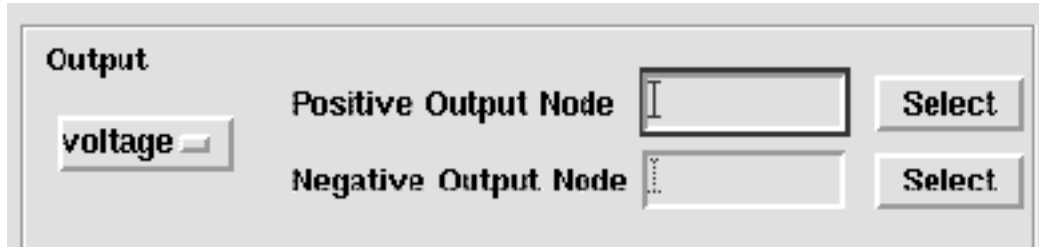
- Typing the instance name into the *Output Probe Instance* field.
- Clicking the adjacent *Select* button and then clicking the node in the schematic.

### Output (Pnoise and QPnoise)

The *Output* cyclic field lets you specify the output for the Pnoise and QPnoise analyses. Choices are *voltage* or *probe*.

## Voltage

The Pnoise or QPnoise analysis computes the noise voltage across the two nodes.



The screenshot shows a dialog box titled "Output". On the left, there is a dropdown menu with "voltage" selected. To the right of the dropdown are two rows of input fields. The first row is labeled "Positive Output Node" and has a text input field followed by a "Select" button. The second row is labeled "Negative Output Node" and has a text input field followed by a "Select" button.

1. Select *Voltage*.

The form changes to let you specify a *Positive Output Node* and a *Negative Output Node*.

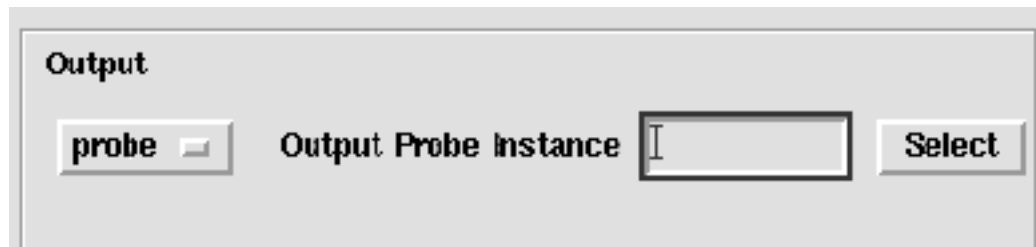
2. Specify the nodes by either

- Typing a node name into the type-in field.
- Clicking the adjacent *Select* button and then clicking the appropriate node in the schematic.

If you leave the *Negative Output Node* field empty, it defaults to `gnd`.

## Probe

The Pnoise or QPnoise analysis computes the noise voltage across the port. The noise contribution of the port is subtracted during the noise figure calculation.



The screenshot shows a dialog box titled "Output". On the left, there is a dropdown menu with "probe" selected. To the right of the dropdown is a text input field labeled "Output Probe Instance" followed by a "Select" button.

1. Select *Probe*.

The form changes to let you specify the *Output Probe Instance*.

2. Specify the node either by

- ❑ Typing the node name into the type-in field.
- ❑ Clicking the adjacent *Select* button and then clicking the appropriate node in the schematic.

## Output Harmonics (PSS and ENVLP)

Lets you select output harmonics. When you select from the *Output Harmonics* cyclic field, the form changes to let you specify appropriate data.

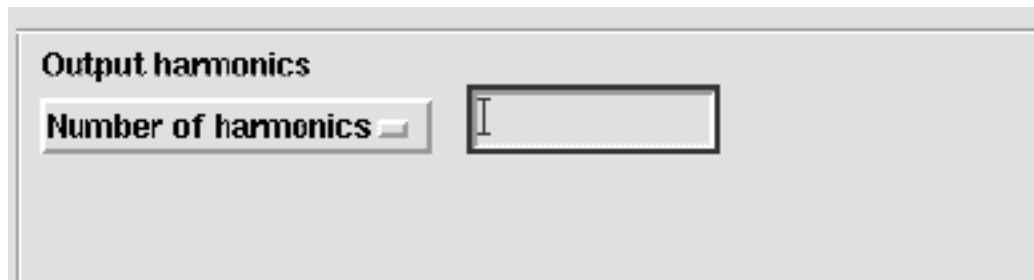
For PSS analysis, choices are: *Number of Harmonics*, *Select from Range*, *Array of Coefficients*, *Array of Indices*.

For ENVLP analysis, choices are: *Number of Harmonics*, and *Array of Indices*.

You can use the separate choices in the *Output Harmonics* cyclic field in combination. For example, if you add a harmonic using *Array of Coefficients*, the value you add appears in the *Select from Range* list box and as a currently active index field for the *Array of Indices*.

## Number of Harmonics (PSS and ENVLP)

Specifies a single value for the number of output harmonics.



The screenshot shows a form titled "Output harmonics". Below the title is a dropdown menu currently displaying "Number of harmonics" with a small arrow icon to its right. To the right of the dropdown menu is a rectangular text input field with a vertical cursor inside, ready for text entry.

1. Select *Number of Harmonics*.

The form changes to let you specify a single integer value.

2. Type the number of harmonics into the *Number of harmonics* field.

Harmonics in this field are relative to the value of the fundamental. Type 0 into the *Number of harmonics* field to specify no harmonics.

### Select from Range (PSS)

Lets you enter a frequency range and select harmonics from within this range.

<b>Output harmonics</b>		From (Hz) <input style="width: 50px;" type="text" value="0"/>	Max. Order
<input type="checkbox"/> <b>Select from range</b>		To (Hz) <input style="width: 50px;" type="text" value="1e12"/>	<input style="width: 30px;" type="text" value="1"/>
Index	Frequency	flo	frf
0	0	0	0
9	900M	0	1
10	16	1	0

1. Select *Select from Range*.

The form changes to display a cyclic field and two data entry fields.

2. In the *From (Hz)* and *To (Hz)* fields, type the lower and upper values for the frequency range.
3. From the *Max. Order* cyclic field, select the maximum order of harmonics that contribute to the output harmonics.

The form changes to display harmonics matching these specifications in the list box. The first column is the index of a harmonic, the second column specifies its frequency, and the remaining columns specify tone coefficients for each fundamental tone that contributed to the listed harmonic.

If, for example, you select 5 in the *Max. Order* cyclic field, the sum of the absolute values of the tone coefficients contributing to the listed output harmonics is less than or equal to five. Negative integers displayed in the list box represent the tone coefficients of harmonics below the fundamental and positive integers represent the tone coefficients of harmonics above the fundamental.

4. Click a harmonic in the list box to select it.

Select adjacent harmonics by clicking and dragging with the mouse over the harmonics to select. Select harmonics that are not adjacent by holding the `Control` key down while

you click the individual harmonics. (Deselect a harmonic by holding the `Control` key down and clicking the harmonic.)

### Array of Coefficients (PSS)

Lets you specify output harmonics by entering their tone coefficients.

Index	Frequency
	fff

1. Select *Array of Coefficients*.

The form changes to display a data entry field.

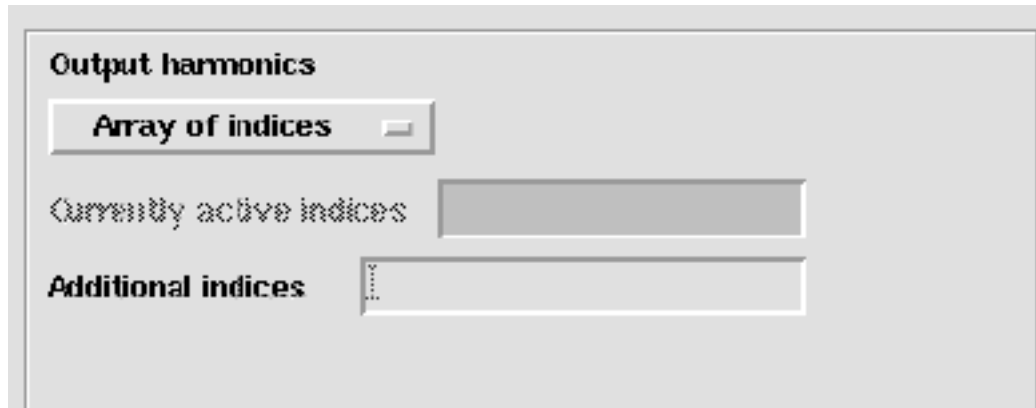
2. Type tone coefficients separated by spaces in the *Tone Coefficients* field and click *Clear/Add*.

Values that appear in the list box are absolute values or relative to the fundamental, depending on the selection in the *Sweep type* cyclic field.

To delete a harmonic, select it in the list box and click *Delete*.

## Array of Indices (PSS and ENVLP)

Lets you specify an array of harmonics by entering their indices.



The screenshot shows a dialog box titled "Output harmonics". At the top, there is a radio button labeled "Array of indices" which is selected. Below this, there are two text input fields. The first is labeled "Currently active indices" and is currently empty. The second is labeled "Additional indices" and contains a vertical ellipsis symbol (three dots) indicating a list of values.

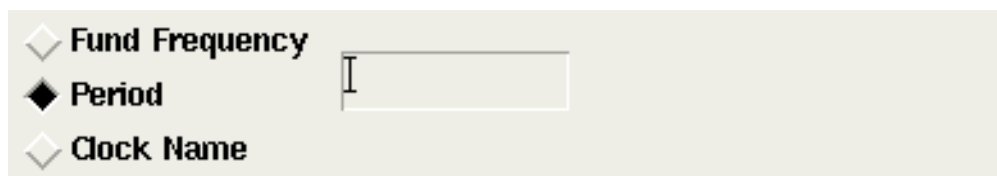
1. Select *Array of Indices*.

The form changes to display a data entry field. Any currently selected indices appear in the *Currently active indices* field.

2. Type the additional harmonic indices separated by spaces and in any order in the *Additional Indices* type-in field.

## Period (ENVLP)

Specifies the period (or, for autonomous circuits, the estimated period) of the clock fundamental.



The screenshot shows a dialog box for "Period (ENVLP)". It has three radio buttons: "Fund Frequency" (unselected), "Period" (selected), and "Clock Name" (unselected). To the right of the "Period" radio button is a text input field with a vertical cursor.

## Periodic Stab Analysis Notification (PSTB)

Defines the beginning and end points or a single point to be used for the analysis. The choices are: *Start-Stop*, *Center-Span*, *Single-Point*.

## Start – Stop

Defines the beginning and ending points for the sweep.



Periodic Stab Analysis Notification

Start-Stop ▾    Start     Stop

1. Select *Start-Stop*.  
The form changes to let you type the start and stop points.
2. Type the initial point for the sweep in the *Start* field.
3. Type the final point in the *Stop* field.

## Center – Span

Defines the center point for the sweep and its span.



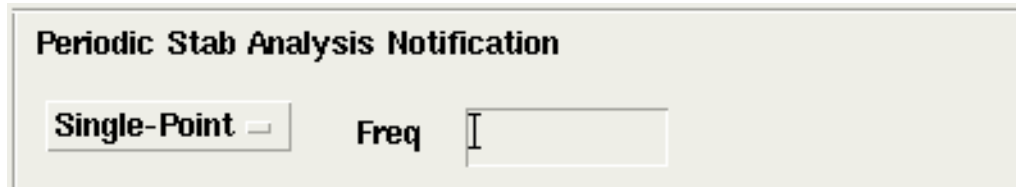
Periodic Stab Analysis Notification

Center-Span ▾    Center     Span

1. Select *Center-Span*.  
The form changes to let you type the center point and span.
2. Type the midpoint for the sweep in the *Center* field.
3. Type the span in the *Span* field.

## Single-Point

Defines the frequency range as a single point.



Periodic Stab Analysis Notification

Single-Point ▾      Freq

1. Select *Single-Point*.  
The form changes to let you type the frequency.
2. Type the specific frequency for the small-signal analysis.

## Probe Instance (PSTB)

In PSTB analysis, a `probe` component is placed in the feedback loop to identify and characterize the particular loop of interest. Introducing the `probe` component must not change the circuit characteristics.

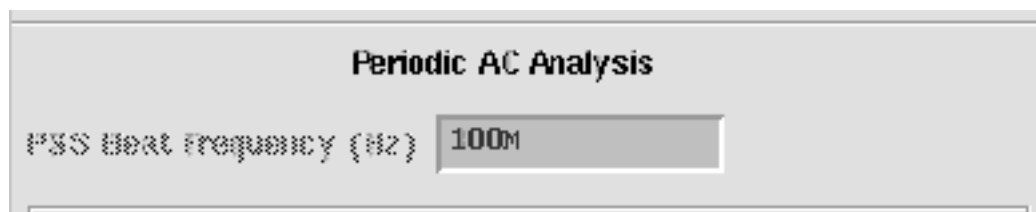


Probe Instance      

- Specify the probe instance by either
  - Typing it into the *Probe Instance* field.
  - Clicking the adjacent *Select* button and then clicking the node in the schematic.

## PSS Beat Frequency (PAC, PSTB, Pnoise, and PXF)

Displays the *Beat Frequency* for the initial PSS analysis.



Periodic AC Analysis

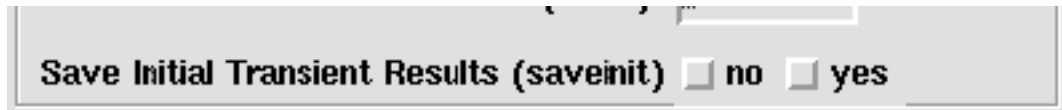
PSS Beat Frequency (Hz)



This is a display-only field. You cannot edit the information here.

### Save Initial Transient Results (PSS and QPSS)

Saves the initial transient waveforms.

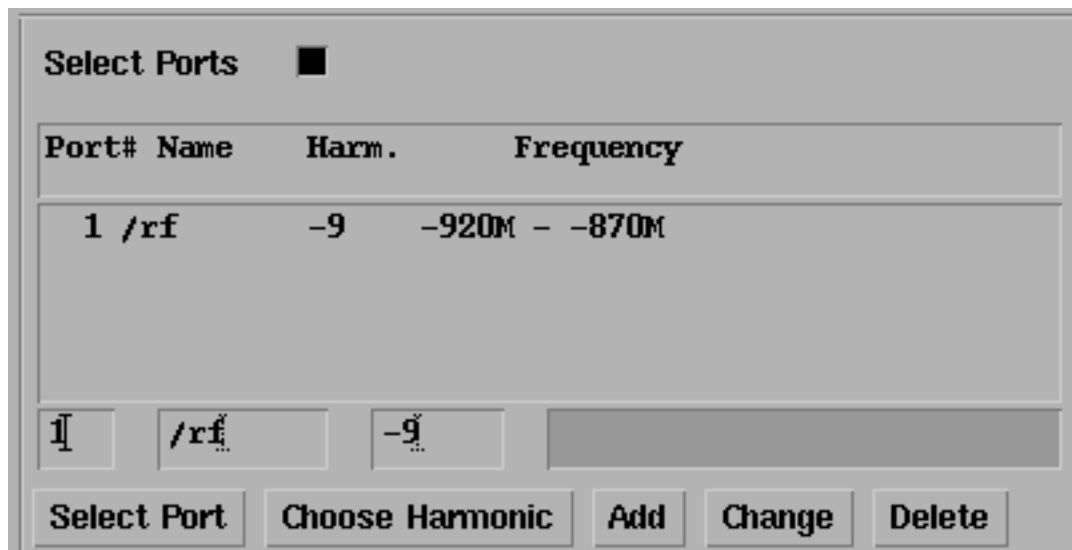


Highlight yes to save the initial transient waveforms. The default is *no*.

### Select Ports (PSP and QPSP)

Specifies the active ports for PSP and QPSP analyses.

The *Select Ports* fields include the following list box, data entry fields, and data entry buttons.



### Select Ports List Box

The *Select Ports* list box displays information about all selected ports. Ports in the list box are arranged numerically by the port numbers (*Port#*) that you assign.

## Select Ports List Box Terms and Data Entry Fields

The list box displays the ports that have already been specified for PSP or QPSP analysis. Enter data for new ports using the data entry fields and buttons below the list box. You can enter new ports or edit values for existing ports. Notice that the frequency field is grayed out and you cannot edit frequency values.

**Port#** – Displays the port number assigned to the port. In the list box, ports are numbered sequentially.

**Name** – Displays the name of the port in the schematic. Use the *Select Port* button to select a port from the schematic. The name of the selected port in the schematic displays in the editable field.

**Harm.** – Displays the harmonic index, the integer which is multiplied by the fundamental to calculate the harmonic frequency. Use the *Choose Harmonic* button to display a list of harmonics and frequencies to select from. The selected harmonic index displays in the editable field. The associated frequency range also displays.

For PSS analysis of port1 named RF with an harmonic index of 1, given the PSS fundamental of 900 MHz, port1 is analyzed from 901 MHz to 1000 MHz. For QPSP analysis, the computation is more complicated because there are more fundamentals and the harmonic specification is a vector of indexes.

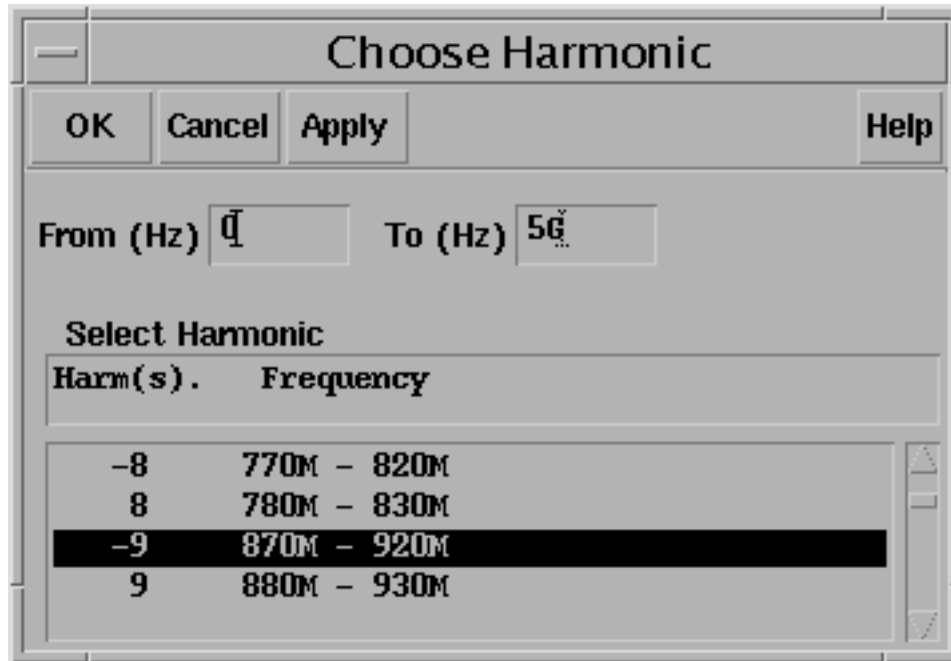
**Frequency** – Displays a range of frequency values associated with each harmonic index. Use the *Choose Harmonic* button to display a list of harmonics and frequencies to select from. The frequency range associated with the selected harmonic index value displays in the field. Notice that the Frequency field is grayed out and you cannot edit frequency range values.

## Select Ports Data Entry Buttons

- **Select Port** – Prompts you to select a port from the schematic. Displays the port's name from the schematic in the editable field.
- **Choose Harmonic** – Opens the Choose Harmonic Pop UP where you can select a harmonic for the port.
- **Add** – Adds a port to the list box using the information in the data entry fields.
- **Change** – Adds a modified port to the list box using the modified information in the data entry fields. Highlight a port in the list box to move its information to the data entry fields.
- **Delete** – Deletes a highlighted port from the list box.

## Choose Harmonic Pop Up

Selects a harmonic for a port. Open the Choose Harmonic pop up from *Select Ports*.



### Choose Harmonic List Box and Data Entry Fields

The *Select Harmonic* list box displays the harmonic indexes and associated frequency ranges to select from for the ports that have already been specified for PSP or QPSP analysis. Changing the values in the *From (Hz)*, *To (Hz)* and *Max. Order* fields, changes the harmonic indexes and frequency ranges displayed in the *Select Harmonic* list box.

**From (Hz) and To (Hz) fields** - The upper and lower bounds for the frequency range.

**Max. Order** - For QPSP analysis only. Displays the maximum order of harmonics that contribute to the harmonics.

**Harm(s)**. – Displays the harmonic index, the integer which is multiplied by the fundamental to calculate the harmonic frequency.

For PSS analysis of port1 named RF with an harmonic index of 1, given the PSS fundamental of 900 MHz, port1 is analyzed from 901 MHz to 1000 MHz. For QPSP analysis, the computation is more complicated because there are more fundamentals and the harmonic specification is a vector of indexes.

**Frequency** – Displays a range of frequency values associated with each harmonic index.

### Using the Select Ports Data Entry Fields and Buttons

Use the data entry fields below the list box to edit the information in the *Select Ports* list box or to add new ports.

To specify a new port,

1. Type an integer in the first field, the *Port#* field. It is directly above the *Select Port* button.

2. Click *Select Port* and follow the prompt at the bottom of the Schematic window.

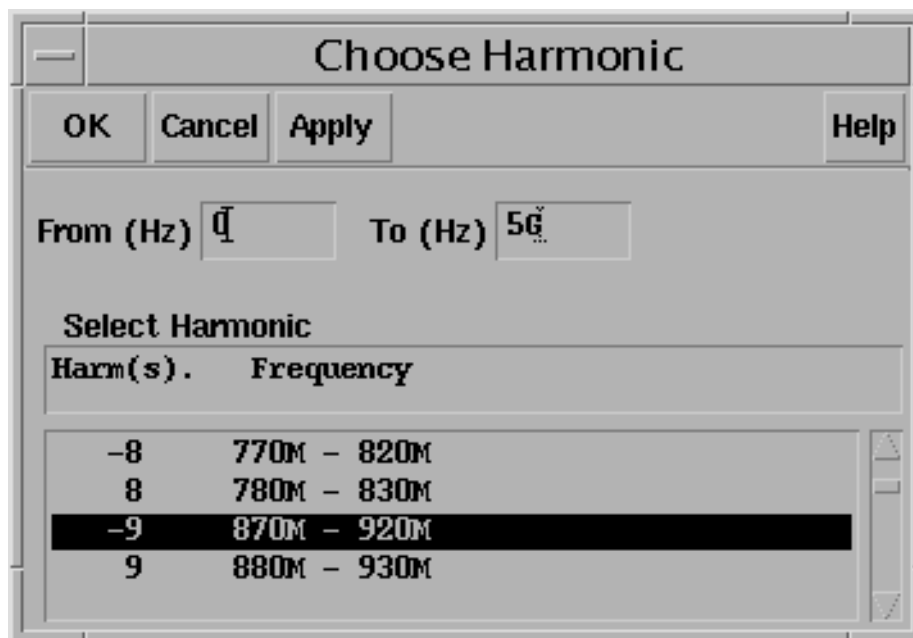
Select source...

3. In the Schematic window, click an appropriate port, for example */rf*.

*/rf* appears in the *Name* field.

4. Click *Choose Harmonic*.

5. The Choose Harmonic form displays with a list of harmonics (by index and frequency) for the *rf* port.



6. In the Choose Harmonic form, scroll through the list and highlight a harmonic to select it.

7. Click *OK*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

8. The Choose Harmonic form closes. In the *Select Ports* area of the Choosing Analyses form, the *Harm(s)* value of the selected harmonic displays in the *Harm.* field.
9. In the *Select Ports* area, click *Add*.
10. Information for the input port displays in the *Select Ports* list box.

Port#	Name	Harm.	Frequency
1	/rf	-9	-920M - -870M

1 /rf -9

Select Port Choose Harmonic Add Change Delete

To edit an existing port,

1. Highlight the port in the list box.
2. The port data appears in the data entry fields where you can edit it.
3. Modify a value in any one of the data entry fields. Values you cannot edit in the data entry fields (such as frequency range values), are grayed out.
4. Click *Change*.

The modified port data replaces the port in the list box.

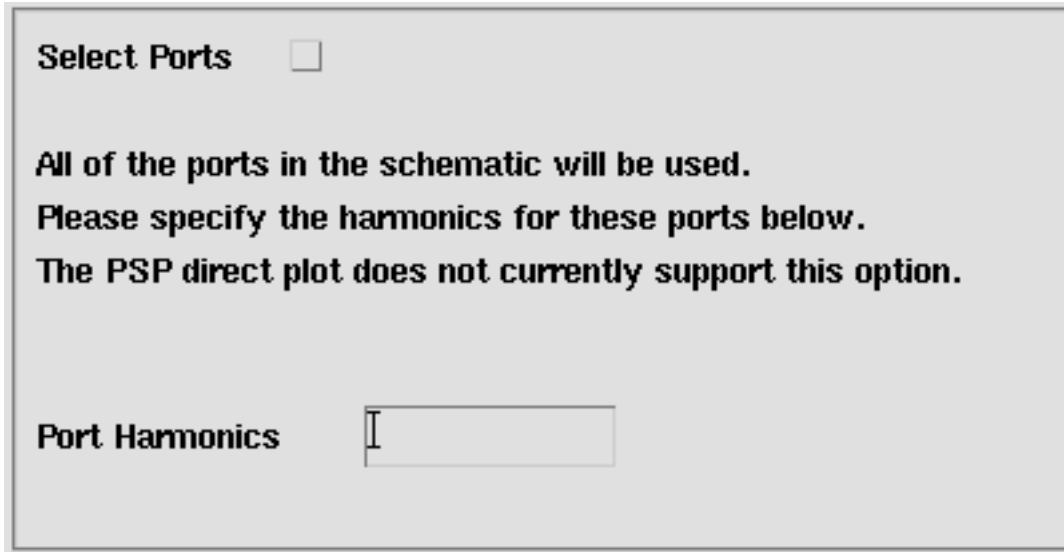
To delete an existing port,

1. Highlight the port in the list box.
2. The port data appears in the data entry fields.
3. Click *Delete*.

The port is removed from the list box.

To use all ports in the schematic,

1. Deselect *Select Ports*.



The screenshot shows a dialog box with a light gray background. At the top left, there is a label "Select Ports" followed by an unchecked checkbox. Below this, there are three lines of bold text: "All of the ports in the schematic will be used.", "Please specify the harmonics for these ports below.", and "The PSP direct plot does not currently support this option." At the bottom left, there is a label "Port Harmonics" followed by a rectangular text input field.

2. In the *Port Harmonics* field, type the harmonics for these ports separated by spaces.  
(PSP direct plot does not currently support this option.)

## Sidebands (PAC, Pnoise, and PXF)

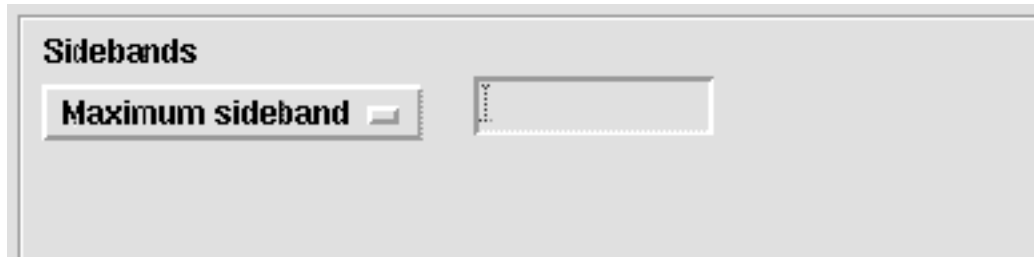
Lets you select the set of periodic small-signal output frequencies of interest. When you select from the *Sidebands* cyclic field, the form changes to let you specify appropriate data.

Choices are: *Maximum sideband*, *Select from Range*, *Array of Coefficients*, *Array of Indices*.

You can use the choices in the *Sidebands* cyclic field in combination. For example, if you add a sideband using *Array of Coefficients*, the sideband value you added appears in the *Select from Range* list box and as a *Currently active index* for the *Array of Indices*.

## Maximum sideband

Specifies the number of frequency conversion terms (values of k) to take into account.



Prompts you for a *Maximum sideband* value and automatically generates a sideband array of the form:

```
[–maximum sideband . . . 0 . . . +maximum sideband]
```

A typical analysis strategy is to begin by setting *Maximum sideband* to 7, the default value. Then increase the *Maximum sideband* value while observing the effect on output noise. If output noise changes, there is significant frequency conversion for values of k greater than 7. Continue to increase the *Maximum sideband* value until the output noise stops changing.

When you set *Maximum sideband* to zero, the reported output noise does not include any frequency conversion terms. For a fundamental oscillator, *Maximum sideband* must be at least 1 to see flicker noise upconversion. In general, small values for *Maximum sideband* are not recommended.

The *Maximum sideband* is ignored in HB small signal analysis when the value is larger than the *harms* or *maxharms* value of the large signal analysis.

### Select from range

Lets you first enter a frequency range and then select sidebands from within this range of sideband values.

Index	Frequency	flo	frf
0	Nan	0	0
-9	Nan	0	1
9	Nan	0	1
-10	Nan	1	0
10	Nan	1	0

Depending on the selection in the *Sweep type* cyclic field, both the values in the list box and the values you specify are either *absolute* values or they are *relative* to the fundamental.

To specify the listed sidebands,

1. In the *From (Hz)* and *To (Hz)* type-in fields, type the lower and upper values for the frequency range.

The sideband frequencies displayed in the list box are within this range of frequencies.

2. From the *Max. Order* cyclic field select the maximum order of harmonics that contribute to the sidebands.

If, for example, you select 3 as the *Max. Order* value, the sum of the *absolute* values of the tone coefficients contributing to the sidebands in the list box must be less than or equal to three.

In the list box, the first column is the index of a sideband, the second and third columns list the frequency range of the sideband. The remaining columns list tone coefficients for each fundamental tone that contributed to the listed sideband.

To select from the listed sidebands,

- Click a sideband in the list box to select it.



Select adjacent sidebands by clicking and dragging with the mouse over the sidebands to select. Select sidebands that are not adjacent by holding the `Control` key down while you click the individual sidebands. (Deselect a sideband by clicking a selected sideband while you hold the `Control` key down.)

### Array of Coefficients

Lets you specify sidebands by typing their tone coefficients.

Index	Frequency	flo	frf
-------	-----------	-----	-----

To specify sidebands by using tone coefficients,

1. Type the tone coefficients, separated by spaces, into the *Tone Coefficients* type-in field.
2. Click *Clear/Add* to add the tone coefficient to the list box.

Values that appear in the list box are *absolute* values or are *relative* to the fundamental, depending on the selection in the *Sweeptype* cyclic field.

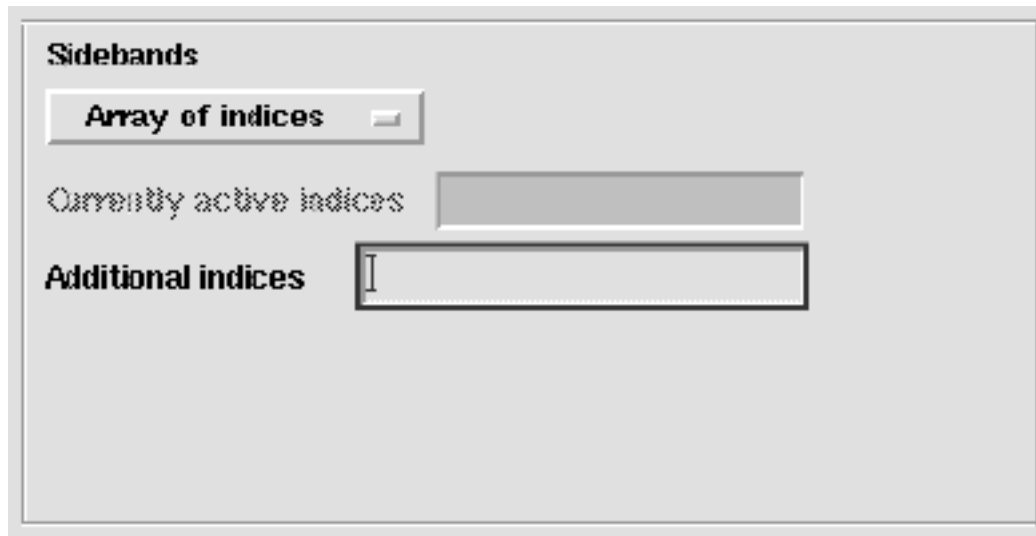
3. Place the specified sideband above or below the fundamental using the *upper* cyclic field.

To delete a sideband,

- Select the sideband in the list box and click *Delete*.

## Array of Indices

Lets you specify an array of sidebands by entering their indices.



To specify an array of sidebands by using indices,

1. Choose *Array of Indices*.

Currently selected indices appear in the *Currently active indices* field.

2. Type the indices of the sidebands you want to specify into the *Additional indices* field.

You can type the additional sideband indices in any order. Separate the indices with spaces.

## Sidebands (QPAC, QPnoise, and QPXF)

Lets you select the set of quasi-periodic small-signal output frequencies of interest for the QPAC, QPnoise, or QPXF analyses.

For the quasi-periodic analyses, sidebands are vectors specified with the `sidevec` and `clockmaxharm` parameters.

When you select from the *Sidebands* cyclic field, the form changes to let you specify appropriate data. Choices are: *Maximum clock order*, *Select from range*, and *Array of coefficients*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

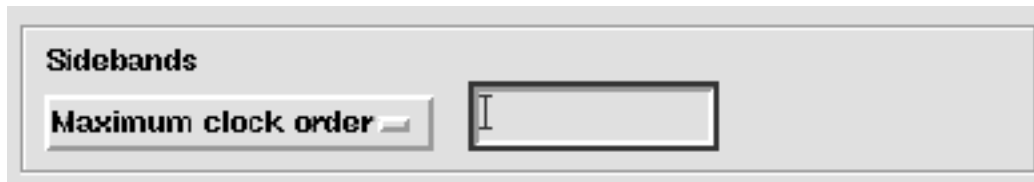
## Spectre RF Simulation Form Reference

---

You can use the choices in the *Sidebands* cyclic field in combination. For example, if you add a sideband using *Array of Coefficients*, the sideband value you added appears in the *Select from Range* list box.

### Maximum clock order

Specifies the largest sideband value and generates the sideband array.



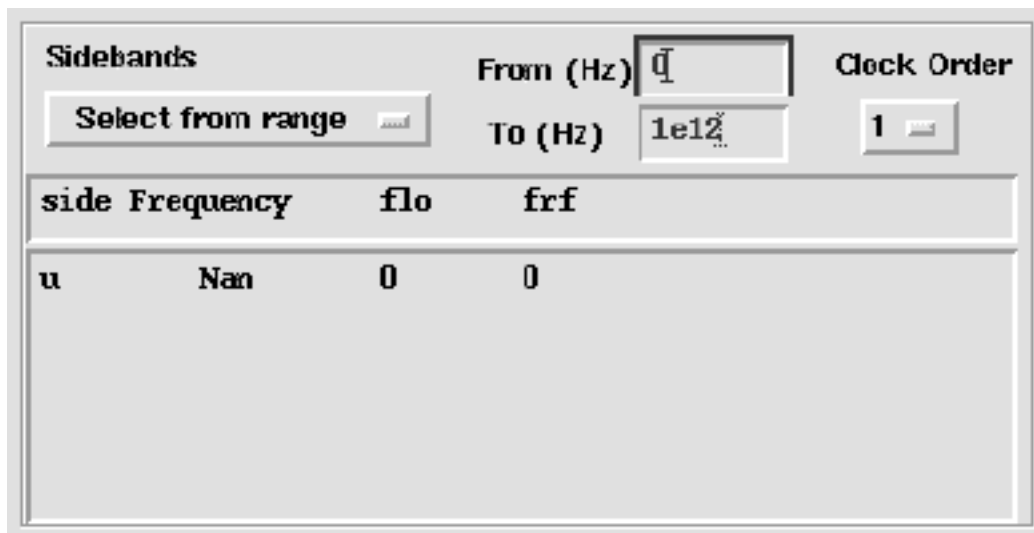
The screenshot shows a panel titled "Sidebands". Inside the panel, there is a dropdown menu labeled "Maximum clock order" with a small arrow on its right side. To the right of this dropdown is a text input field containing the number "1".

Prompts you for a maximum sideband value and automatically generates a sideband array of the form:

```
[-max. sideband . . . 0 . . . + max. sideband]
```

### Select from range

Lets you first a frequency range and select sidebands from within this range of sideband values.



The screenshot shows a panel titled "Sidebands". It contains several controls: a dropdown menu labeled "Select from range" with a small arrow on its right side; two text input fields labeled "From (Hz)" and "To (Hz)" containing the values "0" and "1e12" respectively; and a dropdown menu labeled "Clock Order" containing the value "1". Below these controls is a table with the following structure:

side	Frequency	flo	frf
u	Nan	0	0

Depending on the selection in the *Sweep type* cyclic field, both the values in the list box and the values you specify are either *absolute* values or they are *relative* to the fundamental.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

To specify the listed sidebands,

1. In the *From (Hz)* and *To (Hz)* type-in fields, type the lower and upper values for the frequency range.

The sideband frequencies displayed in the list box are within this range of frequencies.

2. From the *Max. Order* cyclic field, select the maximum order of harmonics that contribute to the sidebands.

If, for example, you select 3 as the *Max. Order* value, the sum of the *absolute* values of the tone coefficients contributing to the sidebands in the list box must be less than or equal to three.

In the list box, the first column is the index of a sideband, the second and third columns list the frequency range of the sideband. The remaining columns list tone coefficients for each fundamental tone that contributed to the listed sideband.

To select from the listed sidebands,

- Click a sideband in the list box to select it.

Select adjacent sidebands by clicking and dragging with the mouse over the sidebands to select. Select sidebands that are not adjacent by holding the `Control` key down while you click the individual sidebands. (Deselect a sideband by clicking a selected sideband while you hold the `Control` key down.)

## Array of Coefficients

Lets you specify sidebands by typing their tone coefficients.

side Frequency	flo	frf
----------------	-----	-----

Tone Coefficients

Clear/Add Delete

To specify sidebands using tone coefficients,

1. Type the tone coefficients, separated by spaces, into the *Tone Coefficients* type-in field.
2. Click *Clear/Add* to add the tone coefficients to the list box.

Values that appear in the list box are *absolute* values or are *relative* to the fundamental, depending on the selection in the Sweeptype cyclic field.

To delete a sideband,

1. Select it in the list box and click *Delete*.

## Specialized Analyses (PAC)

Measures AM and PM small-signal effects for the PAC analysis.

### Modulated

When you select *Modulated* in the *Specialized Analyses* cyclic field, the form changes to let you enter more information.

The *Specialized Analyses* fields for the *Modulated* PAC analysis include the following.

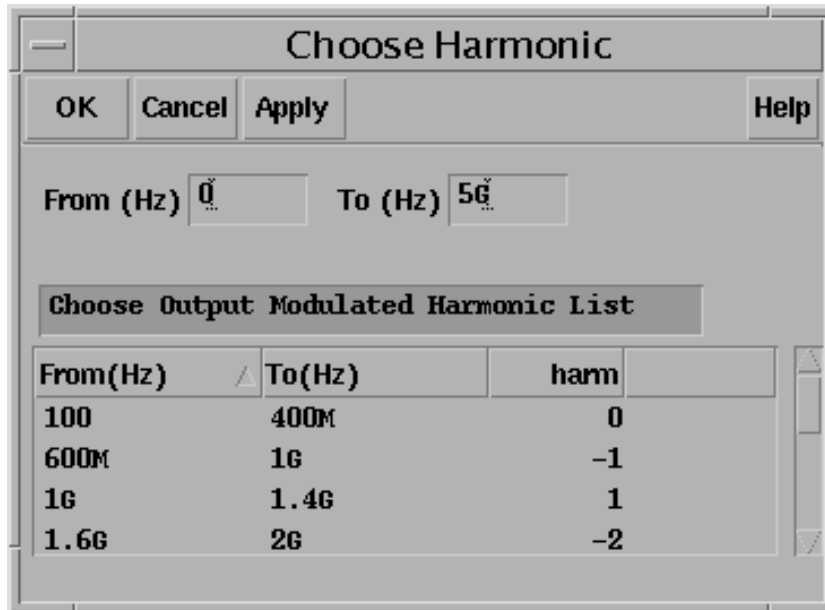
The screenshot shows a dialog box titled "Specialized Analyses". It contains several controls: a "Modulated" button, an "Input Type" dropdown menu set to "SSB", an "Output Modulated Harmonic List" text field containing "1" and a "Choose" button, an "Output Upper Sideband" text field containing "U" and a "Choose" button, an "Enabled" checkbox, and an "Options..." button.

#### Modulated Analysis for PAC Terms and Data Entry Fields

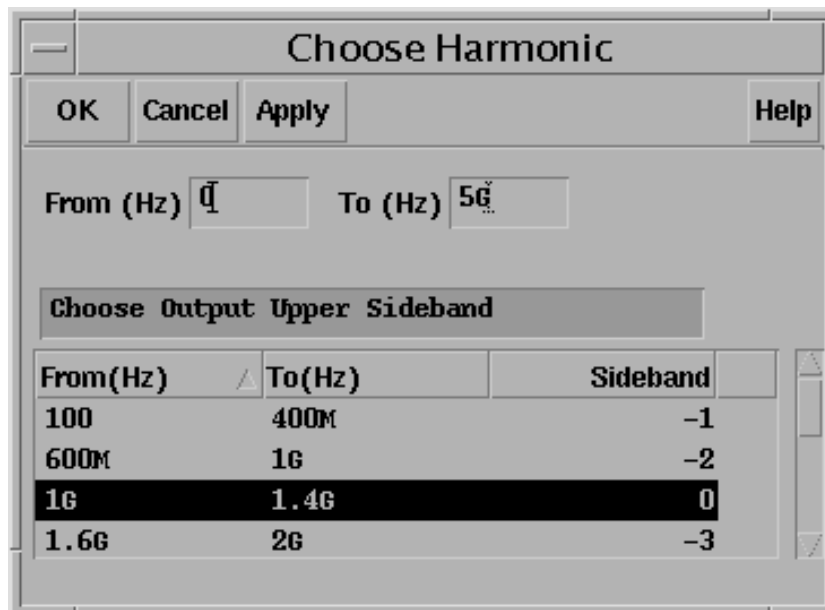
- **Input Type** – This cyclic field displays one of two values: *SSB* and *SSB/AM/PM*.
- **Output Modulated Harmonic List** – Lists the harmonic indexes.
- **Output Upper Sideband** – (Displays when you choose the *Input Type SSB*.) Click *Choose* to display the Choose Harmonic pop up.
- **Input Modulated Harmonic** – (Displays when you choose the *Input Type SSB/AM/PM*.) Click *Choose* to display the Choose Harmonic pop up.

### Choose Harmonic Pop Up

Selects harmonics or sidebands for the analysis. Display the Choose Harmonics pop up from *Modulated Analysis* with the *Choose* button.



OR



### Choose Input/Output Modulated Harmonic List Box and Data Entry Fields

The *Choose Input (or Output) Modulated Harmonic List* box displays the harmonic indexes and associated frequency ranges to select from. Changing the values in the *From (Hz)* and *To (Hz)* fields changes the harmonic indexes and frequency ranges displayed in the list box.

**From (Hz) and To (Hz) fields** - The upper and lower bounds for the frequency range.

**Harm** – Displays the harmonic index, the integer which is multiplied by the fundamental to calculate the harmonic frequency. For example, you set up a PSS analysis of port1 named RF with an harmonic index of 1. Given a PSS fundamental of 900 MHz, port1 is analyzed from 901 MHz to 1000 MHz. For QPSP analysis, the computation is more complicated because there are more fundamentals and the harmonic specification is a vector of indexes.

### Compression Distortion Summary Specialized PAC Analysis

When you select *Compression Distortion Summary* in the *Specialized Analyses* cyclic field, the form changes to let you enter more information.

The *Specialized Analyses* fields for the *Compression Distortion Summary* PAC analysis include the following.

The screenshot shows a software interface titled "Specialized Analyses". At the top, there is a dropdown menu with "Compression Distortion Summary" selected. Below this is a "Contributor Instances" section with a "Select" button and a text field containing "/q58b /q58a /q57 /q56". The "Frequency of Linear Output Signal" is set to "80M". The "Maximum Non-linear Harmonics" is set to "4". Under the "Output" section, there are two radio buttons: "Voltage" (selected) and "Current". To the right, there are two "Select" buttons. The "Out+" button is next to a text field containing "/Pif", and the "Out-" button is next to an empty text field.



### **Compression Distortion Summary Analysis for PAC Terms and Data Entry Fields**

- **Contributor Instances** – Selects and displays an array of device names for the distortion summary. When the field is empty, calculates distortion from each non-linear device.
- **Frequency of Linear Output Signal** – Frequency of the linear output signal. Default is 0.
- **Maximum Non-linear Harmonics** – Frequency of the IM output signal. Default is 0.
- **Output – Voltage** Displays output signals.
- **Out+** Displays the positive output signal.
- **Out-** Default is ground.
- **Output – Current** Displays the output terminal.
- **Term** - Selects and displays the name of the output terminal.

### **Rapid IP3 Specialized PAC Analysis**

When you select *Rapid IP3* in the *Specialized Analyses* cyclic field, the PAC form changes to let you enter more information.

The *Specialized Analysis* fields for the *Rapid IP3* PSS and PAC analysis include the following.

The screenshot shows a dialog box titled "Specialized Analyses" with a dropdown menu set to "Rapid IP3". Below the title bar, there are several fields and controls:

- Source Type**: Three radio buttons labeled "isource", "vsource", and "port". The "isource" button is selected.
- Input Sources 1**: A text field, a "Select" button, and a "Freq" text field.
- Input Sources 2**: A text field, a "Select" button, and a "Freq" text field.
- Input Magnitude**: A text field.
- Frequency of IM Output Signal**: A text field.
- Frequency of Linear Output Signal**: A text field containing "80M".
- Maximum Non-linear Harmonics**: A text field containing "4".
- Output**: Two radio buttons labeled "Voltage" and "Current". The "Voltage" button is selected.
- Out+**: A text field containing "/Pif" and a "Select" button.
- Out-**: A text field and a "Select" button.

### Rapid IP3 Analysis Terms and Data Entry Fields

- **Source Type** – Specifies whether the source is a current, voltage, or port. Select *isource*, *vsource*, or *port*.
- **Input Sources 1** and **Freq** fields – The RF source magnitude and frequency. Click *Select* and select the RF source in the schematic. In the *Freq* field, type the frequency for the source. The *Freq* value is copied to the *Start* field for the PAC analysis.
- **Input Sources 2** and **Freq** fields – A second RF source magnitude and frequency. Click *Select* and select the second RF source in the schematic. In the *Freq* field, type the frequency for the source. The *Freq* value is copied to the *Stop* field for the PAC analysis.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

The RF1 source and the RF2 source can be the same. The associated frequency values must be different.

- **Input Power (dBm)** – RF source power.
- **Frequency of IM Output Signal** – IM3 frequency at the output.
- **Frequency of Linear Output Signal** – IM1 frequency at the output.
- **Maximum Non-linear Harmonics** – Number of harmonics used for the RF signals. Default is 4.
- **Output – Voltage** Output node 1 where IP3 is measured.
- **Out+** (Displays when you choose *Voltage* for *Output*.) Click *Select* and select a net in the schematic.
- **Out-** (Displays when you choose *Voltage* for *Output*.) Click *Select* and select a second net in the schematic. The default is *gnd!*.
- **Output – Current** Selects a terminal in the schematic.
- **Term -** (Displays when you choose *Current* for *Output*.) For output current in a source. Specify it as `<source_name>:p`.

When the output is current in a port, you must use the ADE *Outputs - To be saved - Select in schematic* menu pick and select the terminal in the schematic where current is to be computed and saved.

### Start ACPR Wizard (ENVLP)

Opens the ACPR Wizard form. The ACPR Wizard helps you through the complex process of measuring ACPR (Adjacent Channel Power Ratio) and PSD (Power Spectral Density).



See "[ACPR Wizard](#)" on page 150 for related information.

You can also open the ACPR Wizard from the Simulation window by choosing *Tools – RF – Wizards – ACPR*.

## Stop Time (ENVLP)

Specifies the end point in an ENVLP analysis.



The screenshot shows a rectangular input field with the text "Stop Time" to its left. The field is currently empty.

- Type a time value in the *Stop Time* field.

Make the time interval long enough to let slow signals complete at least one cycle. See [“Clock Name and Select Clock Name Button \(ENVLP\)”](#) on page 36 for related information.

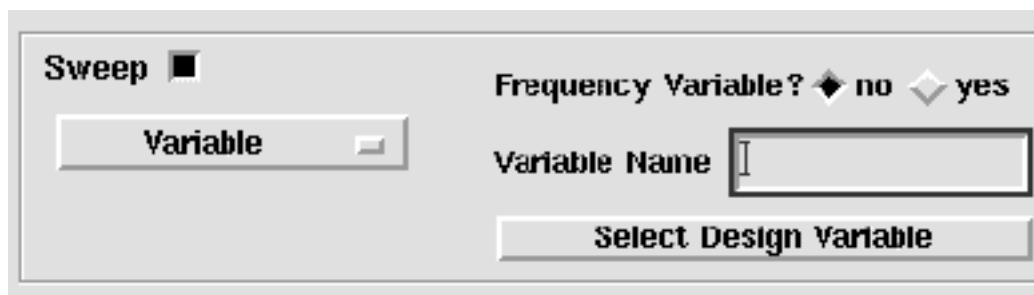
## Sweep (PSS and QPSS)

Specifies how a sweep is performed. Choices are: *Variable*, *Temperature*, *Component Param*, and *Model Param*.

When you activate *Sweep* on the PSS or QPSS analysis form, the *Frequency Sweep Range* on the small-signal analysis forms is restricted to a single point.

### Variable

Sweeps a design variable.



The screenshot shows a control panel for the Sweep function. It includes a "Sweep" checkbox which is checked. Below it is a dropdown menu currently showing "Variable". To the right, there is a "Frequency Variable?" section with radio buttons for "no" and "yes", where "yes" is selected. Below this is a "Variable Name" text input field. At the bottom right is a "Select Design Variable" button.

To sweep a design variable,

1. Select *Variable* from the cyclic field.

The form changes to accept data for the variable sweep.

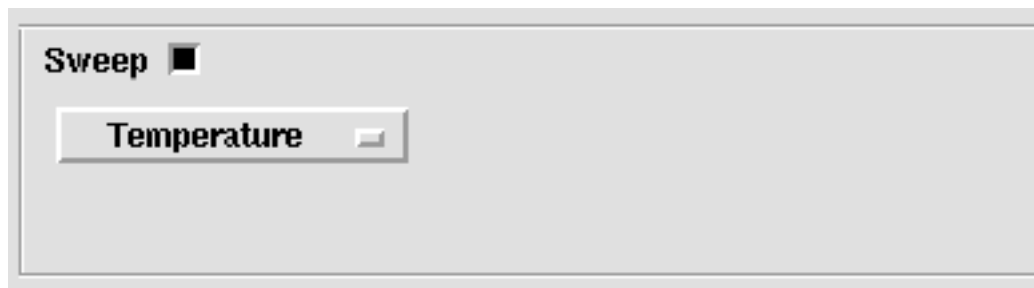
2. Click *Select Design Variable* to display the Select Design Variable form.



3. In the Select Design Variable form, select a variable and click *OK*.  
The variable name appears in the *Variable Name* field. (You can also simply type the name in the *Variable Name* field.)
4. If the selected variable is a frequency variable, highlight *yes* for *Frequency Variable*.

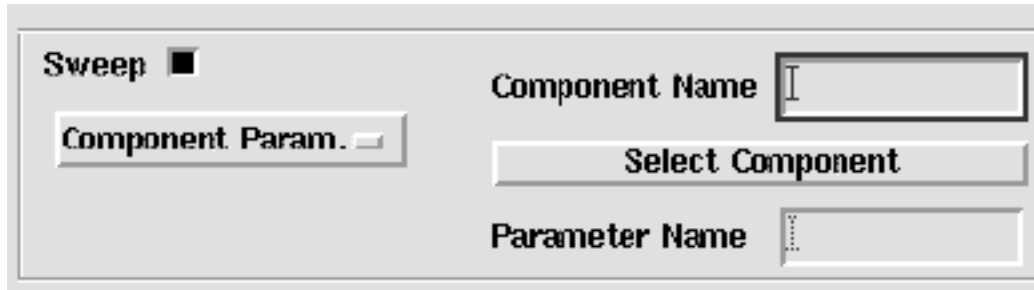
## Temperature

Collects temperature data during the sweep.



## Component Param

Sweeps a component parameter.



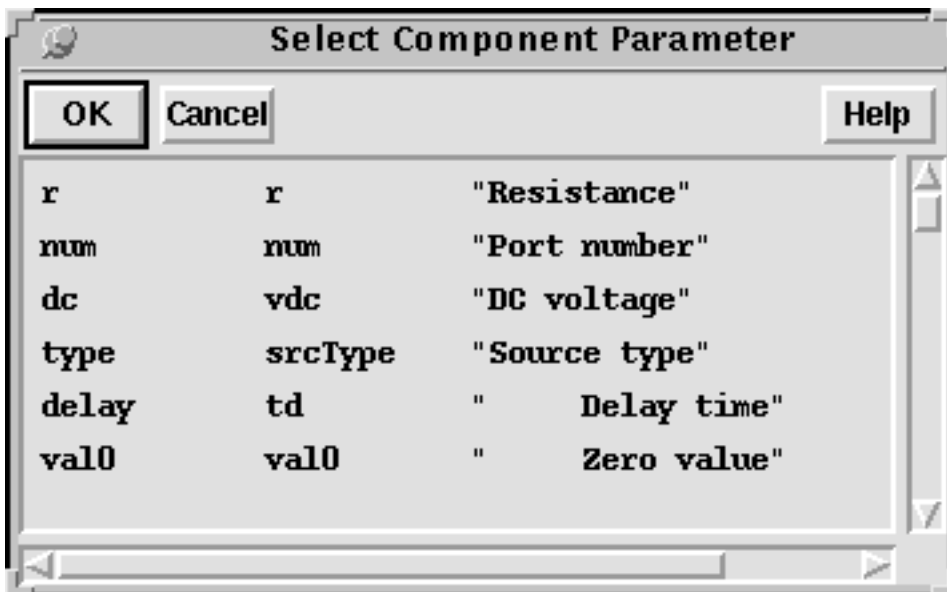
The screenshot shows a dialog box titled "Component Param". It contains a "Sweep" checkbox which is checked. Below it is a dropdown menu showing "Component Param.". To the right is a text field labeled "Component Name". Below the dropdown is a button labeled "Select Component". At the bottom right is a text field labeled "Parameter Name".

1. Select *Component Param.*

The form changes to accept data for the component parameter sweep.

2. Click *Select Component.*
3. Click the component in the schematic.

The Select Component Parameter form appears.



The screenshot shows a dialog box titled "Select Component Parameter". It has "OK", "Cancel", and "Help" buttons. The main area contains a table of parameters:

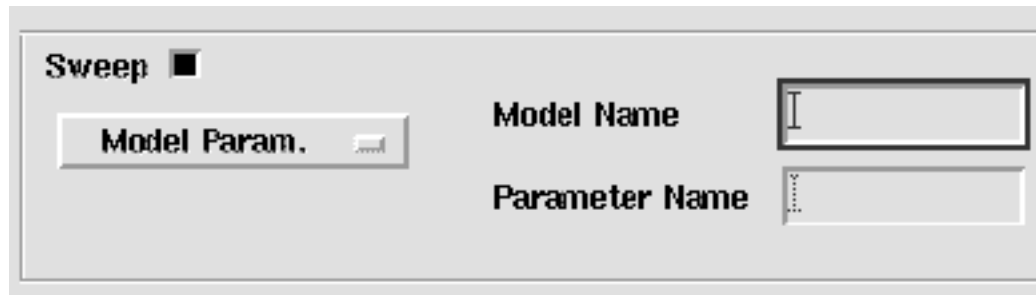
r	r	"Resistance"
num	num	"Port number"
dc	vdc	"DC voltage"
type	srcType	"Source type"
delay	td	" Delay time"
val0	val0	" Zero value"

4. Select a parameter in this form and then click *OK.*

Both the component and parameter names appear in their respective fields. (You can also simply type the component and parameter names in their respective fields.)

## Model Param

Sweeps a model parameter.



The screenshot shows a dialog box titled "Sweep" with a checked checkbox. Below the checkbox is a button labeled "Model Param.". To the right of the button are two input fields: "Model Name" and "Parameter Name". The "Model Name" field is currently empty, and the "Parameter Name" field contains a vertical ellipsis character.

1. Select *Model Param*.

The form changes to accept data for the model parameter sweep.

2. Type the model name and the parameter name in their respective type-in fields.

## Sweep Range, Sweep Type, and Add Specific Points (PSS and QPSS)

Defines the analysis sweep range, the type of sweep, and any additional individual sweep points for the large-signal PSS analysis or the medium-signal QPSS analysis that precedes a small-signal analysis.

When you highlight *Sweep*, the *Sweep Range*, *Sweep Type*, and *Add Specific Points* fields open below the *Sweep* fields.

### Sweep Range

Specifies the bounds for the sweep either as beginning and ending points or as a center point and a span.

### ***Start – Stop***

Defines the beginning and ending points for the sweep.



The screenshot shows a dialog box titled "Sweep Range". On the left, there are two radio button options: "Start-Stop" (which is selected, indicated by a filled diamond) and "Center-Span" (indicated by an empty diamond). To the right of these options are two text input fields. The first field is labeled "Start" and the second is labeled "Stop". Both fields are currently empty.

1. Highlight *Start-Stop*.  
The form changes to let you type the start and stop points.
  2. Type the initial point for the sweep in the *Start* field.
  3. Type the final point in the *Stop* field.
- The *Start* and *Stop* sweep values can be frequencies, periods, or design variable values that correspond to your selection in the *Sweep* cyclic field.

### ***Center – Span***

Defines the center point for the sweep and its span.



The screenshot shows the same "Sweep Range" dialog box. In this view, the "Center-Span" radio button is selected (filled diamond), and the "Start-Stop" radio button is unselected (empty diamond). The text input fields are now labeled "Center" and "Span". Both fields are currently empty.

1. Highlight *Center-Span*.  
The form changes to let you type the center point and span.
2. Type the midpoint for the sweep in the *Center* field.
3. Type the span in the *Span* field.



## Sweep Type

Specifies whether the sweep is linear or logarithmic.

### *Linear*

Specifies a linear sweep.

1.



The screenshot shows a dialog box titled "Sweep Type". It contains four radio buttons: "Linear" (selected), "Logarithmic", "Step Size", and "Number of Steps". To the right of these buttons is a text input field with a vertical cursor.

2. Highlight *Linear*.

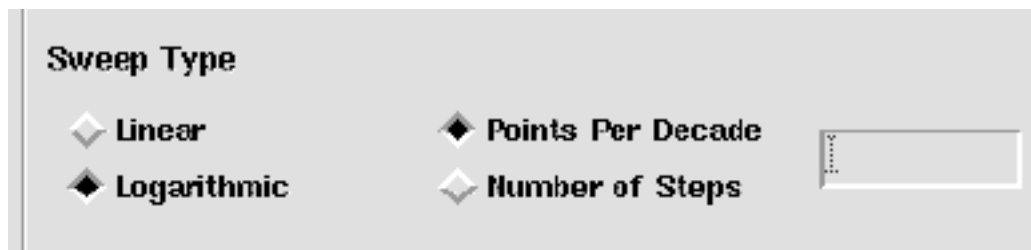
The form changes to let you type either the step size or the total number of points (steps).

3. Either

- Highlight *Step Size* and type the size of each step in the field.
- Highlight *Number of Steps* and type the number of steps in the field.

### *Logarithmic*

Specifies a logarithmic sweep.



The screenshot shows a dialog box titled "Sweep Type". It contains four radio buttons: "Linear", "Logarithmic" (selected), "Points Per Decade", and "Number of Steps". To the right of these buttons is a text input field with a vertical cursor.

1. Highlight *Logarithmic*.

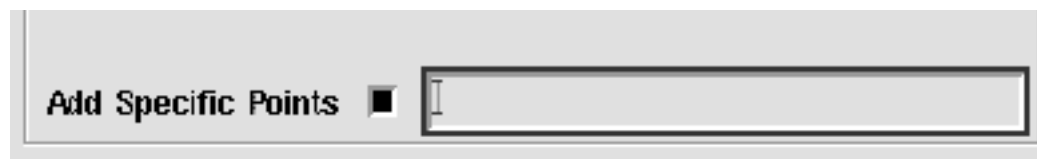
The form changes to let you type either the number of points per decade or the total number of points (steps).

2. Either

- Highlight *Points per Decade* and type the number of points per decade in the field.
- Highlight *Number of Steps* and type the number of steps in the field.

### Add Specific Points

Specifies additional sweep points for the analysis.



1. Highlight *Add Specific Points*.
2. Type the additional sweep point values into the field.  
Separate the sweep points with spaces.

### Sweep Type (PSTB)

Specifies whether the sweep is linear, logarithmic, or chosen automatically.

#### Linear

Specifies a linear sweep.



1. Select *Linear*.  
The form changes to let you specify the step size or the total number of points (steps).
2. Either
  - Highlight *Step Size* and type the size of each step in the field.
  - Highlight *Number of Steps* and type the number of steps in the field.

## Logarithmic

Specifies a logarithmic sweep.



Sweep Type

Logarithmic ▾

◆ Points Per Decade

◇ Number of Steps

1. Select *Logarithmic*.

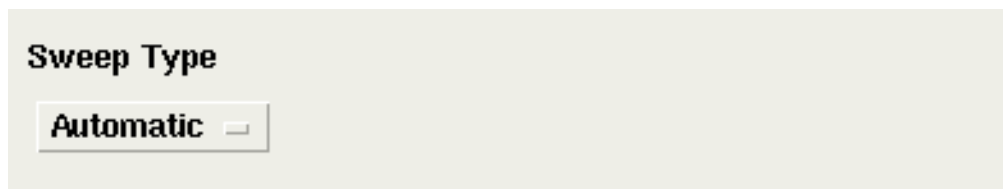
The form changes to let you specify the number of points per decade or the total number of points (steps).

2. Either

- Highlight *Points per Decade* and type the number of points per decade in the field.
- Highlight *Number of Steps* and type the number of steps in the field.

## Automatic

Automatically chooses either the linear or logarithmic sweep types. The sweep is linear when the ratio of stop to start values is less than 10 and the sweep is logarithmic when the ratio is 10 or greater.



Sweep Type

Automatic ▾

- Select *Automatic*.

## Sweeptype (Pnoise)

Controls the inclusion of the *sweeptype* parameter in the Spectre netlist. Choices are: *absolute*, *relative*, and *blank*, with *blank* selecting the appropriate Spectre RF default.

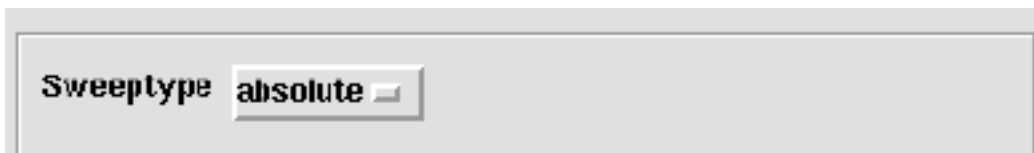
The results vary depending on whether you are simulating an autonomous circuit (an oscillator) or a driven circuit (a mixer) as determined by the *Oscillator* button selection on the PSS Choosing Analyses form.

In general,

- When you simulate an autonomous circuit (the *Oscillator* section of the PSS Choosing Analyses form is active), you can select *relative* for *Sweeptype*. If you leave *Sweeptype* blank, it defaults to *relative*.
- When you simulate a driven circuit (the *Oscillator* section of the PSS Choosing Analyses form is not active), you can either select either *relative* or *absolute* for *Sweeptype*. If you leave *Sweeptype* blank, it defaults to *absolute*.

### Absolute

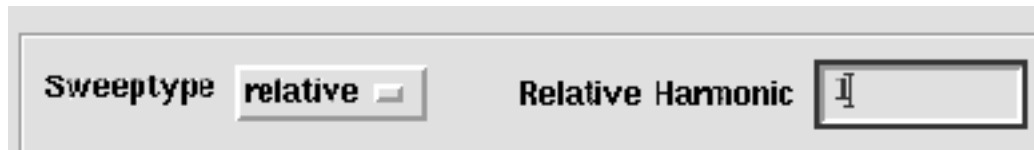
Puts the *Sweeptype* as *absolute* in the Spectre netlist.



In the output for the `Pnoise` sweep, the x axis corresponds to the *Start* and *Stop* values. There is no indication on the plot that you selected *Sweeptype* as *absolute*.

### Relative

Puts the *Sweeptype* as *relative* in the Spectre netlist.




When you select *Sweeptype* as *relative*, you must also enter a value for *Relative Harmonic*. If you enter a 1, it appears in the netlist as `relharmnum=1`.

In the output for a `Pnoise` analysis, the x axis corresponds to the *Start* and *Stop* values in the `Pnoise` sweep. The plot label indicates the selected relative harmonic (for example, `relharmnum=1`). In prior versions of the software, there is no indication.

## Default

Does not put a *Sweeptype* parameter in the Spectre netlist. Sets the appropriate *Sweeptype* value depending on the circuit type.



Sweeptype **default**  Sweep is Currently Absolute

Sets the appropriate *Sweeptype* value according to whether the circuit is autonomous or driven.

- For autonomous circuits, Spectre RF automatically sets *Sweeptype* to *relative* and the *Relative Harmonic* to 1.
- For driven circuits, Spectre RF automatically sets *Sweeptype* to *absolute* and displays the message

`Sweep is Currently Absolute`

In the output, the x axis corresponds to the *Start* and *Stop* values in the `Pnoise` sweep.

- For autonomous circuits, the plot label indicates the default *Relative Harmonic* (`relharmnum=1`). In prior versions of the software, there is no indication.
- For driven circuits, Spectre RF automatically sets *Sweeptype* to *absolute*. There is no indication in the output that you selected *Sweeptype* as *default*.

## Sweeptype (PAC and PXF)

Controls the inclusion of the *sweeptype* newlink parameter in the Spectre netlist. Choices are: *absolute*, *relative*, and *blank*, with *blank* selecting the appropriate Spectre RF default.

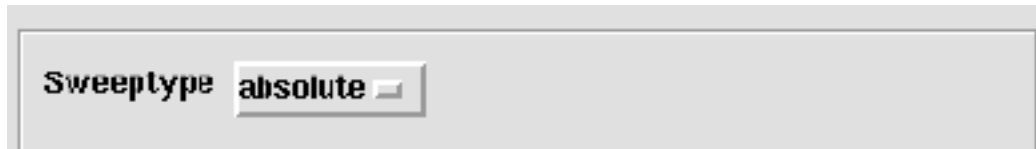
The results vary depending on whether you are simulating an autonomous circuit (an oscillator) or a driven circuit (a mixer) as determined by the *Oscillator* button selection on the PSS Choose Analysis form.

In general,

- When you simulate an autonomous circuit (the *Oscillator* section of the PSS Choosing Analyses form is active), you can select *relative* for *Sweeptype*. If you leave *Sweeptype* blank, it defaults to *relative*.
- When you simulate a driven circuit (the *Oscillator* section of the PSS Choosing Analyses form is not active), you can either select either *relative* or *absolute* for *Sweeptype*. If you leave *Sweeptype* blank, it defaults to *absolute*.

## Absolute

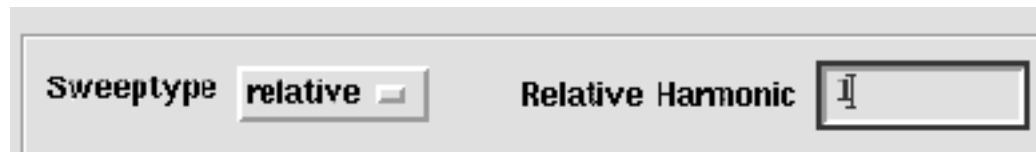
Puts the *Sweeptype* as *absolute* in the Spectre netlist.



There is no indication on the plot that you selected *Sweeptype* as *absolute*.

## Relative

Puts the *Sweeptype* as *relative* in the Spectre netlist.



When you select *Sweeptype* as *relative*, you must also enter a value for *Relative Harmonic*. If you type 1, it appears in the netlist as `relharmnum=1`.

For PXF analysis with *sweeptype* set to *relative* and `freqaxis=in`, the simulation output is shifted frequency with the x axis labeled

```
relative frequency (offset from xx HZ)
```

For PAC analysis with *sweeptype* set to *relative* and `freqaxis=out`, the simulation output is shifted frequency with the x axis labeled

```
relative frequency (offset from xx HZ)
```

For PXF analysis with *sweeptype* set to *relative* and `freqaxis=absin`, the simulation output is absolute frequency. For PAC analysis with *sweeptype* set to *relative* and `freqaxis=absout`, the simulation output is absolute frequency.

At the bottom of the direct plot form for the analysis, the `freqaxis` value displays along with one of the following messages-`relative frequency (offset xxx)` or `relative freq`. The plot label also indicates the selected relative harmonic (for example, `relharmnum=1`). In prior versions of the software, there is no indication.

## Default

Does not put a *Sweeptype* parameter in the Spectre netlist.



Sweeptype **default**  Sweep is Currently Absolute

Sets the appropriate Spectre RF default depending on whether the circuit is autonomous or driven.

For autonomous circuits, Spectre RF automatically sets the *Sweeptype* to *relative* and the *Relative Harmonic* to 1.

For driven circuits, Spectre RF automatically sets the *Sweeptype* to *absolute* and displays the message *Sweep is Currently Absolute*.

In the output, the x axis corresponds to the *Start* and *Stop* values in the *Pnoise* sweep.

- For autonomous circuits, the plot label indicates the default *Relative Harmonic* (*relharmnum=1*). In prior versions of the software, there is no indication.
- For driven circuits, Spectre RF automatically sets the *Sweeptype* to *absolute*. There is no indication you selected *Sweeptype* as *blank*.

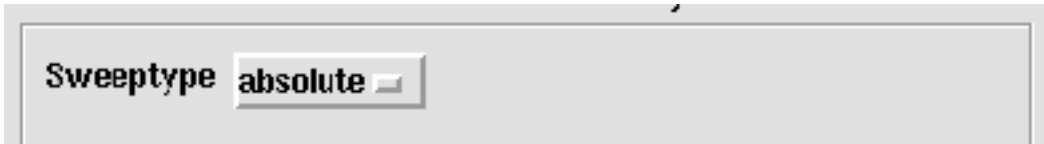
## Sweeptype (PSP and QPSP)

Controls the inclusion of the *sweeptype* parameter in the Spectre netlist. Choices are: *absolute*, *relative*, and *blank*, with *blank* selecting the appropriate Spectre RF default.

Because the computations for PSP analysis involve inputs and outputs at frequencies that are relative to multiple harmonics, *Sweeptype* behaves differently in PSP and QPSP analysis than it does in PAC, *Pnoise*, and *PXF* analyses. With PSP and QPSP analysis, the frequency of the input and the frequency of the response are usually different.

## Absolute

Puts the *Sweeptype* as *absolute* in the Spectre netlist.

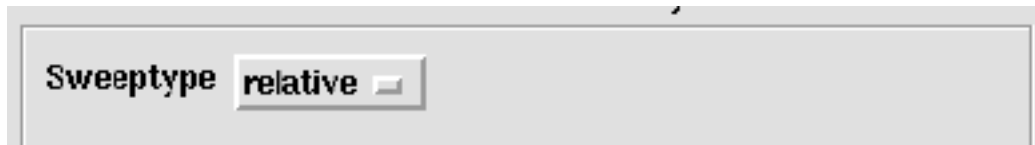


Sweeptype **absolute**

Specifying *Sweeptype* as *absolute*, sweeps the absolute input source frequency. There is no indication on the plot that you selected *Sweeptype* as *absolute*.

### Relative

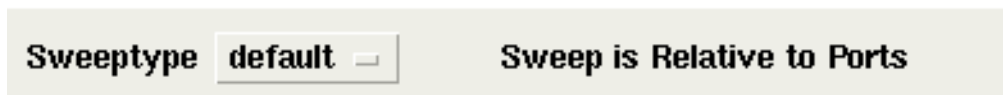
Puts the *Sweeptype* as *relative* in the Spectre netlist.



Specifying *Sweeptype* as *relative*, indicates to sweep relative to the analysis harmonics (rather than the PSS or QPSS fundamental).

### Default

Does not put a *Sweeptype* parameter in the Spectre netlist. Sets the appropriate Spectre RF default depending on the circuit type.



## Options Forms

Each Options form lets you specify parameter values for a Spectre RF analysis. Options that are not relevant for a particular analysis do not appear on its Options form.

- ▶ On the Choosing Analyses form, click the *Options* button to open the Options form corresponding to the analysis type that is currently highlighted on the Choosing Analyses form.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

For example, the following figure shows the top portion of the Options form for the Envelope analysis.

The screenshot shows the 'Envelope Following Options' dialog box. It features a title bar with standard window controls (minimize, maximize, close). Below the title bar are five buttons: 'OK', 'Cancel', 'Defaults', 'Apply', and 'Help'. The main content area is organized into three sections, each with a bolded title and several input fields:

- SIMULATION INTERVAL PARAMETERS**
  - start**: A text input field.
  - outputstart**: A text input field.
  - tstab**: A text input field.
- SIMULATION BANDWIDTH PARAMETERS**
  - modulationbw**: A text input field.
- TIME STEP PARAMETERS**
  - maxstep**: A text input field.
  - envmaxstep**: A text input field.
  - fixstepsize**: A radio button control with 'yes' and 'no' options.
  - stepsize**: A text input field.

Use the Options form for an analysis to define its parameter values. Only those options that are relevant for a particular analysis are available on its Options form.

## Field Descriptions for the Options Forms

The following sections describe all the simulation parameters whose values you specify on Options forms. The sections are arranged alphabetically, according to the top-level headings on the forms. The top-level headings are usually found along the leftmost margin of the form.

### Accuracy Parameters (PSS, QPSS, and ENVLP)



In most cases, the `errpreset` parameter is the only accuracy parameter you should set.

Use the following links to locate detailed descriptions of how the `errpreset` parameter works to set accuracy parameters.

- ❑ [The `errpreset` Parameter in PSS Analysis](#)
- ❑ [The `errpreset` Parameter in QPSS Analysis](#)
- ❑ [The `errpreset` Parameter in Envlp Analysis](#)

**envlteratio** (ENVLP only) the ratio the simulator uses to compute envelope LTE tolerances for Envelope analysis. The default value is based on the accuracy default.

**fdharms** (PSS only) sets the number of harmonics considered for distributed (frequency-domain) components such as `nport`, `delay`, `mtline`, and delayed controlled sources. This parameter is supported only for the shooting engine.

#### **finitediff** (PSS and QPSS)

- For PSS analysis, uses the finite difference (FD) refinement method after the shooting method for driven circuits. The *finitediff* parameter refines the simulation results and is only meaningful when *highorder* is set to `no`. The possible settings are *no*, *yes* or *refine*.
- For QPSS analysis, uses the finite difference (FD) refinement method to refine the simulation results after the quasi-periodic shooting method. The possible settings are *no*, *yes* or *refine*.

**no** turns off the finite difference refinement method.

**yes** applies the finite difference refinement method to the PSS or QPSS analysis. The finite difference method tries to improve the initial small time steps if necessary.

**refine** applies the finite difference refinement method to the PSS or QPSS analysis.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

For the PSS analysis the finite difference method tries to refine the time steps. When the simulation uses the *gear2* method, uniform 2nd order gear is used.

When you use *readpss* and *writepss* to re-use the PSS analysis results for a driven circuit, *finitediff* automatically changes from *no* to *yes*. If you are concerned that the accuracy of your simulation might be affected by a loose *steadyratio*, you might want to try *finitediff*.

You might also set *finitediff* to *yes* to get more uniform time steps and reduce the noise floor. This does not always work. At lower power levels, the finite difference method might sometimes reduce the noise floor. We recommend that you reduce the noise floor by setting *highorder* to *yes*.

#### *Important*

Be aware of the following information for the finite difference method.

- ❑ The finite difference method works for driven circuits only. If you use the finite difference method with an oscillator circuit, it serves only as a loading routine. Even when a circuit or analysis parameter has changed, the old solution is loaded and may be inaccurate.
- ❑ For the QPSS analysis the finite difference method tries to refine the solution after the quasi-periodic shooting method.
- ❑ When you use *readqpss* and *writeqpss* to re-use the QPSS analysis results, *finitediff* automatically changes from *no* to *samegrid*.

**fullpssvec** (PSS only) Uses the full vector containing solutions at all PSS time steps in the linear solver. Default behavior is derived from the size of the equation and the property of the PSS time steps. Possible values are *no* or *yes*.

**highorder** (PSS only) executes high-order refinement after low order convergence when *errpreset* is either *moderate* or *conservative*. Uses the Multi-Interval Chebyshev (MIC) polynomial spectral algorithm. The *highorder* parameter works for both driven and autonomous circuits. The possible settings are *no* or *yes*. *yes* turns on the MIC method and tries harder to converge. *no* turns off the MIC method.

When you set *errpreset* to either *moderate* or *conservative* and you have not set *highorder*, MIC is used but it does not aggressively try to converge. MIC does try harder to converge when *highorder* is explicitly set to *yes*.

**inexactNewton** (PSS and QPSS) determines whether the inexact Newton method is used. The possible settings are *no* or *yes*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

**itres=1e<sup>-4</sup>** (for the shooting engine), **itres=0.9** (for the HB engine) (PSS only) sets the relative tolerance for the linear solver.

**Insolver** (PSS and QPSS) specifies the linear solver to be used.

**bicgstab** specifies that the biconjugate gradient stabilized (bicgstab) variant of the conjugate gradient (cg) solver is to be used. The bicgstab solver is formulated for nonsymmetric linear systems. The bicgstab solver uses less memory than either gmres or qmr but is the least robust of the solvers.

**gmres** specifies that the general minimum residual (gmres) linear solver is to be used. This is the default. If memory issues arise when using this solver, consider using either bicgstab or qmr, which use less memory.

**qmr** specifies that the quasi minimal residual (qmr) linear solver is to be used. The qmr solver uses less memory than gmres but is not as robust.

**resgmres** specifies that the restarted GMRES solver (resgmres) linear solver is to be used. The resgmres solver can reduce memory cost and simulation time.

**Iteratio** is the ratio the simulator uses to compute LTE tolerances from the Newton tolerance. The default value is based on the accuracy default.

**maxacfreq** (not QPSS) is the maximum frequency used in a subsequent periodic small-signal analysis. This parameter automatically adjusts *maxstep* to reduce errors due to aliasing in frequency-domain results. The default is based on *maxstep* and *harms* values. See "[Virtuoso Spectre Circuit Simulator RF Analysis Theory](#)" for more information about specifying this parameter.

**maxorder** is the maximum order of the MIC polynomials used during waveform approximation. Values range from 2 to 16. The default value is 16 for driven circuits and 12 for autonomous circuits.

**maxperiods** is the maximum number of simulated periods allowed for the simulation to reach steady-state. Default is 20.

**psaratio=1** is the ratio used to compute the MIC accuracy from the Newton tolerance.

**relref** is the reference used for relative convergence criteria. Your **Accuracy Defaults** choice sets the default values.

**allglobal** is the same as sigglobal except that it also compares the residues for each node to the historical maximum.

**alllocal** compares the relative error at each node to the largest values ever found for that node.

**pointlocal** compares the relative errors in quantities at each node to that node alone.

**sigglobal** compares relative errors in each signal to the maximum value for all signals.

**resgmrescycle** specifies the length of the computation cycle and determines whether the recycling feature of the resgmres solver is used.

- The *instant*, *short*, and *long* values are most appropriate when memory is the primary concern.
- The *recycleinstant*, *recycleshort*, and *recyclelong* values are most appropriate when simulation time is the major concern. Circuits where neighboring frequency points are close benefit most from using the three recycle values. The recycle values are not supported for large signal analyses.
- For large signal analyses, Cadence recommends using the resgmres solver only for large signal single tone analyses such as PSS. For the best performance in large signal analyses, try the gmres solver first and then, if necessary, try the resgmres solver.

Possible values are

**instant** uses the shortest computation cycle, which means the least amount of memory is used. Use this value if the circuit is almost linear or is only weakly non-linear.

**short** uses a mid-length computation cycle, which requires more memory than the shortest computation cycle. This is the default value.

**long** uses the longest computation cycle, which requires the most memory. Use this value if the circuit is strongly non-linear.

**recycleinstant** turns on the recycling features of the solver, uses the shortest computation cycle, and uses the least amount of memory. Use this value if the circuit is almost linear or is only weakly non-linear. This value is not supported for large signal analyses.

**recycleshort** turns on the recycling features of the solver, uses a mid-length computation cycle, and uses more memory than the shortest computation cycle. This value is not supported for large signal analyses.

**recyclelong** turns on the recycling features of the solver, uses the longest computation cycle, and requires the most memory. Use this value if the circuit is strongly non-linear. This value is not supported for large signal analyses.

**steadyratio** is the ratio the simulator uses to compute steady-state tolerances from the LTE tolerance. This parameter adjusts the maximum allowed mismatch in node voltages and current branches during the steady-state period. The default is based on the accuracy default.

## Additional Parameters (All)

**additionalParams** provides a place where you can enter parameters that are not supported in the graphical user interface.

## Annotation Parameters (All)

**annotate** lets you specify what information is printed at the beginning of the output to identify the results. Default is `status`. Choices are `no`, `title`, `sweep`, `status` and `steps`.

**stats** tells the simulator to generate analysis statistics. Default is `no`.

## Convergence Parameters (All)

**cmin** is the minimum capacitance from each node to ground. Default is 0.

**gear\_order** is the order used for Gear-type interpolations. Default is 2 (second order).

**Insolver** (PAC) specifies the linear solver to be used.

**bicgstab** specifies that the biconjugate gradient stabilized (bicgstab) variant of the conjugate gradient (cg) solver is to be used. The bicgstab solver is formulated for nonsymmetric linear systems. The bicgstab solver uses less memory than either gmres or qmr but is the least robust of the solvers.

**gmres** specifies that the general minimum residual (gmres) linear solver is to be used. This is the default. If memory issues arise when using this solver, consider using either bicgstab or qmr, which use less memory.

**qmr** specifies that the quasi minimal residual (qmr) linear solver is to be used. The qmr solver uses less memory than gmres but is not as robust.

**resgmres** specifies that the restarted GMRES solver (resgmres) linear solver is to be used. The resgmres solver can reduce memory cost and simulation time.

**oscsolver** lets you specify the type of solver to use for an oscillator circuit. Possible values are `std`, `turbo` or `ira`. The default is `turbo`.

**ira** uses the implicitly restarted Arnoldi algorithm to calculate the dominant eigenvalue and the corresponding eigenvector for small-signal analysis of oscillator circuits. `Ira` uses less memory than `turbo`.

**std** uses a full eigen analysis to calculate the dominant eigenvalue and the corresponding eigenvector. Using the `std` solver for small-signal analysis of oscillator

circuits is slower but more robust than simulation using the `turbo` setting. Use the `std` setting if simulation with the `turbo` setting is unsuccessful.

**turbo** uses the Arnoldi algorithm based on the Krylov subspace to calculate the dominant eigenvalue and the corresponding eigenvector for oscillator small-signal analysis. `Turbo` is significantly faster than `std` and uses significantly less memory. In rare situations `turbo` might be *less* accurate.

**readns** lets you specify the name of a file that contains an estimate of the initial transient solution. Enter the complete path to the file. No default.

**solver** lets you specify the type of solver to use for a linear system. Possible values are `std` or `turbo`. The default is `turbo`.

**std** for each frequency value, solves the linear system using the full GMRES algorithm. Simulations using the `std` solver are slower but more robust than simulations using the `turbo` setting. Use the `std` setting if simulation with the `turbo` setting is unsuccessful.

**turbo** for each frequency value, solves the linear system using the recycled Krylov subspace algorithm. Using `turbo` is significantly faster than using `std`. In rare situations `turbo` might be *less* accurate.

**tolerance** is the relative tolerance for the linear solver when solving for convergence. Default is  $10^{-9}$ .

## Initial Condition Parameters (PSS, QPSS, and ENVLP)

**ic** specifies the methods used to set the initial condition (`dc`, `node`, `dev` and `all`). For an explanation of each of these methods, consult the *Virtuoso Spectre Reference* manual. Default is `all`.

**readic** lets you specify the name of the file that contains the initial conditions.

### **skipdc**

**no calculates** the initial solution using the normal DC analysis. This is the default.

**yes** omits the DC analysis from the transient analysis and gives the Initial solution either in the file specified by the `readic` parameter or by the values specified on the `ic` statements.

**sigrampup** independent source values start at 0 and ramp up to their initial values in the first phase of the simulation. Enables waveform production in the time-varying independent source after the rampup phase.

- The rampup simulation is from `tstart` to `time=0` s.

- ❑ The main simulation is from time=0 s to *tstab*.
- ❑ If you do not specify the *tstart* parameter, the default *tstart* time is set to -0.1 multiplied by *tstab*.

## Integration Method Parameters (PSS, QPSS, and ENVLP)

**envmethod** (ENVLP) specifies the integration method for the ENVLP analysis. Your accuracy default choice sets the default value. The possible settings are

**euler** is backward Euler.

**gear2** is the backward Euler and second-order Gear methods.

**gear2only** is Gear's second-order backward difference method only.

**trap** is the backward Euler and trapezoidal methods.

**trapgear2** is the backward Euler, trapezoidal and second-order Gear methods.

**traponly** is the trapezoidal rule only.

The trapezoidal rule is best when you want high accuracy, but it can exhibit point-to-point ringing, which you can control with tighter error tolerances. Euler and Gear work better with looser tolerances for quick simulation, but they can make systems appear more stable than they actually are.

**method** (PSS, QPSS and ENVLP) specifies the integration method. Your accuracy default choice sets the default value. The possible settings are

**euler** is backward Euler.

**gear2** is the backward Euler and second-order Gear methods.

**gear2only** is Gear's second-order backward difference method only.

**trap** is the backward Euler and trapezoidal methods.

**traponly** is the trapezoidal rule only.

The trapezoidal rule is best when you want high accuracy, but it can exhibit point-to-point ringing, which you can control with tighter error tolerances. Euler and Gear work better with looser tolerances for quick simulation, but they can make systems appear more stable than they actually are.

Use the *gear\_order* option on the small-signal analyses Options form to set the order of a Gear-type interpolation.



**tstabmethod** (PSS) specifies the tstab integration method. Your accuracy default choice sets the default value. The possible settings are

**euler** is backward Euler.

**gear2** is the backward Euler and second-order Gear methods.

**gear2only** is Gear's second-order backward difference method only.

**trap** is the backward Euler and trapezoidal methods.

**traponly** is the trapezoidal rule only.

The trapezoidal rule is best when you want high accuracy, but it can exhibit point-to-point ringing, which you can control with tighter error tolerances. Euler and Gear work better with looser tolerances for quick simulation, but they can make systems appear more stable than they actually are.

Use the *gear\_order* option on the small-signal analyses Options form to set the order of a Gear-type interpolation.

## Multitone Stabilization Parameter (QPSS)

**stabcycles** specifies the number of stabilization cycles to perform when both large and moderate sources are enabled. The default is 2.

## Newton Parameters (PSS, QPSS, and ENVLP)

**envmaxiters** specifies the maximum number of Newton iterations per envelope step for the ENVLP analysis. The default is 3.

**maxiters** is the maximum number of iterations per time step.

**restart** tells the simulator not to use the previous DC solution as an initial guess. Default is yes (means do not use).

## Output Parameters (All)

**compression** directs the simulator to perform data compression on the output. Default is *no*.

**freqaxis** specifies what version of the frequency to plot the output against in spectral plots.

- For the PAC and QPAC analysis

*absout* is the absolute value of the output frequency.  
*in* is the input frequency.  
*out* is the output frequency.

- For the PXF and QPXF analysis

*absin* is the absolute value of the input frequency.  
*in* is the input frequency.  
*out* is the output frequency.

- For the PSP and QPSP analysis

*absin* is the absolute value of the frequency swept at the input.  
*in* is the scattered frequency at the input.  
*out* is the scattered frequency at the output.

**oppoint** specifies whether the simulator outputs the operating point information. You can send the information to a rawfile, the logfile, or the screen. Default is *NO*.

**outputperiod** lets you specify the time-domain output period. The time-domain small-signal response is computed for the period specified, rounded to the nearest integer multiple of the PSS analysis period.

**outputtype** lets you specify the output type for ENVLP Analysis. Possible values are *both*, *envelope*, or *spectrum*. The default is *both*.

**save** tells the simulator what signals to save. You have the following choices:

**all** saves all signals.

**allpub** saves only signals that are normally useful. Normally useful signals include shared node voltages and currents through voltage sources and iprobes.

**lvl** saves all signals up to *nestlvl* deep in the subcircuit hierarchy. *lvl* is relevant for subcircuits. Click *lvl* to activate the *nestlvl* type-in field. Then enter a value for the *nestlvl* parameter.

**lvlpub** saves all normally useful signals up to *nestlvl* deep in the subcircuit hierarchy. *lvlpub* is equivalent to *allpub* for subcircuits. Normally useful signals include shared node voltages and currents through voltage sources and iprobes. Click *lvlpub* to activate the *nestlvl* type-in field. Then enter a value for the *nestlvl* parameter.

**selected** saves only the signals you request on the *Outputs* menu in the Simulation window. This is the default setting.

Use *lvl* or *all* (instead of *lvlpub* or *allpub*) to include internal node voltages and currents through other components that compute current.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

Use *lv/pub* or *all/pub* to exclude signals at internal nodes on devices (the internal collector, base, emitter on a BJT, the internal drain and source on a FET, and so on). *lv/pub* and *all/pub* also exclude the currents through inductors, controlled sources, transmission lines, transformers, etc.

**saveallsidebands** lets you save all sidebands for Pnoise and Qnoise analyses.

**skipcount** specifies how many points to skip before saving a point. No default.

**skipstart** specifies when the simulator starts skipping output data. Default is *starttime s* (seconds).

**skipstop** specifies when the simulator stops skipping output data. Default is *stoptime s* (seconds).

**stimuli** specifies what *PXF* and *QPXF* uses for inputs to the transfer functions.

**nodes\_and\_terminals** specifies that all possible transfer functions are computed.

**sources** specifies that the sources present in the circuit are used as the inputs to the transfer functions.

**strobedelay** is the delay (phase shift) between the skipstart time and the first strobe point. Default is 0.

**strobeperiod** is the output strobe interval in seconds. The actual strobe interval is rounded to an integer multiple of the clock period.

Strobing is crucial to getting the very low noise floors required for ACPR measurements. Some users require noise floors 70 to 80 dB below the peak, which is not possible if the Fourier analysis has to interpolate between unevenly spaced data points. The *strobeperiod* option forces the output envelope to have evenly spaced data points. The subsequent Fourier analysis can proceed without interpolation.

**use ppv for osc** lets you save a perturbation projection vector (PPV) file. The PPV is the periodic eigenfunction corresponding to the unity eigenvalue of the linearized autonomous circuit's adjoint equations. A PPV can be thought of as representing the oscillator's phase sensitivity to perturbations in the voltage or current at the nodes of the oscillator. The saved PPV file is used in phase noise calculations, such as when you prepare to generate a VCO macromodel.

The PPV file is saved into the raw directory after simulation. For example, after running an autonomous circuit named *vco.ckt* with the *Save osc PPV* option set to *yes*, the PPV file is placed in the *vco.raw* directory. The PPV file is named *analysisID.td.ppv.pss*.

If the *Save osc PPV* option is set to *yes* and the PNOISE analysis *use ppv for osc* option is set to *yes* also, only the PPV file for the PNOISE analysis is saved.

## **Simulation Bandwidth Parameters (ENVLP)**

**modulationbw** specifies the modulation bandwidth.

## **Simulation Interval Parameters (ENVLP, PSS and QPSS)**

**outputstart** (ENVLP) specifies the timepoint when the simulator starts to save output.

**start** (ENVLP) specifies the analysis start time.

**tstab** (ENVLP) specifies the initial stabilization time. Default is 0.

**tstart** (PSS and QPSS) is the start time you specify for transient analysis. It can be negative or positive. Default is 0.

## **State File Parameters (ENVLP, PSS and QPSS)**

**checkpss** Options are *yes* and *no*.

**readpss** specifies the file from which the steady-state solution is read. Small-signal analyses can read the steady-state solution from this file so rerunning the PSS analysis is unnecessary.

**readqpss** specifies the file from which the steady-state solution is read. Small-signal analyses can read the steady-state solution from this file so rerunning the PSS analysis is unnecessary.

**recover** (PSS) specifies the *tstab* analysis states file to be restored. The *saveperiod*, *savetime*, *savefile*, and *recover* parameters are used to save and restart the *tstab* part of the PSS, QPSS, and ENVLP analyses. This feature provides functionality similar to that of the *save* and *restart* feature of transient analysis.

**saveclock** (PSS) saves the *tran* analysis periodically at specified wall clock times.

**savefile** (PSS) saves the *tstab* part of *tran* analysis states into the specified file.

**saveperiod** (PSS) saves the *tstab* part of *tran* analysis states periodically at specified simulation times.

**savetime** (PSS) saves the tstab part of tran analysis states into files at the specified time points. The savetime parameter takes precedence over the saveperiod parameter when both are specified.

**swapfile** is a temporary file that holds steady-state information. If you enter a filename, the simulator stores the operating point in that file rather than in virtual memory. Use this option if you receive a warning about not having enough memory to complete the analysis. Enter the complete path to the file.

**write** lets you specify the name of the file to which the Spectre RF simulation writes the initial transient solution.

**writefinal** lets you specify the name of the file to which the Spectre RF simulation writes the final transient solution.

**writepss** specifies the file to which the steady-state solution is written. Small-signal analyses can read the steady-state solution from this file so rerunning the PSS analysis is unnecessary.

**writeqpss** specifies the file to which the steady-state solution is written. Small-signal analyses can read the steady-state solution from this file so rerunning the PSS analysis is unnecessary.

## **Time Step Parameters (ENVLP, PSS, QPSS)**

**envmaxstep** specifies the maximum outer envelope size. The default is set by the accuracy default.

**fixstepsize** (ENVLP) fixes the envelope step size, which can potentially speed up an ENVLP analysis.

**liberal** = 0.1/max AC frequency

**moderate** = 2 x liberal

**conservative** = 4 x liberal

**maxacfreq** (QPSS shooting engine only) specifies the maximum frequency requested in a subsequent periodic small-signal analysis. The default is derived from *Accuracy Defaults (errpreset)* and *Harms*.

**maxstep** is the largest allowable time step. The default is set by the *Accuracy Defaults (errpreset)*.

**step** is the smallest simulator time step used to improve the look of the results. Default is 0.001 x fundamental period seconds.

**stepsize** (ENVLP) specifies the number of cycles skipped for each step when *fixstepsize* is set to *yes*.

## Direct Plot Form

Use the *Direct Plot* command in the Simulation window to plot most RF simulation results.

### Opening the Direct Plot Form

Use the *Results – Direct Plot – Main Form* command in the Simulation window, to access results for the most recently performed analyses. The Direct Plot form is similar to the one shown in [Figure 2-2](#) on page 119.

Figure 2-2 Direct Plot Form

OK Cancel Help

Plot Mode  Append  Replace

Analysis

pss

Function

Voltage  Current  
 Power  Voltage Gain  
 Current Gain  Power Gain  
 Transconductance  Transimpedance  
 Compression Point  IPN Curves  
 Power Contours  Reflection Contours  
 Harmonic Frequency  Power Added Eff.  
 Power Gain Vs Pout  Comp. Vs Pout  
 Node Complex Imp.

Harmonic Frequency

0	0
1	40K
2	80K
3	120K
4	160K
5	200K

Add To Outputs

> Select Harmonic Frequency on this form...

The *Analysis* section lists one or more analyses with available data.

For periodic large and small-signal analyses, choose *pss* to access the results from the initial PSS analysis. Choose *pac*, *pnoise*, *pxf*, or *psp* to access the results of any available periodic small-signal analyses that ran after the PSS analysis.

For quasi-periodic large and small-signal analyses, choose *qpss* to access the results from the initial QPSS analysis. Choose *qpac*, *qpnoise*, *qpxf*, or *qpsp* to access the results of any available quasi-periodic small-signal analyses that ran after the QPSS analysis.

Choose *Envlp* to access the results from an envelope analysis.

## Defining Measurements in a Plot Form

You define the RF simulation measurements to display by entering and selecting items in the Direct Plot form such as functions, plots, and modifiers and selecting nets, terminals, or other objects on the schematic.

While making selections in the Direct Plot form, follow the messages at the bottom of the form for instructions and prompts. For example,

> *Select Net on schematic...*

and

> *To plot, press Sij-button on this form...*

When you click the plot button (or perform another specified action), a simulation plot appears, by default, in the current Waveform Window. If the Waveform window or Schematic window is not open, selecting a direct plot function automatically opens both windows.

Informative messages often also display at the bottom of the Waveform and Schematic windows.

## Plotting Data for Swept Simulations

For swept analyses, the last section in the Direct Plot form includes one or more list boxes that display different values of the design variable you selected when setting up the swept analysis. [Figure 2-3](#) on page 121, the Direct Plot form for an IP# measurement, has two such list boxes. Following the messages at the bottom of the Direct Plot form, in the list box, click the design variable values you want to plot. The name and type of the variable are displayed at the top of the list box.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

For some measurements, for example, second and third order intercept functions, the Direct Plot form displays a second list of variable values for the reference variable. The following example shows the Direct Plot form for 3rd order IPN curves.

**Figure 2-3 Direct Plot form for IP3 Curve**

**Function**

<input type="radio"/> Voltage	<input type="radio"/> Current
<input type="radio"/> Power	<input type="radio"/> Voltage Gain
<input type="radio"/> Current Gain	<input type="radio"/> Power Gain
<input type="radio"/> Transconductance	<input type="radio"/> Transimpedance
<input type="radio"/> Compression Point	<input checked="" type="radio"/> IPN Curves
<input type="radio"/> Power Contours	<input type="radio"/> Reflection Contours
<input type="radio"/> Harmonic Frequency	<input type="radio"/> Power Added Eff.
<input type="radio"/> Power Gain Vs Pout	<input type="radio"/> Comp. Vs Pout
<input type="radio"/> Node Complex Imp.	

Select

Single Point Input Power Value (dBm)

     Order

3rd Order Harmonic	1st Order Harmonic
0      0	0      0
1      40M	1      40M
2      80M	2      80M
3      120M	3      120M
4      160M	4      160M
5      200M	5      200M

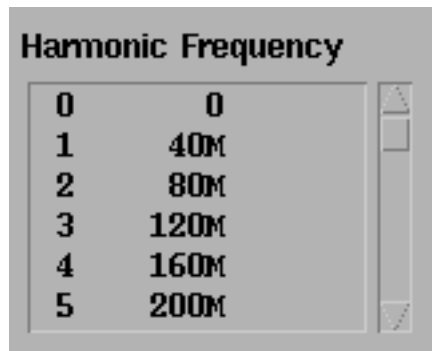
Add To Outputs

> Assign Single Point Input Power on this form...

Follow the messages at the bottom of the Direct Plot form and click the design variable values you want to plot.

## Selecting Sidebands and Harmonics

When Sidebands or Harmonics are available, they are displayed in one or more list boxes at the bottom of the Direct Plot form.



As before, follow the messages at the bottom of the Direct Plot form and click sideband or harmonic values in the list box to select them.

To select one value, simply click it.

- To select two or more adjacent values, click and drag with the mouse over the values you want to select.
- To select two or more values that are not adjacent, hold the `Control` key down while you click the individual values.
- To deselect a value, hold the `Control` key down while you click a selected value.

## Generating a Spectral Plot

Spectral plots show the function level at each frequency component of a single analysis point (one step in a sweep of frequency, input power, or design variable).

In the Simulation window, choose *Results – Direct Plot – Main Form* to open the Direct Plot form, the Waveform window, and the design in the Schematic window.

Follow the prompts displayed at the bottom of the Direct Plot form for instructions. Sometimes additional information is also displayed elsewhere in the form.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

In the Direct Plot form, click a *Plot Mode* button to specify whether the curves you plot are appended to or replace any existing curves in the Waveform window.

Click an *Analysis* button to select the simulation for which you want to plot results.

Click a *Function* button to select the function or measurement you want to plot.

The *Function* values displayed vary with the simulations run. If you need a function that is not included on the Direct Plot form, use the RF version of the analog design environment calculator. See the *Waveform Calculator* documentation for more information.

In the *Select* cyclic field, determine what to select in the Schematic window.

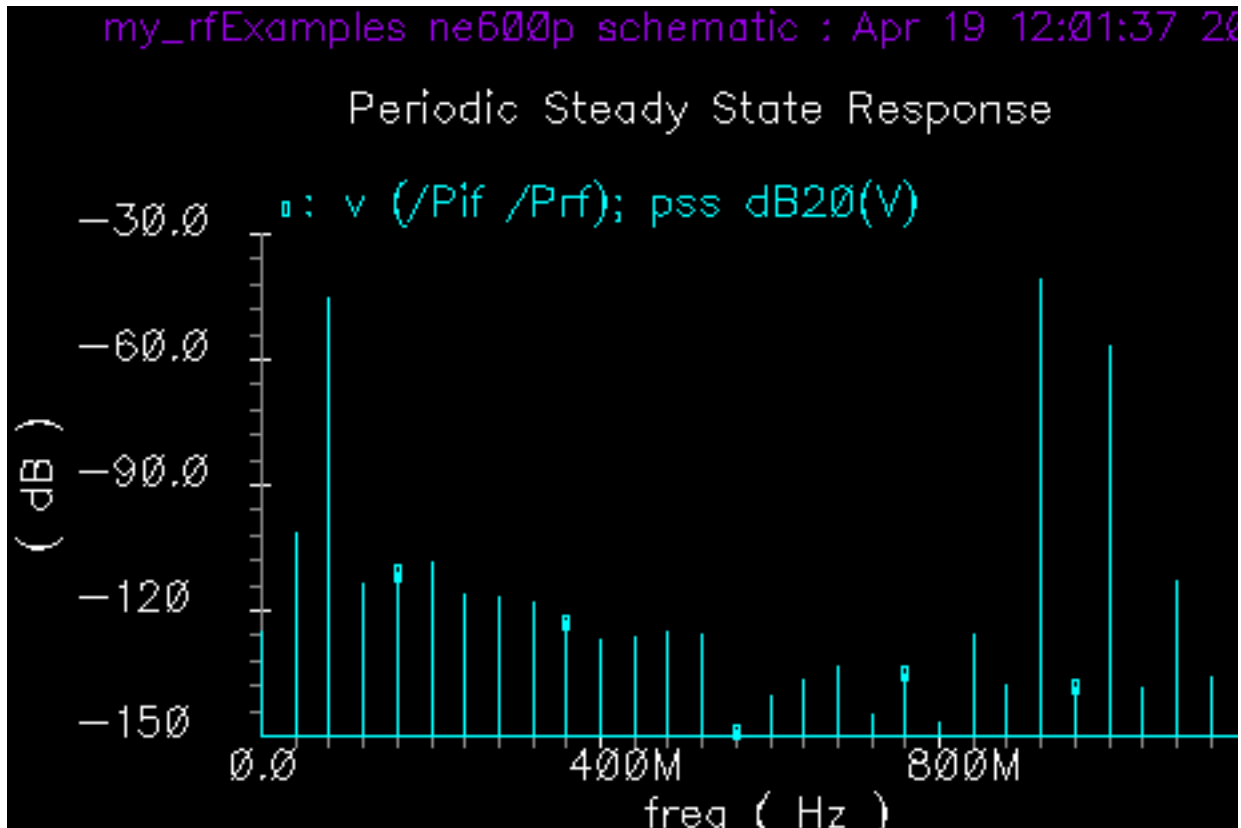
All *Function* values now have several *Select* options. For example, for the *Voltage* function, you can now choose between *Net*, *Differential Nets* and *Instance with 2 Terminals*. Follow the prompts displayed at the bottom of the Direct Plot form for more information.

For PSS sweeps, set *Sweep* to *spectrum*.

For some RF functions, set *Signal Level*, choices are *peak* or *rms*.

1. For some RF functions, set a plot *Modifier*, choices are *Magnitude*, *Phase*, *dB20*, *Real*, or *Imaginary*.

2. Follow the prompts at the bottom of the Direct Plot form to draw the curve in the Waveform window.



### Saving a Displayed Output and Displaying Saved Outputs.

In the Plot form, first click the *Add to Outputs* button and then click the *Replot* button to add a plot displayed in the Waveform window to the *Outputs* list in the Simulation window.



- When you select a curve to plot from the list of curves in the *Outputs* section of the Simulation window, the new expression is *always* plotted as the *Magnitude* of the signal, not the plot with your original *Modifier* selection. If you want a specific scale, for example *dB20*, you have two choices. You can either
  - Modify the expression using *Curve – Edit – Scale* in the Waveform window.

- ❑ Paste the expression from the *Outputs Setup* form into the calculator buffer and edit it.

## Changing the Noise Floor of a Spectral Plot

- Use the *Axes – Y axis* command in the Waveform window to change the noise floor of a spectral plot. In the *Axes – Y axis* form, set *Range* to *Min-Max* and specify a *Min* value.

## Generating a Time Waveform

Time waveforms plot voltage and current against time.

Choose *Results – Direct Plot – Main Form* to open the Waveform window, the design in the Schematic window, and the Direct Plot form.

Follow the prompts displayed at the bottom of the Direct Plot form for information on what to do next. Sometimes additional information is also displayed elsewhere in the form.

In the Direct Plot form, click a *Plot Mode* button to specify whether the curves you plot are appended to or replace any existing curves in the Waveform window.

Click an *Analysis* button to select the simulation for which you want to plot results.

Click a *Function* button to select the function or measurement you want to plot.

The *Function* values displayed vary with the simulations run. If you need a function that is not included on the Direct Plot form, use the RF version of the analog design environment calculator. See the *Waveform Calculator* documentation for more information.

In the *Select* cyclic field, determine what to select in the Schematic window.

All *Function* values now have several *Select* options. For example, for the *Voltage* function, you can now choose between *Net*, *Differential Nets* and *Instance with 2 Terminals*. Follow the prompts displayed at the bottom of the Direct Plot form for more information.

For PSS sweeps, set *Sweep* to *time*.

For some RF functions, set *Signal Level*, choices are *peak* or *rms*.

For some RF functions, set a plot *Modifier*, choices are *Magnitude*, *Phase*, *dB20*, *Real*, or *Imaginary*.

Follow the prompts at the bottom of the Direct Plot form to draw the curve in the Waveform window.

### **Saving a Displayed Output and Displaying Saved Outputs.**

In the Plot form, first click the *Add to Outputs* button and then click the *Replot* button to add a plot displayed in the Waveform window to the *Outputs* list in the Simulation window.



When you select a curve to plot from the list of curves in the *Outputs* section of the Simulation window, the new expression is *always* plotted as the *Magnitude* of the signal, not the plot with your original *Modifier* selection. If you want a specific scale, for example *dB20*, you have two choices. You can either

- Modify the expression using *Curve – Edit – Scale* in the Waveform window.
- Paste the expression from the *Outputs Setup* form into the calculator buffer and edit it.

## Plotting Complex Impedance

Spectre RF does not provide a GUI to plot the complex impedance of a node in a Smith Chart. Follow this procedure to create a node complex impedance Smith Chart plot.

To produce the data used to create the transimpedance expression used in [step](#), perform a swept PSS analysis and set the Direct Plot Sweep to *variable*. Otherwise, the harmonic call in the numerator of the transimpedance expression does not appear and you have to manually add this call in [step 1](#).

Select the net that connects to the terminal of interest. Then set the input and output harmonic on the Direct Plot form to the same index.

Use the Direct Plot form to send the transimpedance expression to the ADE Outputs window.

Your transimpedance expression should look similar to the following.

```
(harmonic(v("/net1" ?result "pss_fd") '(1)) /  
:harmonic(i("/I8/out" ?result "pss_fd") '(1)))
```

1. Use a *vi* window (or the Waveform Calculator buffer) along with cutting and pasting from the ADE Setting Outputs form to construct the final gamma expression.

2. (  
transimpedance  
- 50.) / (  
transimpedance  
+ 50.)

Your final gamma expression looks like this.

```
(  
(harmonic(v("/net1" ?result "pss_fd") '(1)) /  
harmonic(i("/I8/out" ?result "pss_fd") '(1)))  
- 50.) / (  
(harmonic(v("/net1" ?result "pss_fd") '(1)) /  
harmonic(i("/I8/out" ?result "pss_fd") '(1)))  
+ 50.)
```

3. Add this new expression to the ADE Outputs form.
4. Click *Plot Outputs* to plot the outputs.
5. In the waveform window, choose *Axes — To Smith — Impedance* to create the desired node complex impedance Smith Chart plot.
6. Use ADE *Save State* to save your expression.

## Field Descriptions for the Direct Plot Form

The following sections describe the fields on the Direct Plot form. The sections are arranged alphabetically, according to the top-level headings on the forms. The top-level headings are usually found along the leftmost margin of the forms.

### 1st Order Harmonic

See [First-Order Harmonic](#).

### 2nd-7th Order Harmonic

See [Nth Order Harmonic](#).

### Add To Outputs

Adds the expression plotted in the Waveform window to the *Outputs* list in the Simulation window.

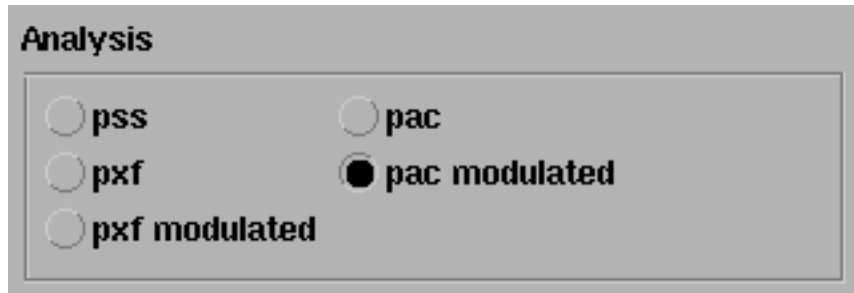


- Highlight *Add To Outputs* to automatically add each new plot to the *Outputs* list whenever you click *Replot*.
- With *Add To Outputs* dehighlighted, click the *Add To Outputs* button to add only the current plot to the *Outputs* list.



## Analysis

Selects the analysis to plot results for.



**Analysis**

pss                       pac

pxf                          pac modulated

pxf modulated

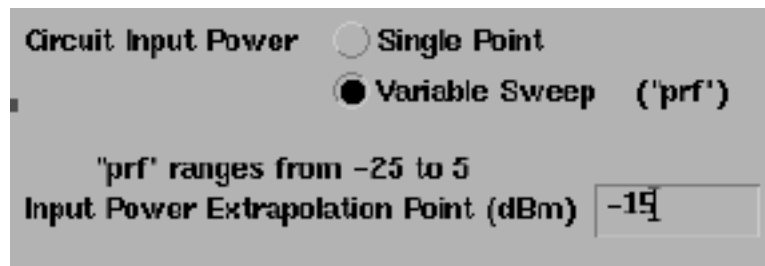
The *Analysis* area lists analyses for which results are available. If you performed one or more small-signal analysis, each small-signal analysis has its own button separate from the button for the large-signal analysis performed as a prerequisite.

When noise separation information has been generated, a *noise separation* choice becomes available. This choice turns on the ability to plot the noise contributions of selected sidebands and objects. For more information, see “[Function](#)” on page 132.

Highlighting an analysis changes the Direct Plot form to display fields for that analysis.

## Circuit Input Power (QPSS, PAC)

Selects between *Single Point* and an appropriate *Sweep*.



**Circuit Input Power**     Single Point

Variable Sweep ('prf')

"prf" ranges from -25 to 5

Input Power Extrapolation Point (dBm)

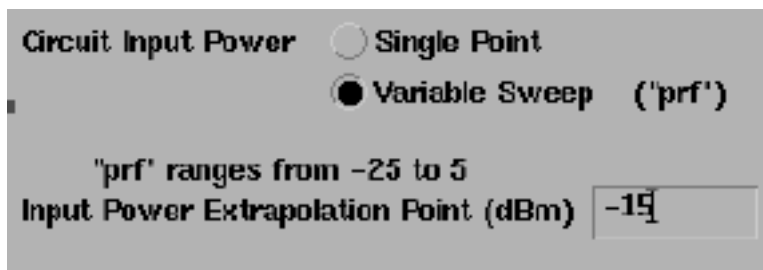
- If you choose *Single Point*, the *Input Power Value (dBm)* field appears, and you must type a value into it.
- If you choose a *Sweep*, the *Extrapolation Point* field appears, and you can type a value. Placing a value in the *Extrapolation Point* field is optional.

## Close Contours (PSS and ENVLP)

See “[Maximum Reflection Magnitude \(ENVLP\)](#)” on page 138.

## Extrapolation Point (PSS and QPSS)

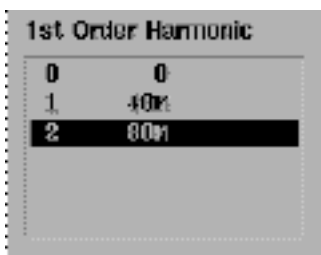
Specifies the value of the design variable from which the straight line approximations of first- and third-order harmonics are produced. See “[Circuit Input Power \(QPSS, PAC\)](#)” on page 129 for more information.



The *Extrapolation Point* field is optional. It specifies the value of the design variable from which the straight line approximations of first- and third-order harmonics are produced. If you do not specify a value, the default value is the smallest x axis sweep value.

## First-Order Harmonic (PSS)

Lists available first-order harmonics when you select the *IPN Curves Function* for a PSS analysis.

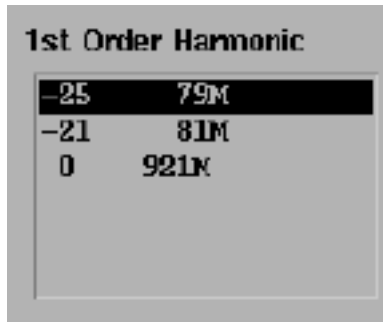


Lists the harmonics available for plotting by number and associated frequency values.

Select one harmonic from the list box and then select the appropriate net on the schematic.

## First Order Harmonic

Lists available first-order harmonics when you select the *IPN Curves Function* for an analysis.



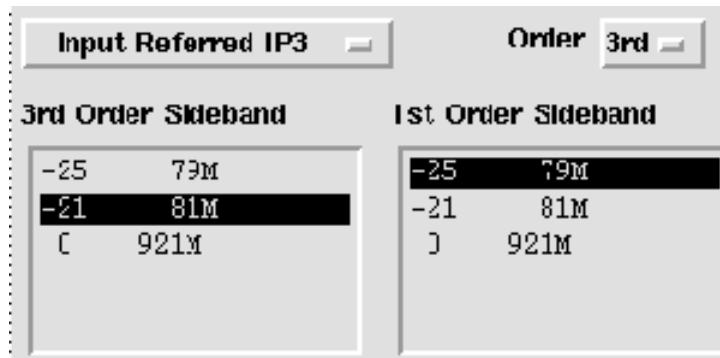
-25	79M
-21	81M
0	921M

In the list box, the first column lists the frequency value of a harmonic, the following columns list the tone coefficient for each fundamental tone that contributed to the harmonic.

Select one harmonic from the list box and then select the appropriate net on the schematic.

## First Order Sideband (PAC)

Lists available first-order sidebands when you select the *IPN Curves Function* for a PAC analysis.



Input Referred IP3  Order

3rd Order Sideband		1st Order Sideband	
-25	79M	-25	79M
-21	81M	-21	81M
0	921M	0	921M

Lists the first-order sidebands available for plotting by number and associated frequency values.

Select one sideband from the list box and then select the appropriate net on the schematic.

## Freq. Multiplier (Pnoise)

Specifies the ratio of the jitter output signal frequency to the PSS fundamental frequency.

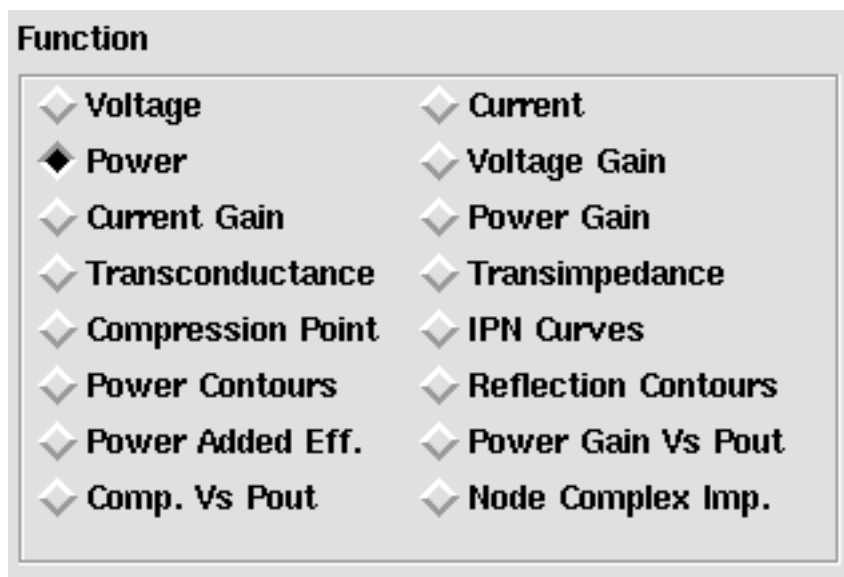


Set this parameter to an integer greater than 1 only when the selected output frequency for measuring jitter is not the same as the PSS fundamental frequency.

For example, if you have an oscillator with a counter and you measure the jitter at the output of the oscillator where the frequency is a harmonic of the fundamental (rather than measuring at the output of the counter where the frequency is the same as the PSS fundamental), then set the *Freq. Multiplier* value to the number of the harmonic. So, if the measured output is the second harmonic of the fundamental, you set *Freq. Multiplier* to 2.

## Function

Specifies a quantity to plot.



Each *Function* button specifies a different quantity that you can plot. The available *Functions* vary depending on the *Analysis* you select. For some functions, you must select one or two objects on the schematic after selecting the *Function* button.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

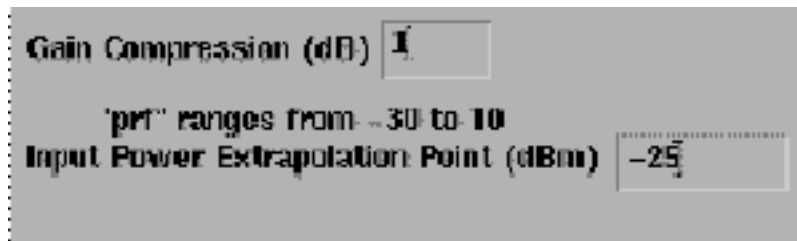
---

When the chosen *Analysis* value is *pnoise separation*, the following become available in the Function pane:

<i>Sideband Output</i>	Plots the noise contribution of selected sidebands.
<i>Source Output</i>	Plots the noise contribution of primary noise sources such as re and rb in a BJT to the output at one selected sideband.
<i>Primary Source</i>	Plots the primary noise sources such as re and rb in a BJT at one selected sideband.
<i>Instance Output</i>	Plots the noise contribution to the output of instances, such as MOS and BJT, of a selected sideband.
<i>Instance Source</i>	Plots the noise sources of some instances at one selected sideband.
<i>Src. Noise Gain</i>	Plots the noise gains of primary noise sources such as re and rb in a BJT from source to output at one selected sideband.

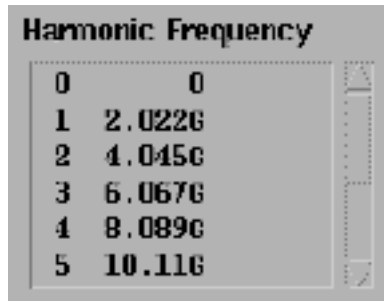
### Gain Compression (PSS and QPSS)

Specifies the *Gain Compression* when you plot the *Compression Point*.



## Harmonic Frequency (PSS)

Lists available harmonic frequencies by number when you select the *Harmonic Frequency* function.



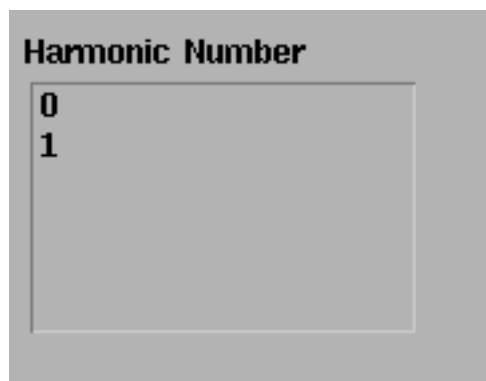
Harmonic Frequency	
0	0
1	2.022G
2	4.045G
3	6.067G
4	8.089G
5	10.11G

Lists the harmonics available for plotting by number and associated frequency values.

Select one harmonic from the list box and then select the appropriate net on the schematic.

## Harmonic Number (ENVLP)

Lists available harmonics by number when you select the *Power Function* for an ENVLP analysis.



Harmonic Number	
0	
1	

Lists the harmonics available for plotting by number and associated frequency values.

Select one harmonic from the list box and then select the appropriate net on the schematic.

### Input Harmonic (PSS)

Lists available input harmonics by number.

Input Harmonic	
0	0
1	900M
2	1.8G
3	2.7G
4	3.6G
5	4.5G

Lists available input harmonics by number. the list box appears on the PSS Plot form when you select one of the following functions: *Voltage Gain*, *Current Gain*, *Power Gain*, *Transconductance*, or *Transimpedance Functions*.

The values in the list box are those you requested in the *Output Harmonics* specification in the Choosing Analyses form.

### Input Harmonic (QPSS)

Lists available input harmonics.

	Freq. (Hz)	f1o	fund2	frf
Input Harmonic	0	0	0	0
	20M	0	1	-1
	40M	0	2	-2
	60M	1	-2	1
	80M	1	-1	0

In the list box, the first column lists the frequency value of a harmonic. The following columns list the tone coefficients for each fundamental tone that contributed to the harmonic.

## Input Power Value (dBm) (PSS, QPSS, and PAC)

Specifies an input power value.

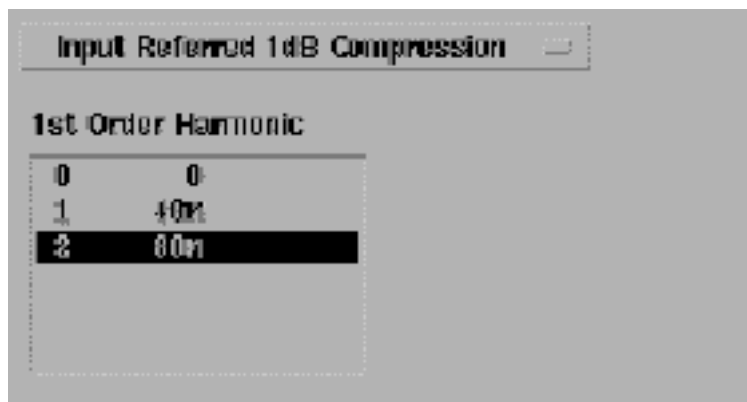
Input Power Value (dBm)

This field appears

- When you select *Single Point* for *Circuit Input Power*.
- When you select the IPN Curves Function.
- When you want information about a single point after running a power sweep.
- Assign an input power value.

## Input or Output Referred 1dB Compression (PSS and QPSS)

Selects input or output referred 1 dB compression.



- Output referred compression is referred to the y axis.
- Input referred compression is referred to the x axis.



## Input or Output Referred IPN and Order (PSS, QPSS, and PAC)

Selects Input or Output Referred Nth-Order Intercept point.

The screenshot shows a software interface with two main settings: 'Input Referred IP3' set to 'Input Referred IP3' and 'Order' set to '3rd'. Below these are two tables for harmonic analysis. The first table is for the '3rd Order Harmonic' and the second is for the '1st Order Harmonic'. Both tables have columns for 'Freq. (Hz)', 'flo', 'fund2', and 'frf'. The '1st Order Harmonic' table has a row for '20M' that is highlighted in black.

	Freq. (Hz)	flo	fund2	frf
3rd Order Harmonic	0	0	0	0
	20M	0	1	-1
	40M	0	2	-2
	60M	1	-2	1
	80M	1	-1	0
1st Order Harmonic	0	0	0	0
	20M	0	1	-1
	40M	0	2	-2
	60M	1	-2	1
	80M	1	-1	0

- Select the *Order*, 2nd through 7th, in the *Order* cyclic field.
- Select *Input Referred IPN* or *Output Referred IPN* in the Input/Output Referred IPN cyclic field.
- Output referred IPN is referred to the y axis.
- Input referred IP3 is referred to the x axis.

## Maximum Reflection Magnitude (ENVLP)

When you select the *Reflection Contours Function* in the envelope analysis:

Output Harmonic	
0	0
1	1G
2	2G
3	3G

- Specifies a maximum reflection magnitude.
- Specifies a minimum reflection magnitude.
- Sets the resistance of the port adapter when you plot *Power* or *Reflection Contours*.
- Specifies the number of *Power* or *Reflection Contours* to plot.
- Sets open or closed contours for *Power* and *Reflection Contours*.
- When you select *Close Contours*, the plot appears as a closed figure. If *Close Contours* is not selected, the plot appears as an open figure. The default is to leave the two most distant points in the plot unconnected.
- Selects the Output harmonic.

## Min Reflection Mag

See [“Maximum Reflection Magnitude \(ENVLP\)”](#) on page 138.

## Modifier

Sets the units for the y axis of the plot.



Choices vary depending on the *Function* highlighted.

**Magnitude** is the raw value, in volts, amps, or no units at all.

**Phase** sets the y axis to degrees.

**dB20** sets the y axis to decibels with tick marks every 20 dB.

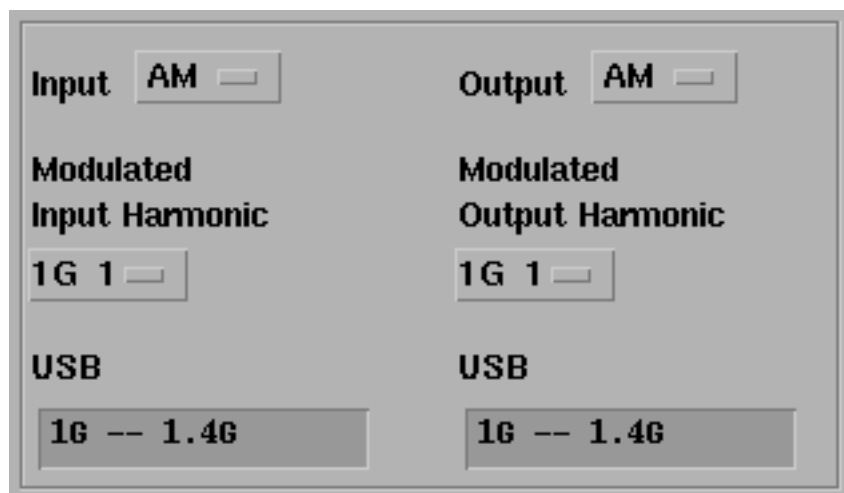
**dBm** sets the y axis to dB 10 plus 30.

**dB10** sets the y axis to decibels with tick marks every 10 dB.

**Real** and **Imaginary** restrict plots to only the real or imaginary range of the curve.

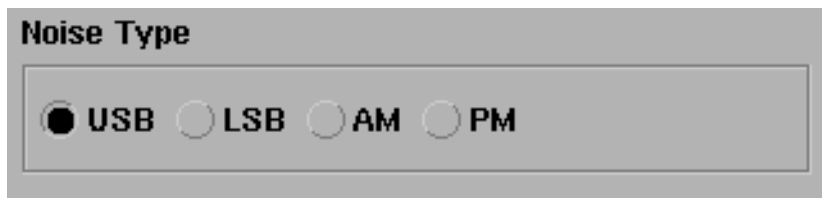
## Modulated Input/Output (PAC and PXF)

Displays information so you can plot PAC and PXF modulated input and output curves.



## Noise Type

Lists the types of noise calculated following a Pnoise analysis with *modulated Noise Type* selected.



Choices may vary.

**USB** is the upper sideband noise.

**LSB** is the lower sideband noise.

**AM** is the amplitude modulated noise.

**PM** is the phase modulated noise.

## Number of Contours

See "[Maximum Reflection Magnitude \(ENVLP\)](#)" on page 138.

Lists the harmonics available for plotting by number and associated frequency values. (Select the *Order*, 2nd through 7th, in the *Order* cyclic field.)

- Select one harmonic from the list box and then select the appropriate net on the schematic.

### Nth Order Harmonic (QPSS)

Lists available Nth Order Harmonics when you select the *IPN Curves Function* for the QPSS analysis.

3rd Order Harmonic	
-25	79M
-21	81M
0	921M

In the list box, the first column lists the frequency value of a harmonic. The following columns list the tone coefficient for each fundamental tone that contributed to the harmonic.

- Select one harmonic from the list box and then select the appropriate net on the schematic.

### Nth Order Sideband (PAC)

Lists available Nth Order sidebands when you select the *IPN Curves Function* for a PAC analysis.

Input Referred IP3		Order
		3rd
3rd Order Sideband		1st Order Sideband
-25	79M	-25 79M
-21	81M	-21 81M
0	921M	0 921M

Lists the sidebands available for plotting by number and associated frequency values. (Select the *Order*, 2nd through 7th, in the *Order* cyclic field.)

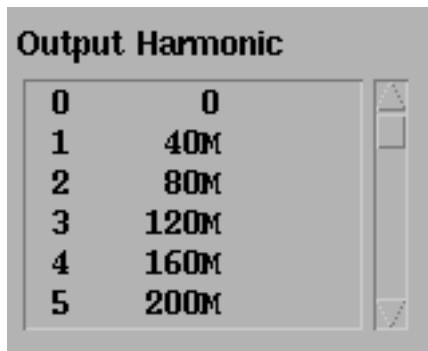
- Select one sideband from the list box and then select the appropriate net on the schematic.

## Order

See “[Input or Output Referred IPN and Order \(PSS, QPSS, and PAC\)](#)” on page 137.

## Output Harmonic (PSS)

Lists available Output Harmonics for PSS analysis.



Lists the harmonics available for plotting by number and associated frequency.

Click harmonics in the list box to select them.

Select adjacent harmonics by clicking and dragging with the mouse over the harmonics you want to select.

Select harmonics that are not adjacent by holding the `Control` key down while you click the individual sidebands.

Deselect harmonics by holding the `Control` key down while you click a selected harmonic.

## Output Harmonic (For QPSS)

Lists available Output Harmonics for QPSS analysis.

	Freq. (Hz)	f1o	fund2	frf
<b>Output Harmonic</b>	0	0	0	0
	20M	0	1	-1
	40M	0	2	-2
	60M	1	-2	1
	80M	1	-1	0

In the list box, the first column is the frequency of a harmonic. The second and third columns specify the tone coefficients for each fundamental tone that contributed to the listed harmonic.

Click harmonics in the list box to select them.

Select adjacent harmonics by clicking and dragging with the mouse over the harmonics you want to select.

Select harmonics that are not adjacent by holding the `Control` key down while you click the individual sidebands.

Deselect harmonics by holding the `Control` key down while you click a selected harmonic.

## Output Sideband (PAC and PXF)

Lists the sidebands you requested on the small-signal Choosing Analyses form.

<b>Output Harmonic</b>	
-25	1G
-21	840M
0	921

This list box appears when you choose variable for *Sweep* on a small-signal Direct Plot form. It lists all the sidebands you requested on the small-signal Choosing Analyses form.

Click sidebands in the list box to select them.

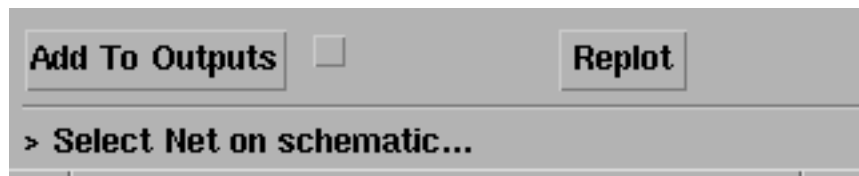
Select adjacent sidebands by clicking and dragging with the mouse over the sidebands you want to select.

Select sidebands that are not adjacent by holding the `Control` key down while you click the individual sidebands.

Deselect sidebands by holding the `Control` key down while you click a selected sideband.

## Plot and Replot

Displays a plot in the Waveform window.



## Plot Mode

Determines whether to add the next plot to those currently displayed in the Waveform window or to clear the window and display only the next plot.



- *Append* combines the next plot with other curves already plotted in the Waveform window.
- *Replace* clears the Waveform window just prior to displaying the next plot.



## Power Spectral Density Parameters (ENVLP)

Determines how the Power Spectral Density is calculated.

**Power Spectral Density Parameters**

**Time Interval**

From  To

Nyquist half-bandwidth

Frequency bin width

Max. plotting frequency

Min. plotting frequency

Windowing

Detrending

- **Time Interval** – The starting and ending times for the spectral analysis interval. They are usually the start and stop times for the simulator.
- **Get From Data** -- Sets the *From* and *To* values to match the values recorded in the results data.
- **Nyquist half-bandwidth** – The maximum frequency at which there are signals of interest. This is usually three to five times the maximum band frequency.
- **Frequency bin width** – The frequency resolution, such as the width of the frequency bins.
- **Max. plotting frequency** – Sets the maximum x axis value for the waveform you want to plot.
- **Min. plotting frequency** – Sets the minimum x axis value for the waveform you want to plot.

- **Windowing** – A preset list of available windowing functions used during the spectrum calculation.
- **Detrending** – Removes trends from the data before the spectral analysis.

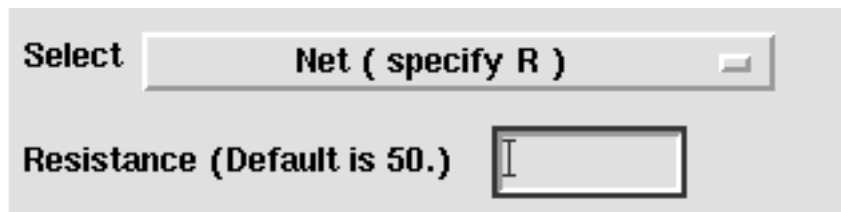
### Reference Resistance (ENVLP)

See “[Maximum Reflection Magnitude \(ENVLP\)](#)” on page 138.

### Resistance

Sets the resistance of the port adapter when you plot *Power* or *Reflection Contours*.

Sets the resistance of the port adapter when you select Net (specify R) and plot Power or Reflection Contours.



The image shows a dialog box with a dropdown menu and a text input field. The dropdown menu is labeled "Select" and currently shows "Net ( specify R )". Below it, the text "Resistance (Default is 50.)" is followed by an empty text input field.

## Select

Determines the type and number of objects to select on the schematic. There is a prompt at the bottom of the form describing how to make the selection.

Select

Signal Level  peak  rms

Modifier

Magnitude  Phase  dB20  
 Real  Imaginary

Add To Outputs  Replot

> Select Net on schematic...

Choices available in the *Select* cyclic field vary depending on the highlighted *Function*. The message at the bottom of the form prompts you to make an appropriate selection.

**Differential Nets** -- Select differential nets on the schematic.

**Differential Nets (dB, 1ohm reference)** -- Select differential nets on the schematic.

**Differential Terminal** -- Select a differential terminal on schematic.

**Instance with 2 Terminals** -- Select an instance with two terminals on the schematic.

**Net** -- Select a net on the schematic.

**Net (dB, 1ohm reference)** -- Select a net on the schematic.

**Out. and In. Ports (fixed R(OutPort))** -- Select output and input ports on the schematic.

- **Out. and In. Instances with 2 Terminals** -- Select output and input instances with two terminals on the schematic.
- **Output and Input Nets** -- Select output and input nets on the schematic.
- **+ Output and + Input Nets** -- Select output and input nets on the schematic.

- **Output and Input Terminals** -- Select output and input terminals on the schematic.
- **+ Output and +- Input Terminals** -- Select output and input terminals on the schematic.
- **Output Net and Input Terminal** -- Select an output net and an input terminal on the schematic.
- **+ Output Net and Input Terminal** -- Select an output net and an input terminal on the schematic.
- **Port (fixed R(port))** -- Select a port on the schematic.
- **+ Power and +- Refl Terminals** -- Select power and reflection terminals on the schematic.
- **Separate Power and Refl Terminals** -- Select separate power and reflection terminals on the schematic.
- **Single Power/Refl Terminal** -- Select one power or reflection terminal on the schematic.
- **Single Power/Refl Term and ref Term** -- Select one power or reflection terminal and a reference terminal on the schematic.
- **Terminal** -- Select a terminal on the schematic.
- **Terminal and V-Reference Terminal** -- Select a terminal and a V-Reference terminal on the schematic.

## Signal Level (PSS, QPSS)

Determines the signal value to plot.



- **Peak** -- Plots the maximum value of the signal.
- **rms** -- Plots the root-mean-square value, or effective value, of the signal.

## Sweep (PSS, PXF, and ENVLP)

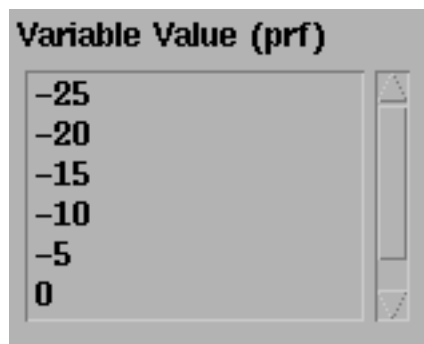
Sets the units for the x axis of the plot. Choices vary depending on the *Function* highlighted.



- *spectrum*, *sideband*, and *frequency* set the x axis to display frequency.
- *time* sets the x axis to display seconds.
- *variable* sets the x axis to display the value of a design variable.

## Variable Value (PSS, QPSS, and PAC)

Lists the swept variable values that you can plot for a PSS, QPSS, or PAC analysis.



The PSS, QPSS, and PAC Results forms display this list of sweep values that you can specify for a specific variable, temperature, component parameter, or model parameter. The variable name is included in the title.

The range of values is determined by the *Sweep Range* specification in the Choosing Analyses form. The number of values is determined by the *Sweep Type* specification in the Choosing Analyses form.

In the sample figure, which shows the PSS version, the values are listed for the design variable `prf`. The QPSS version of the form is formatted slightly differently but gives the same information.

## ACPR Wizard

The ACPR Wizard simplifies the procedure for measuring ACPR and PSD.

Open the ACPR wizard in one of two ways.

In the Simulation window,

- Choose *Tools – RF – Wizards – ACPR*
- or
- In the *ENVLP Choosing Analyses* form, click *Start ACPR Wizard*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

The ACPR wizard form opens.

ACPR Wizard

OK Cancel Apply Help

Clock Name

How to Measure

Net

Channel Definitions

Main Channel Width (Hz)

Adiacent frequencies are specified relative to the center of main channel

name	from (Hz)	to (Hz)
lower	-915K	-885K
upper	885K	915K

Simulation Control

Stabilization Time (Sec)

Resolution Bandwidth (Hz)

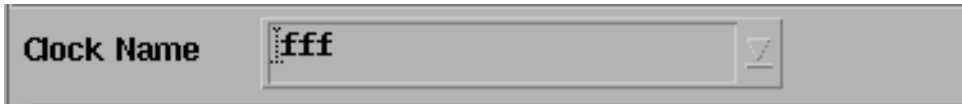
Repetitions

Windowing Function

The following sections describe the fields on the ACPR Wizard form.

## Clock Name

In the *Clock Name* cyclic field, select the clock signal from those listed in the cyclic field.



The *Clock Name* identifies the source of the modulated signal.

## How to Measure

The *How to Measure* section allows you to choose whether to measure ACPR for a single *Net* or between *Differential Nets*. Use the *How to Measure* cyclic field to choose a single *Net* or *Differential Nets*.

To measure ACPR for a single net,

1. Select *Net* in the *How to Measure* cyclic field.
2. Click *Select*. Then select the output net in the Schematic window.
3. *RFOUT* displays in the *Net* field.

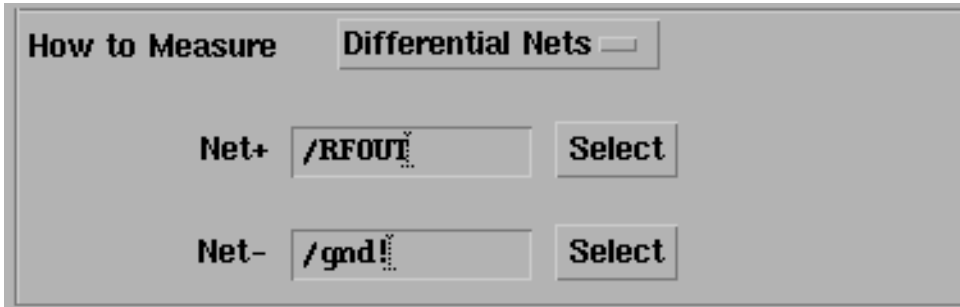


To measure ACPR for differential nets,

1. Select *Differential Nets* in the *How to Measure* cyclic field.
2. Click *Select* next to the *Net+* field. Then select the positive net in the Schematic window.
3. The signal name displays in the *Net+* field.
4. Click *Select* next to the *Net-* field. Then select the negative net in the Schematic window.



5. The signal name displays in the *Net-* field.



The screenshot shows a dialog box titled "How to Measure" with a dropdown menu set to "Differential Nets". Below this, there are two rows of input fields. The first row is labeled "Net+" and contains the text "/RFOUT" in a text box, followed by a "Select" button. The second row is labeled "Net-" and contains the text "/gnd" in a text box, followed by a "Select" button.

## Channel Definitions

In the *Channel Definitions* cyclic field, select a preset channel definition. Choices include *Custom*, *IS-95*, and *W-CDMA*.

When you select *IS-95*

- The *Main Channel Width (Hz)* field is calculated. This is the width of the main channel in Hz. Enter a number greater than zero. When you click *OK* or *Apply*, the content of this field is verified.
- The *adjacent frequencies* are determined and display in the list box. Note that adjacent frequencies are specified relative to the center of the main channel.
- Use the edit fields and the *Add*, *Change*, and *Delete* buttons to enter or modify channel definitions in the list box. You can hand edit or enter adjacent frequency names and

upper and lower boundaries. Specify adjacent frequencies relative to the center of the main channel. All channel widths must be greater than zero.

**Channel Definitions** IS-95

**Main Channel Width (Hz)** 1.2288M

Adjacent frequencies are specified relative to the center of main channel

name	from (Hz)	to (Hz)
lower	-915K	-885K
upper	885K	915K

Add Change Delete

## Simulation Control

*Stabilization Time (Sec.)* specifies the number of seconds to wait before using the data for analysis.

*Resolution Bandwidth (Hz)* specifies the spacing of data points on the on the power density curve, in Hz. Use the Calculate button to calculate the resolution bandwidth. When you decrease the resolution bandwidth, simulation time is longer and the data file is larger.

*Repetitions* specifies the number of times to repeat the DFT for averaging. When you increase the number of repetitions, the power density curve is smoother, simulation time is longer and the data file is larger.

In the *Windowing Function* cyclic field, select a preset windowing function. Choices include: *Blackman, Cosine2, Cosine4, ExtCosBell, HalfCycleSine, HalfCycleSine3, HalfCycleSine6, Hamming, Hanning, Kaiser, Parzen, Rectangular* and *Triangular*.

A *Windowing Function* tapers the signal before performing the DFT to reduce the effect of any edge discontinuities.

**Simulation Control**

Stabilization Time (Sec)

Resolution Bandwidth (Hz)

Repetitions

Windowing Function

## OK and Apply

When you click *OK* or *Apply*, the values you entered in the ACPR wizard are used to determine values for the required ENVLP analysis and the ENVLP choosing analyses form is filled in.

**ACPR Wizard**

Clock Name

In the ENVLP choosing analyses form, values are calculated as follows.

- The *Clock Name* field in the ENVLP form is the same as the *Clock Name* field on the ACPR wizard.
- The *Stop Time* value in the ENVLP form is calculated.
- The *Output Harmonics* field in the ENVLP form is set to 1.
- The ENVLP choosing analyses form is enabled.

- The *Start* field in the ENVLP Options form is left blank.
- The *modulationbw* value for the ENVLP Options form is calculated.
- The *strobeperiod* value for the ENVLP Options form is calculated.
- You can modify these values entered on the ENVLP analysis form, but your changes are not propagated back to the ACPR wizard.
- The *Analyses* area in the Simulation window reflects the ENVLP analysis.

Analyses					
#	Type	Arguments.....			Enable
1	envlp	0	266.9u	fff	1 .. yes

When the ENVLP analysis completes, the ACPR values for each channel and the PSD waveform display in the *Outputs* area in the Simulation window.

Outputs				
#	Name/Signal/Expr	Value	Plot	Save March
1	ACPR psd /RFOUT	wave	yes	
2	ACPR lower	-61.57		
3	ACPR upper	-60.88		

Plotting mode: Replace

## Large Signal S-Parameter Wizard

The Large Signal S-Parameter (LSSP) Wizard simplifies the procedure for measuring large signal S-parameters with the Spectre RF simulator.

To open the LSSP Wizard form,

- In the Simulation window, choose *Tools – RF – Wizards – LSSP*
- The LSSP Wizard form opens.

**Define Input/Output**

#	Name	Res	Freq	Value	Power	Value	Type
2	PORT2	50	fout	1G	pout	13.66	Output
1	PORT1	50	fin	1G	pin	-10	Input

PORT1   50   fin   pin   type

Change

**Sweep**    Amplitude    Frequency    Disable

**Sweep Range**

Start-Stop   Start    Stop

Center-Span

**Sweep Type**

Linear    Step Size  

Logarithmic    Number of Steps

The following sections describe the fields on the LSSP Wizard form.

## Define Input/Output

In the *Define Input/Output* table, specify both the input and output ports. Ensure that the frequency and the amplitude of the ports are set to variables because it must be possible to sweep both the frequency and the power for both the input and output.

Two PSS sweeps are required for LSSP. The two sweeps require different source statuses. For the input power sweep, `port2` acts as a resistance when `port1` is a normal sine source. For the output power sweep, `port1` acts as a resistance when `port2` is a normal sine source.

**Define Input/Output**

#	Name	Res	Freq	Value	Power	Value	Type
2	PORT2	50	fout	1G	pout	13.66	Output
1	PORT1	50	fin	1G	pin	-10	Input

PORT1

50

fin

pin

type

-

**Change**

Use the *type* field to specify whether a port is *Input* or *Output*.

Use the *Change* button to modify port definitions.

## Sweep

The *Sweep* setting affects only the input port. The sweep for the output port is set up by the simulator after the input sweep finishes.

The screenshot shows a dialog box for configuring the sweep. At the top, there are three radio buttons: **Sweep** (selected), **Amplitude**, **Frequency**, and **Disable**. Below this is the **Sweep Range** section, which has two radio buttons: **Start-Stop** (selected) and **Center-Span**. To the right of these are two input fields: **Start** with the value `500M` and **Stop** with the value `5G`. The **Sweep Type** section has two radio buttons: **Linear** (selected) and **Logarithmic**. To the right are two more radio buttons: **Step Size** (selected) and **Number of Steps**. Next to **Step Size** is an input field containing the value `1`.

## Sweep Range

Defines the sweep range for the analysis. Choices are: *Start-Stop*, and *Center-Span*. When you select one of these choices, the adjacent fields change to let you specify appropriate data.

This screenshot shows a close-up of the Sweep Range section. It features two radio buttons: **Start-Stop** (selected) and **Center-Span**. To the right are two input fields: **Start** with the value `500M` and **Stop** with the value `5G`.

## Start - Stop

Defines the beginning and ending points for the sweep.

This screenshot shows a close-up of the Start-Stop Sweep Range section. It features two radio buttons: **Start-Stop** (selected) and **Center-Span**. To the right are two input fields: **Start** with the value `500M` and **Stop** with the value `5G`.

1. Highlight *Start-Stop*.

The form changes to let you type the start and stop points.

2. Type the initial point for the sweep in the *Start* field.
3. Type the final point in the *Stop* field.

### ***Center - Span***

Defines the center point for the sweep and its span.

The screenshot shows a form titled "Sweep Range". On the left, there are two radio button options: "Start-Stop" (which is unselected) and "Center-Span" (which is selected). To the right of these options are two input fields. The first is labeled "Center" and the second is labeled "Span". Both fields are currently empty.

1. Highlight *Center-Span*.

The form changes to let you type the center point and span.

2. Type the midpoint for the sweep in the *Center* field.
3. Type the span in the *Span* field.

### **Sweep Type**

Specifies whether the sweep is linear or logarithmic.

#### ***Linear***

Specifies a linear sweep.

The screenshot shows a form titled "Sweep Type". On the left, there are two radio button options: "Linear" (which is selected) and "Logarithmic" (which is unselected). To the right of these options are two radio button options: "Step Size" (which is selected) and "Number of Steps" (which is unselected). To the right of the "Step Size" option is an input field containing the number "1".

1. Select *Linear*.
2. Do one of the following:
  - Highlight *Step Size* and type the size of each step in the field.



- Highlight *Number of Steps* and type the number of steps (points) in the field.

### **Logarithmic**

Specifies a logarithmic sweep.

**Sweep Type**

<input type="radio"/> Linear	<input checked="" type="radio"/> Points Per Decade	<input type="text" value="1"/>
<input checked="" type="radio"/> Logarithmic	<input type="radio"/> Number of Steps	

1. Select *Logarithmic*.
2. Do one of the following:
  - Highlight *Points per Decade* and type the number of points per decade in the field.
  - Highlight *Number of Steps* and type the number of steps (steps) in the field.

### **OK and Apply**

When you click *OK* or *Apply*, the values you entered in the LSSP wizard are used.

OK Cancel Apply Help

## The Spectre RF Simulation Forms Quick Reference

The Spectre RF simulation forms include the following:

- [“Choosing Analyses Form”](#) on page 162
- [“Option Forms”](#) on page 172 (One for each analysis)
- [“Direct Plot Form”](#) on page 174
- [“ACPR Wizard”](#) on page 150
- [“Large Signal S-Parameter Wizard”](#) on page 156

The simulation forms change to display only the fields relevant for the currently selected analysis. The field description topics for each analysis form are briefly described here and linked to the detailed description in this chapter.

### Choosing Analyses Form

The Choosing Analyses form changes depending on which analysis is selected. For guidance on what the form contains for a particular analysis, choose the appropriate link here.

- [“Periodic Steady-State \(PSS\) Choosing Analyses Form”](#) on page 163
- [“Quasi-Periodic Steady State \(QPSS\) Choosing Analyses Form”](#) on page 164
- [“Envelope \(ENVLP\) Choosing Analyses Form”](#) on page 165
- [“Periodic AC \(PAC\) Choosing Analyses Form”](#) on page 165
- [“Periodic Stability \(PSTB\) Choosing Analyses Form”](#) on page 166
- [“Periodic Noise \(Pnoise\) Choosing Analyses Form”](#) on page 167
- [“Periodic Transfer Function \(PXF\) Choosing Analyses Form”](#) on page 168
- [“Periodic S-Parameter \(PSP\) Choosing Analyses Form”](#) on page 169
- [“Quasi-Periodic Noise \(QPnoise\) Choosing Analyses Form”](#) on page 169
- [“Quasi-Periodic AC \(QPAC\) Choosing Analyses Form”](#) on page 170
- [“Quasi-Periodic Transfer Function \(QPXF\) Choosing Analyses Form”](#) on page 171
- [“Quasi-Periodic S-Parameter \(QPSP\) Choosing Analyses Form”](#) on page 171

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

### Periodic Steady-State (PSS) Choosing Analyses Form

Field or Pane	User Interface Help (page)
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Engine</i> specifies the method used to analyze the design.	<a href="#">“Engine (ENVLP, PSS, QPSS)”</a> on page 37
<i>Fundamental Tones</i> displays and edits information for top level tones in the circuit.	<a href="#">“Fundamental Tones (PSS and QPSS)”</a> on page 42
<i>Beat Frequency, Beat Period, Auto Calculate</i> determine whether the PSS analysis uses <i>Beat Frequency</i> or <i>Beat Period</i> .	<a href="#">“Beat Frequency, Beat Period, and Auto Calculate (PSS)”</a> on page 35
<i>Output Harmonics</i> selects and defines output harmonics.	<a href="#">“Output Harmonics (PSS and ENVLP)”</a> on page 67
<i>Accuracy Defaults</i> quickly adjusts simulation parameters.	<a href="#">“Accuracy Defaults (errpreset) (PSS, QPSS, and ENVLP)”</a> on page 33
<i>Additional Time for Stabilization</i> allows time for stabilization.	<a href="#">“Additional Time for Stabilization (tstab) (PSS and QPSS)”</a> on page 34
<i>Save Initial Transient Results</i> saves the initial transient solution.	<a href="#">“Save Initial Transient Results (PSS and QPSS)”</a> on page 73
<i>Oscillator</i> defines the simulation for an oscillator circuit. (Displays additional fields to specify oscillator analysis.)	<a href="#">“Oscillator (PSS)”</a> on page 63
<i>Sweep</i> selects swept analysis. (Displays additional fields to specify sweep.)	<a href="#">“Sweep (PSS and QPSS)”</a> on page 92
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

### Modifications to PSS Form for Oscillator Analysis

[Oscillator Node](#) and [Reference Node](#) specify how the PSS oscillator analysis is performed.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

---

### **Modifications to PSS Form for Swept PSS Analysis**

Sweep, Sweep Range, Sweep Type, and Add Specific Points specify how the PSS sweep is performed.

### **Quasi-Periodic Steady State (QPSS) Choosing Analyses Form**

---

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<u>"Analysis"</u> on page 34
<i>Engine</i> specifies the method used to analyze the design.	<u>"Engine (ENVLP, PSS, QPSS)"</u> on page 37
<i>Fundamental Tones</i> displays and edits information for top level tones in the circuit.	<u>"Fundamental Tones (PSS and QPSS)"</u> on page 42
<i>Harmonics</i> displays fields used to specify harmonics.	<u>"Harmonics (QPSS)"</u> on page 46
<i>Accuracy Defaults</i> quickly adjusts simulation parameters.	<u>"Accuracy Defaults (errpreset) (PSS, QPSS, and ENVLP)"</u> on page 33
<i>Additional Time for Stabilization</i> allows time for stabilization.	<u>"Additional Time for Stabilization (tstab) (PSS and QPSS)"</u> on page 34
<i>Save Initial Transient Results</i> saves the initial transient solution.	<u>"Save Initial Transient Results (PSS and QPSS)"</u> on page 73
<i>Sweep</i> selects swept analysis. (Displays additional fields to specify sweep.)	<u>"Sweep (PSS and QPSS)"</u> on page 92
<i>Enabled</i> includes this analysis in the next simulation.	<u>"Enabled"</u> on page 37
<i>Options</i> displays the Options form for this analysis.	<u>"Options"</u> on page 62

---

### **Modifications to QPSS Form for Swept QPSS Analysis**

Sweep, Sweep Range, Sweep Type, and Add Specific Points specifies how the QPSS sweep is performed.

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Spectre RF Simulation Form Reference

---

### Envelope (ENVLP) Choosing Analyses Form

---

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Engine</i> specifies the method used to analyze the design.	<a href="#">“Engine (ENVLP, PSS, QPSS)”</a> on page 37
<i>Fund Frequency</i> specifies the frequency of the clock fundamental.	<a href="#">“Fundamental Tones (PSS and QPSS)”</a> on page 42
<i>Period</i> specifies the period of the clock fundamental. For autonomous circuits, <i>Period</i> specifies the estimated period.	<a href="#">“Period (ENVLP)”</a> on page 70
<i>Clock Name</i> and <i>Select Clock Name</i> select the clock signal for the analysis.	<a href="#">“Clock Name and Select Clock Name Button (ENVLP)”</a> on page 36
<i>Stop Time</i> specifies the end time for the analysis.	<a href="#">“Stop Time (ENVLP)”</a> on page 92
<i>Output Harmonics</i> selects and defines output harmonics.	<a href="#">“Output Harmonics (PSS and ENVLP)”</a> on page 67
<i>Accuracy Defaults</i> quickly adjusts simulation parameters.	<a href="#">“Accuracy Defaults (errpreset) (PSS, QPSS, and ENVLP)”</a> on page 33
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

---

### Periodic AC (PAC) Choosing Analyses Form

---

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>PSS Beat Frequency (Hz)</i> displays the Beat Frequency for the associated PSS analysis.	<a href="#">“PSS Beat Frequency (PAC, PSTB, Pnoise, and PXF)”</a> on page 72

---

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Spectre RF Simulation Form Reference

Field or Pane	User Interface Help (page)
<i>Sweeptype</i> , <i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweeptype (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38
<i>Sidebands</i> selects the set of periodic small-signal output frequencies of interest.	<a href="#">“Sidebands (PAC, Pnoise, and PXF)”</a> on page 78
<i>Specialized Analyses (PAC)</i> specifies Modulated analysis.	<a href="#">“<b>Specialized Analyses (PAC)</b>”</a> on page 85
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

**Modifications to PAC Form for Swept PSS Analysis**

[Sweeptype](#), [Frequency Sweep Range](#), [Single-Point](#), and [Freq](#) specify the frequency for the small-signal analysis that follows each swept PSS analysis.

**Periodic Stability (PSTB) Choosing Analyses Form**

Field or Pane	User Interface Help (page)
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>PSS Beat Frequency (Hz)</i> displays the <i>Beat Frequency</i> for the associated PSS analysis.	<a href="#">“PSS Beat Frequency (PAC, PSTB, Pnoise, and PXF)”</a> on page 72
<i>Periodic Stab Analysis Notification (PSTB)</i> sets up the beginning and end points of the sweep.	<a href="#">“Periodic Stab Analysis Notification (PSTB)”</a> on page 70
<i>Sweep Type (PSTB)</i> determines whether the sweep is linear, logarithmic, or chosen automatically.	<a href="#">“Sweep Type (PSTB)”</a> on page 98
<i>Add Specific Points</i> helps set up the sweep for the analysis.	<a href="#">“Probe Instance (PSTB)”</a> on page 72

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

Field or Pane	User Interface Help (page)
<i>Probe Instance (PSTB)</i> specifies a probe that is placed in the feedback loop to identify and characterize the particular loop of interest. Introducing the probe component must not change the circuit characteristics. The probe must be a gain instance, such as a bjt transistor or a mos transistor.	<a href="#">“Probe Instance (PSTB)”</a> on page 72
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

### Periodic Noise (Pnoise) Choosing Analyses Form

Field or Pane	User Interface Help (page)
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>PSS Beat Frequency (Hz)</i> displays the <i>Beat Frequency</i> for the associated PSS analysis.	<a href="#">“PSS Beat Frequency (PAC, PSTB, Pnoise, and PXF)”</a> on page 72
<i>Sweeptype</i> , <i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweeptype (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38
<i>Sidebands</i> selects the set of periodic small-signal output frequencies of interest.	<a href="#">“Sidebands (PAC, Pnoise, and PXF)”</a> on page 78
<i>Output</i> , <i>Input Source</i> , and <i>Reference Side-Band</i> selects the output, noise generator, and reference sidebands for the Pnoise analysis. (Displays additional fields.)	<a href="#">“Output (Pnoise and QPnoise)”</a> on page 65, <a href="#">“Input Source and Reference Side-Band (Pnoise)”</a> on page 48
<i>Noise Type</i> selects the type of noise to compute. (Active only when PSS analysis is not swept.)	<a href="#">“Noise Type (Pnoise)”</a> on page 59
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

Field or Pane	User Interface Help (page)
---------------	----------------------------

<i>Options</i> displays the Options form for this analysis.	“ <a href="#">Options</a> ” on page 62
---	--

### **Modifications to Pnoise Form for Swept PSS analysis**

[Sweeptype](#), [Frequency Sweep Range](#), [Single-Point](#), and [Freq](#) specify the frequency for the small-signal analysis that follows each swept PSS analysis.

([Noise Type](#) is not active when PSS analysis is swept.)

### **Periodic Transfer Function (PXF) Choosing Analyses Form**

Field or Pane	User Interface Help (page)
---------------	----------------------------

<i>Analysis</i> selects the type of analysis to set up.	“ <a href="#">Analysis</a> ” on page 34
---	---

<i>PSS Beat Frequency (Hz)</i> displays the <i>Beat Frequency</i> for the associated PSS analysis.	“ <a href="#">PSS Beat Frequency (PAC, PSTB, Pnoise, and PXF)</a> ” on page 72
--	--

<i>Sweeptype</i> , <i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	“ <a href="#">Sweeptype (PAC and PXF)</a> ” on page 101, “ <a href="#">Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)</a> ” on page 38
--	---

<i>Sidebands</i> selects the set of periodic small-signal output frequencies of interest.	“ <a href="#">Sidebands (PAC, Pnoise, and PXF)</a> ” on page 78
---	---

<i>Output</i> selects the output.	“ <a href="#">Output (PXF and QPXF)</a> ” on page 64
-----------------------------------	--

<i>Modulated Analysis (PAC, PXF)</i> — One of the Specialized Analyses, specifies Modulated analysis.	<b>“<a href="#">Modulated Analysis (PAC, PXF) — One of the Specialized Analyses</a>”</b> on page 56
---	---

<i>Enabled</i> includes this analysis in the next simulation.	“ <a href="#">Enabled</a> ” on page 37
---	--

<i>Options</i> displays the Options form for this analysis.	“ <a href="#">Options</a> ” on page 62
---	--



# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

---

### **Modifications to PXF Form for Swept PSS Analysis**

Sweeptype, Frequency Sweep Range, Single-Point, and Freq specify the frequency for the small-signal analysis that follows each swept PSS analysis.

### **Periodic S-Parameter (PSP) Choosing Analyses Form**

---

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Sweeptype</i> , <i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweeptype (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38
<i>Select Ports</i> selects the active ports for the PSP analysis.	<a href="#">“Select Ports (PSP and QPSP)”</a> on page 73
<i>Do Noise</i> selects whether or not to measure noise during the PSP analysis.	<a href="#">“Do Noise (PSP and QPSP)”</a> on page 37
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

---

### **Modifications to PSP Form for Swept PSS Analysis**

Sweeptype, Frequency Sweep Range, Single-Point, and Freq specify the frequency for the small-signal analysis that follows each swept PSS analysis.

### **Quasi-Periodic Noise (QPnoise) Choosing Analyses Form**

---

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweeptype (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38

---

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Spectre RF Simulation Form Reference

---

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Sidebands</i> selects the set of periodic small-signal output frequencies of interest.	<a href="#">“Sidebands (QPAC, QPnoise, and QPXF)”</a> on page 82
<i>Output</i> , <i>Input Source</i> , <i>Reference Side-Band</i> , and <i>refsidebandoption</i> selects the output, noise generator, reference sidebands, and <i>refsidebandoption</i> for the QPnoise analysis. (Displays additional fields.)	<a href="#">“Output (Pnoise and QPnoise)”</a> on page 65, <a href="#">“Input Source and Reference Side-Band (QPnoise)”</a> on page 52
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

**Modifications to QPnoise Form for Swept QPSS Analysis**

[Frequency Sweep Range](#), [Single-Point](#), and [Freq](#) specify the frequency for the small-signal analysis that follows each swept QPSS analysis.

**Quasi-Periodic AC (QPAC) Choosing Analyses Form**

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweeptype (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38
<i>Sidebands</i> selects the set of periodic small-signal output frequencies of interest.	<a href="#">“Sidebands (QPAC, QPnoise, and QPXF)”</a> on page 82
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

### **Modifications to QPAC Form for Swept QPSS Analysis**

Frequency Sweep Range, Single-Point, and Freq specify the frequency for the small-signal analysis that follows each swept QPSS analysis.

### **Quasi-Periodic Transfer Function (QPXF) Choosing Analyses Form**

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweep type (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38
<i>Sidebands</i> selects the set of periodic small-signal output frequencies of interest.	<a href="#">“Sidebands (QPAC, QPnoise, and QPXF)”</a> on page 82
<i>Output</i> selects the output.	<a href="#">“Output (PXF and QPXF)”</a> on page 64
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

### **Modifications to QPXF Form for Swept QPSS Analysis**

Frequency Sweep Range, Single-Point, and Freq specify the frequency for the small-signal analysis that follows each swept QPSS analysis.

### **Quasi-Periodic S-Parameter (QPSP) Choosing Analyses Form**

<b>Field or Pane</b>	<b>User Interface Help (page)</b>
<i>Analysis</i> selects the type of analysis to set up.	<a href="#">“Analysis”</a> on page 34
<i>Sweep type</i> , <i>Frequency Sweep Range</i> , <i>Sweep Type</i> , and <i>Add Specific Points</i> set up the sweep for the small-signal analysis.	<a href="#">“Sweep type (PAC and PXF)”</a> on page 101, <a href="#">“Frequency Sweep Range, Sweep Type, Add Specific Points (Small-Signal)”</a> on page 38

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

---

Field or Pane	User Interface Help (page)
<i>Select Ports</i> selects the active ports for the PSP analysis.	<a href="#">“Select Ports (PSP and QPSP)”</a> on page 73
<i>Do Noise</i> selects whether or not to measure noise during the PSP analysis.	<a href="#">“Do Noise (PSP and QPSP)”</a> on page 37
<i>Enabled</i> includes this analysis in the next simulation.	<a href="#">“Enabled”</a> on page 37
<i>Options</i> displays the Options form for this analysis.	<a href="#">“Options”</a> on page 62

---

### **Modifications to QPSP Form for Swept QPSS Analysis**

[Sweeptype](#), [Frequency Sweep Range](#), [Single-Point](#), and [Freq](#) specify the frequency for the small-signal analysis that follows each swept QPSS analysis.

## **Option Forms**

### **PSS Analysis Options Form**

[Time Step Parameters](#) define the time step used for the PSS analysis.

[Initial Condition Parameters](#) define the initial conditions for the PSS analysis.

[Convergence Parameters](#) provide an initial transient solution and minimum capacitance for the PSS analysis.

[State File Parameters](#) provides the locations of files associated with the PSS analysis.

[Integration Method Parameters](#) define the integration method used for the PSS analysis.

[Accuracy Parameters](#) define the level of accuracy to use for the PSS analysis. Enable/disable/refine use of the Finite difference method of the PSS analysis.

[Annotation Parameters](#) define statistics and other information recorded and displayed for the PSS analysis.

[Output Parameters](#) defines information related to the results of the PSS analysis.

[Newton Parameters](#) defines information about the Newton iterations and previous DC solution for the PSS analysis.

Simulation Interval Parameters define the start time for the initial transient analysis for this PSS analysis.

### **QPSS Analysis Options Form**

Time Step Parameters define the time step used for the QPSS analysis.

Initial Condition Parameters define the initial conditions for the QPSS analysis.

Convergence Parameters provide an initial transient solution and minimum capacitance for the QPSS analysis.

Multitone Stabilization Parameter (QPSS) specifies the number of stabilization cycles to be performed.

State File Parameters provides the locations of files associated with the QPSS analysis.

Integration Method Parameters define the integration method used for the QPSS analysis.

Accuracy Parameters define the level of accuracy to use for the QPSS analysis.

Annotation Parameters define statistics and other information recorded and displayed for the QPSS analysis.

Output Parameters defines information related to the results of the QPSS analysis.

Newton Parameters defines information about the Newton iterations and previous DC solution for the QPSS analysis.

Simulation Interval Parameters define the start time for the initial transient analysis for this QPSS analysis.

### **Envelope Following Analysis Options Form**

Simulation Interval Parameters defines the start time, output start time, and the stabilization time period for the ENVLP analysis.

Simulation Bandwidth Parameters define the modulation bandwidth for the ENVLP analysis.

Time Step Parameters define the time step and the outer envelope size used for the ENVLP analysis.

Initial Condition Parameters define the initial conditions for the ENVLP analysis.

Convergence Parameters provide an initial transient solution and minimum capacitance for the ENVLP analysis.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Spectre RF Simulation Form Reference

---

State File Parameters provides the locations of files associated with the ENVLP analysis.

Integration Method Parameters define the integration method used for the ENVLP analysis.

Accuracy Parameters define the level of accuracy to use for the ENVLP analysis.

Annotation Parameters define statistics and other information recorded and displayed for the ENVLP analysis.

Output Parameters defines information related to the results of the ENVLP analysis.

Newton Parameters defines information about the Newton iterations and previous DC solution for the ENVLP analysis.

### Periodic Small-Signal Analyses Options Forms

Convergence Parameters provide convergence information for the small-signal analysis.

Annotation Parameters define statistics and other information recorded and displayed for the small-signal analysis.

Output Parameters defines information related to the results of the small-signal analysis.

### Quasi-Periodic Small-Signal Analyses Options Forms

Convergence Parameters provide convergence information for the small-signal analysis.

Annotation Parameters define statistics and other information recorded and displayed for the small-signal analysis.

Output Parameters defines information related to the results of the small-signal analysis.

### Direct Plot Form

See Direct Plot Form for information on using the Direct Plot Form.

See Field Descriptions for the Direct Plot Form for descriptions of fields on the Direct Plot Form.

### ACPR Wizard

Clock Name Specifies the clock signal for the ENVLP analysis

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Spectre RF Simulation Form Reference

---

How to Measure Selects whether to measure ACPR for a single net or between two differential nets.

Channel Definitions Select one of three channel definitions: Custom, IS-95 or W-CDMA.

Main Channel Width Specify channel width in Hz.

Adjacent frequencies list box Specifies the adjacent channel frequencies. Use the Add, Change and Delete buttons and the editing fields to modify adjacent frequencies.

Simulation Control Enter the Stabilization Time and Repetitions in the adjacent fields.

Resolution Bandwidth (Hz) and Calculate Click *Calculate* to determine the Resolution Bandwidth.

Windowing Function Selects the windowing function to use.

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Spectre RF Simulation Form Reference

---



---

## Setting Up for the Examples

---

This chapter explains how to set up your software and environment to run the examples in this user guide. This chapter describes the procedure for accessing the latest version of Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF), which ships in the MMSIM release stream.

Before you perform the Spectre RF analyses, you need to set up the component files and start the Cadence<sup>®</sup> software.

### Setting Up Environment Variables and the Path Statement

Set the following environment variables for Spectre RF.

```
setenv CDS_rfExamples `cds_root icms`
```

```
setenv CDS_Netlisting_Mode Analog
```

`CDS_rfExamples` defines the path to the piece-wise linear (PWL) model files in the Cadence software installation hierarchy.

### Using Spectre RF from the MMSIM Hierarchy

Starting with the 5.1.41 USR1 release, you have the option to obtain the Virtuoso Spectre Circuit Simulator (Spectre), Spectre RF, and other simulators, from the MMSIM release stream. While the version of Spectre and Spectre RF that shipped with 5.1.41 USR1 continue to ship with the remaining 5.1.41 releases, documentation for new features and most bug fixes is provided exclusively with the MMSIM release stream. The first MMSIM release, MMSIM6.0, was released at the same time as the 5.1.41 USR1 update.

You must download and install the MMSIM simulators in a separate simulation installation hierarchy than the hierarchy you use for the Cadence software.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

The Spectre RF examples use the following environment variables to point to the two installation hierarchies,

MMSIMHOME	Path to the installation hierarchy for the MMSIM simulators
CDSHOME	Path to the installation hierarchy for the Cadence software

To use the MMSIM simulators, put all paths to the MMSIM simulators such as

```
<path_to_MMSIM_simulators>/tools/bin
```

before any paths to the dfll software in your \$PATH statement.

The Spectre RF examples use the following path

```
path = ( $MMSIMHOME/tools/bin \  
$CDSHOME/tools/bin \  
$CDSHOME/tools/dfII/bin \  
$CDSHOME/tools/java \  
$CDSHOME/tools/java/bin \  
$path )
```

## Accessing the Most Current Spectre RF Documentation

The documentation for the latest features of Spectre RF is always found in the MMSIM hierarchy. If you are using the MMSIM version of Spectre RF, access the Spectre RF documentation from the MMSIM hierarchy.



Note that the help buttons on the forms lead you to the 5.1.41 version of the documentation.

## Creating a Local Editable Copy of the rfExamples Library

Make a copy of the *rfExamples* library and save it in a directory in your account. Change the name and access mode of your local copy so you have write access to your local copy of the library. This permits you to edit the schematics and other files in this library as you follow the examples.

The *rfExamples* library is located at

```
<CDSHOME>/tools/dfII/samples/artist/rfExamples
```

where CDSHOME is the installation directory for your Cadence software.

For the examples in this book, the local copy of the *rfExamples* library is defined as

```
DEFINE my_rfExamples /home/belinda/my_libs/rfExamples
```

## Setting Up the Cadence Libraries

The Cadence Libraries are defined in the UNIX text file *cds.lib*. You can edit this file by using the library path editor or by using a UNIX shell window.

### Using the Library Path Editor

To access the Library Path Editor, use the following procedure.

1. In a UNIX window, type `icms &` to start the Cadence software.

The Command Interpreter Window (CIW) appears.



2. In the CIW, choose *Tools – Library Path Editor*.

The Library Path Editor appears.

## Library Path Editor Window



In the Library Path Editor, follow the instructions at the bottom of the form.

3. Type a name for each required library and the associated path to the library in the software installation hierarchy. You need the libraries listed in [“Library Path Editor Window”](#) on page 180.
4. Use *File – Save* to save your definitions in the `cds.lib` file.
5. When you are using Open Access (OA), in addition to saving the `cds.lib` file an OA `lib.defs` file is also saved.
6. Exit the Library Path Editor.

## Using a UNIX Shell Window

To set up the libraries in a UNIX shell window, use the following procedure:

1. In a UNIX shell window, open the `cds.lib` file for editing using *vi*, *emacs* or a similar text editor.

The `cds.lib` file is in your installation directory.

2. In the `cds.lib` file, define the Cadence provided libraries.
3. Define a user library where the sample circuits can be tested.

You can label your test library any name you choose. This example assumes that you have called the library *my\_rfExamples*. The name *my\_dir* represents the directory into which you copied the *rfExamples* library.

You must use the names *basic*, *sample* and *analogLib*. You cannot rename these libraries.

After these steps, the definitions in the `cds.lib` file look similar to the following:

```
DEFINE rfExamples $CDSHOME/tools/dfII/samples/artist/rfExamples
DEFINE analogLib $CDSHOME/tools/dfII/etc/cdslib/artist/analogLib
DEFINE ahdlLib $CDSHOME/tools/dfII/samples/artist/ahdlLib
DEFINE rfLib $CDSHOME/tools/dfII/samples/artist/rfLib
DEFINE my_rfExamples /home/belinda/my_libs/rfExamples
DEFINE sample $CDSHOME/tools/dfII/samples/cdslib/sample
DEFINE basic $CDSHOME/tools/dfII/etc/cdslib/basic
DEFINE passiveLib $CDSHOME/tools/dfII/samples/artist/passiveLib
DEFINE pllLib $CDSHOME/tools/dfII/samples/artist/pllLib
```

## Setting Up For Simulation

### Opening a Circuit in the Schematic Window

To open a circuit in the schematic window,

1. In the CIW, choose *File – Open*.

The Open File form appears.

2. In the Open File form, choose *my\_rfExamples* in the *Library Name* menu.

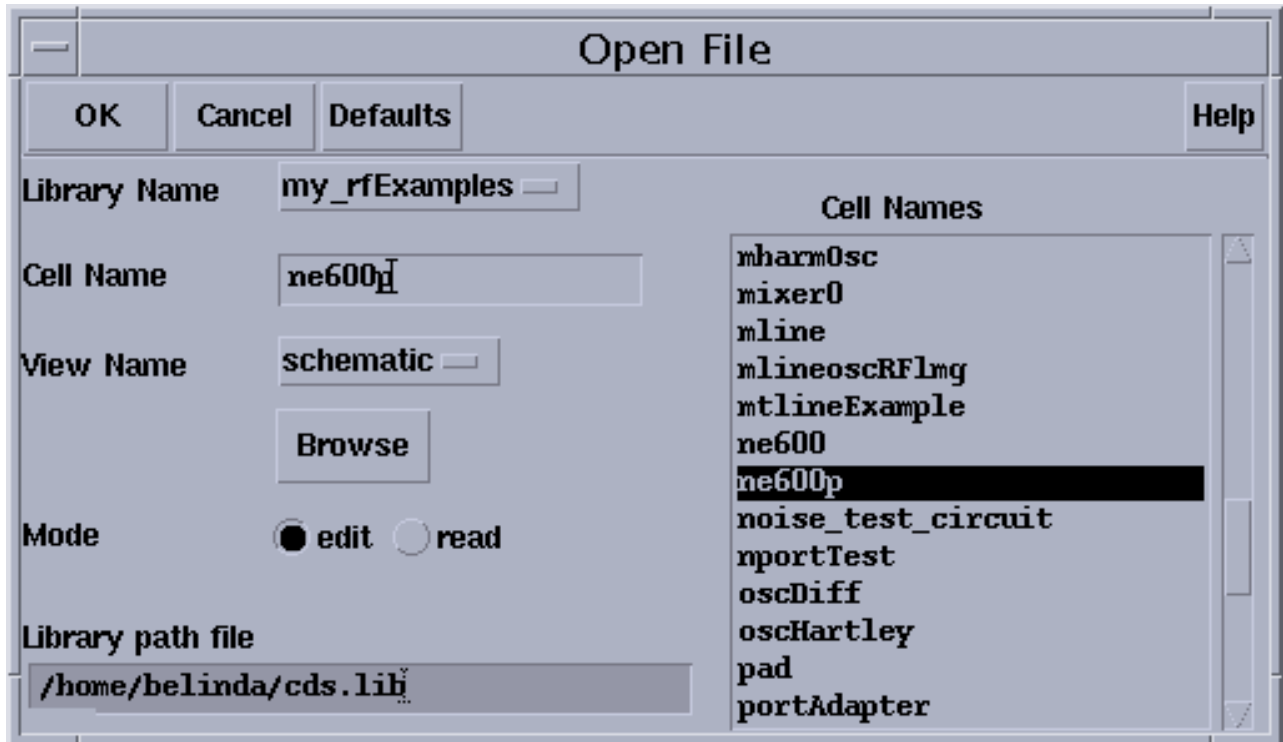
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

3. Choose the circuit you want to work with from the list in the *Cell Names* list box. the *ne600p* circuit is used in this example.

The completed Open File form appears like the one below.



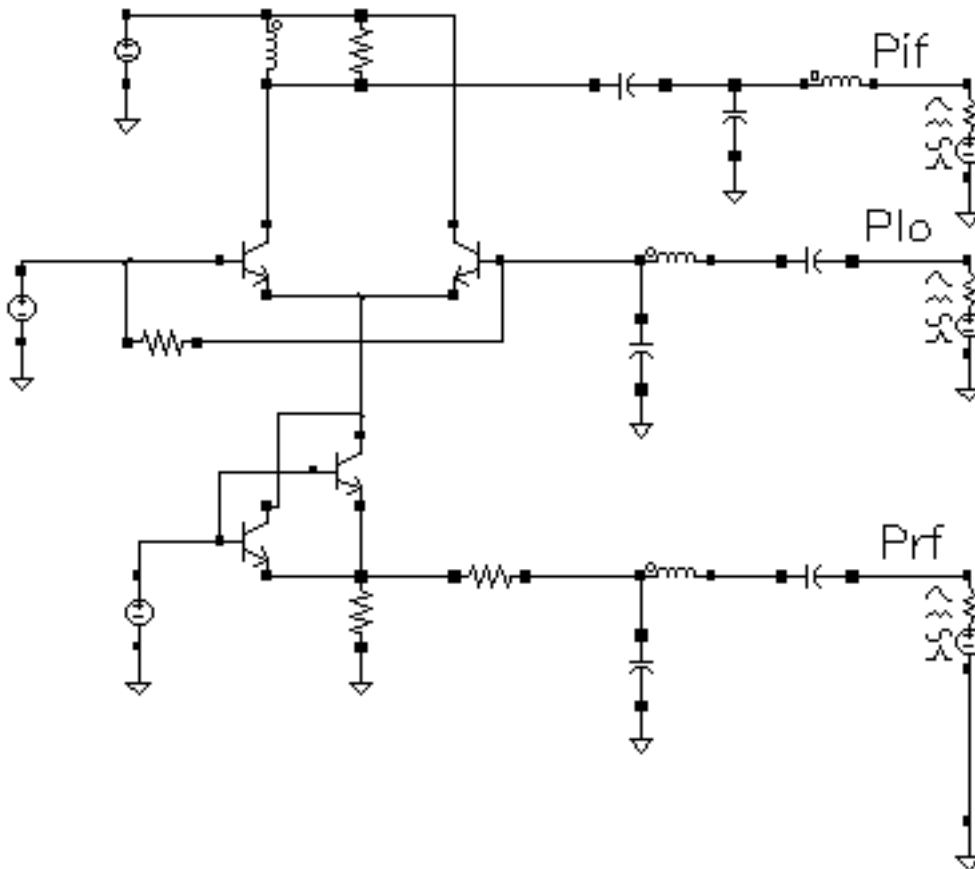
4. Click OK.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

The Schematic window opens to display the circuit. In this case, the *ne600p* mixer appears.



### Opening the Simulator Window

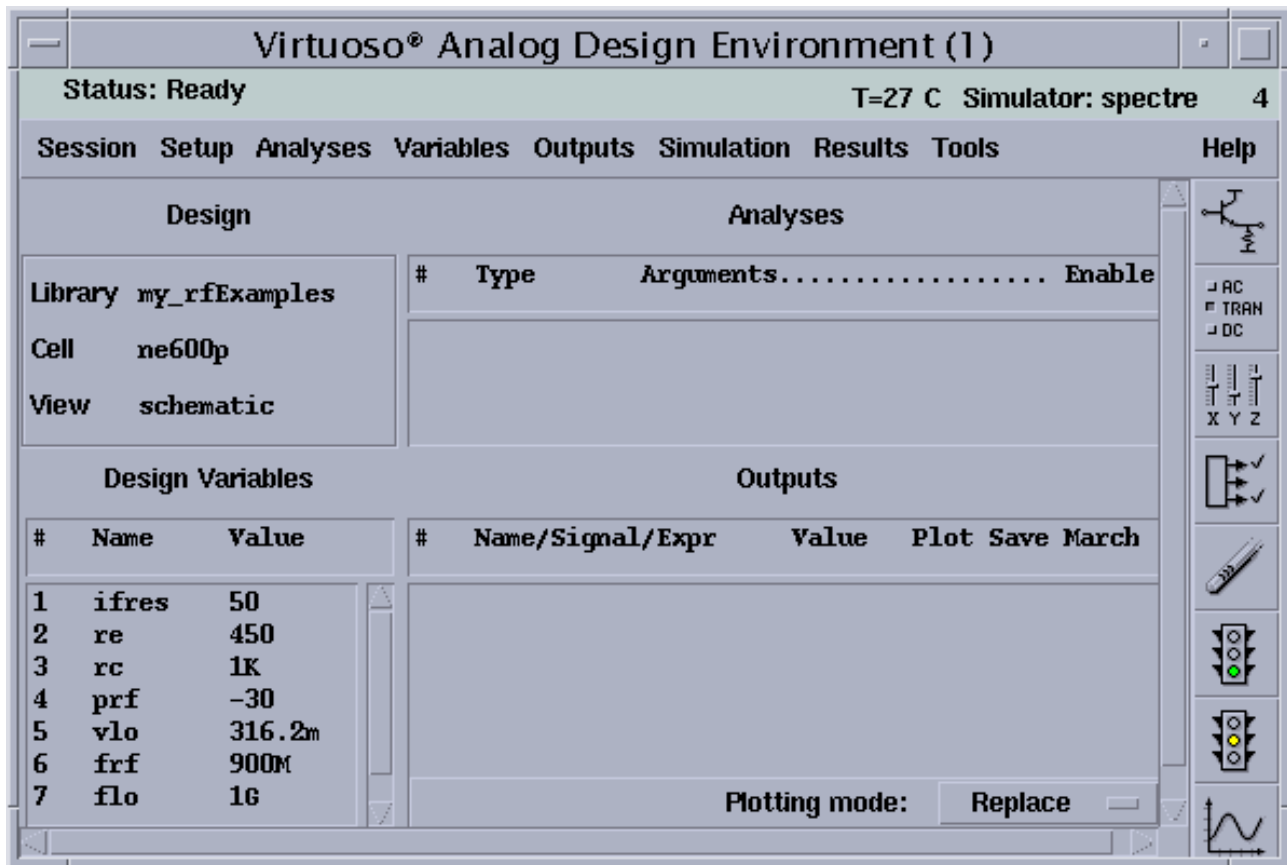
To open the Simulator window,

1. In the Schematic window, choose *Tools– Analog Environment*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

The Simulation window opens. This window is also called the Cadence® Analog Circuit Design Environment.



**Note:** You can also use *Tools – Analog Environment – Simulation* in the CIW to open the Simulation window without opening the design. You can open the design later by choosing *Setup – Design* in the Simulation window and then choosing *ne600p* in the Choosing Design form.

## Choosing Simulator Options

To set up the simulator options,

1. Choose *Setup – Simulator/Directory/Host* in the Simulation window.  
The Simulator/Directory/Host form appears.
2. In the Simulator/Directory/Host form, specify the following:
  - a. Choose *spectre* for the *Simulator*.



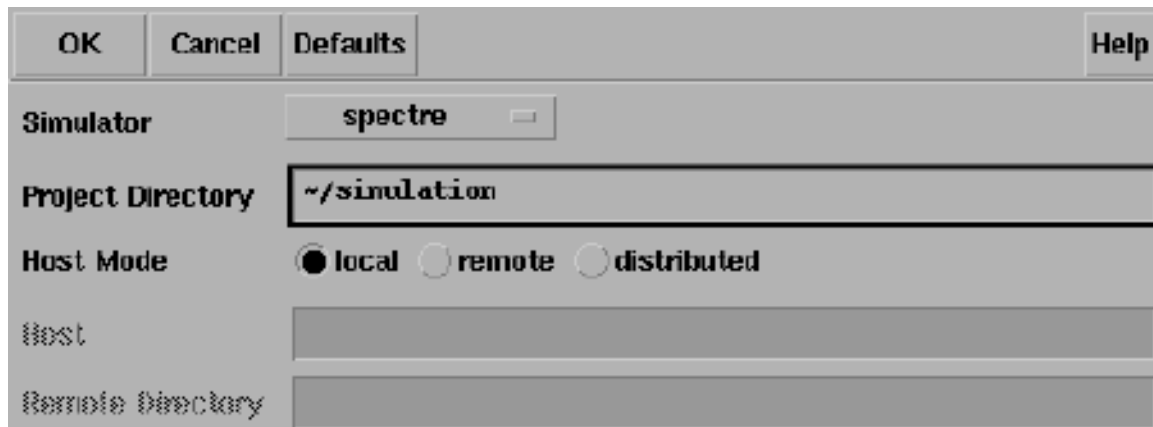
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

- b. Type the name of the project directory, if necessary.
- c. Highlight the *local* or the *remote* button to specify the *Host Mode*.
- d. (Optional) For remote simulation, type the name of the host machine and the remote directory in the appropriate fields.

The completed form appears like the one below.



The screenshot shows a dialog box with the following fields and controls:

- Buttons: OK, Cancel, Defaults, Help
- Simulator: spectre
- Project Directory: ~/simulation
- Host Mode: local (selected), remote, distributed
- Host: (empty text field)
- Remote Directory: (empty text field)

3. In the Simulator/Directory/Host form, click *OK*.

### Specifying Outputs to Save

To specify the outputs that you want to save,

1. In the Simulation window, choose *Outputs – Save All*.

The Save Options form appears.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

2. In the *Select signals to output section*, be sure *allpub* is highlighted.

OK Cancel Defaults Apply Help

Select signals to output (save)  none  selected  lvpub  lv  allpub  all

Select power signals to output (pwr)  none  total  devices  subckts  all

Set level of subcircuit to output (nestlvl)

Select device currents (currents)  selected  nonlinear  all

Set subcircuit probe level (subcktprobe/vl)

Select AC terminal currents (useprobes)  yes  no

Select AHDL variables (saveahdlvars)  selected  all

Save model parameters info

Save elements info

Save output parameters info

## Setting Up Model Libraries

To set up the model libraries,

1. In the Simulation window, choose *Setup – Model Libraries*.

The Model Library Setup form appears.

2. In the *Model Library File* field, type the full path to the model file including the file name, `<CDSHOME>/tools/dfII/samples/artist/models/spectre/rfModels.scs`.

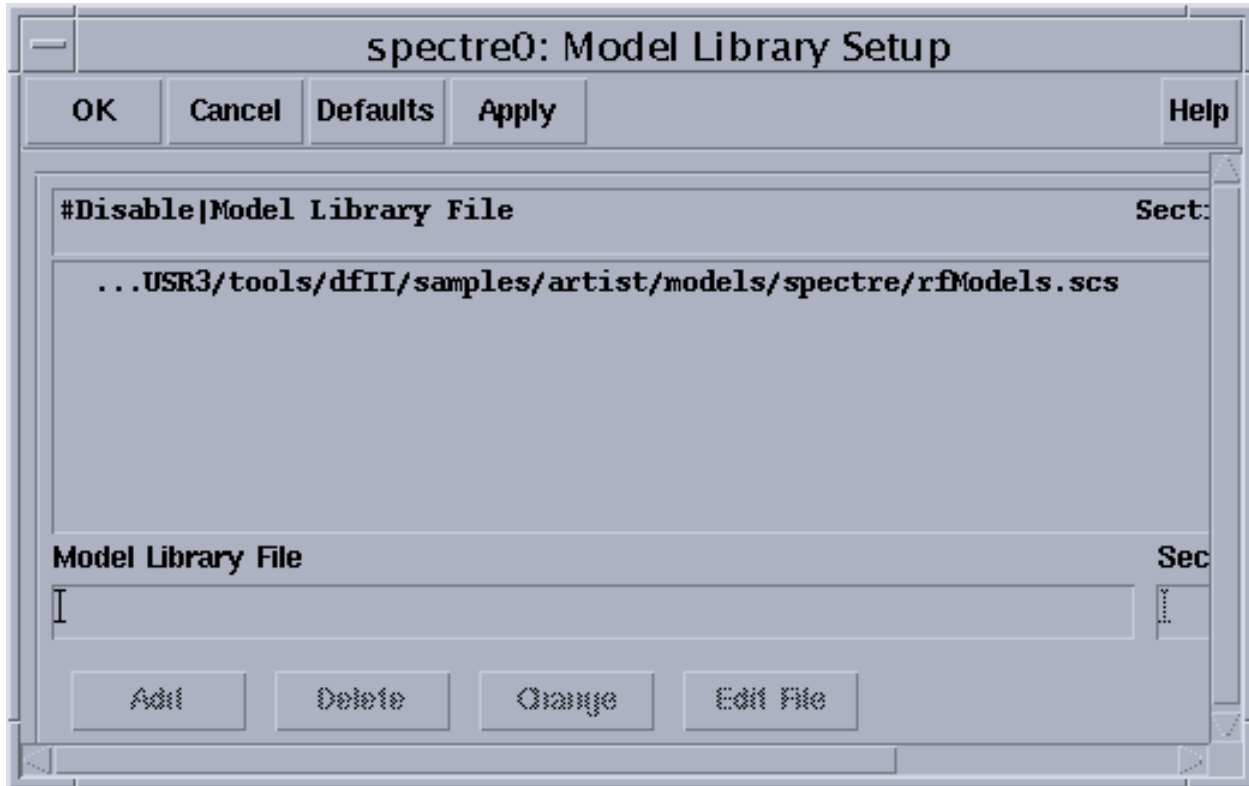
When you type the path to the model library, you must expand `<CDSHOME>` and type in the actual path to your software installation hierarchy.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

The completed form appears like the one below.



3. In the Model Library Setup form, click *Add*.
4. Click *OK*.

## Editing Design Variable Values

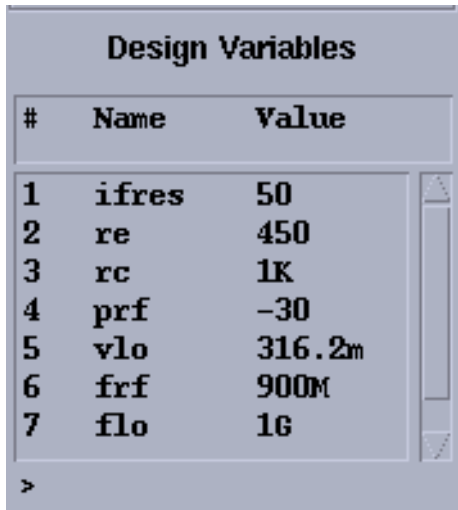
To edit the design variable values,

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

1. Verify the current variables and their values in the Design Variables area in the Simulation window.



#	Name	Value
1	ifres	50
2	re	450
3	rc	1K
4	prf	-30
5	vlo	316.2m
6	frf	900M
7	flo	1G

2. In the Simulation window, use the following procedure to set the design variables to the values required for the simulation.

(See the description of each simulation for the required variable names and values.) This example changes the value of the variable `frf`.

- a. In the Simulation window, choose *Variables – Edit*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

The Editing Design Variables form appears.

Selected Variable		Table of Design Variables		
Name	Value (Expr)	#	Name	Value
		1	ifres	50
		2	re	450
		3	rc	1K
		4	prf	-10
		5	vlo	316.2m
		6	frf	920M
		7	flo	1G

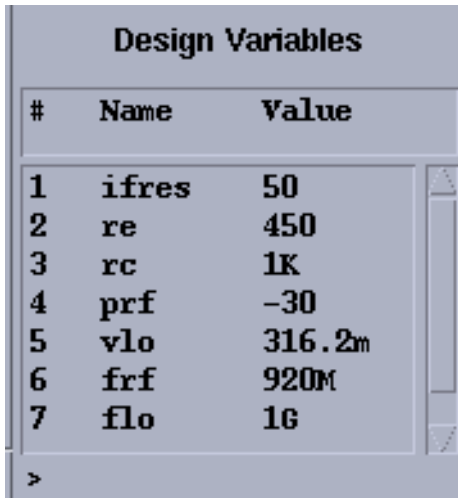
- b. In the *Table of Design Variables*, select a variable to edit.
- c. The variable name and value display in the *Name* and *Value (Expr)* fields.
- d. In the *Value (Expr)* field, type 900M (or 920M) for the value of `frf` and click *Change*.
- e. Edit the variable's value in the *Value (Expr)* field. Then click *Change*.
- f. The variable value changes in the *Table of Design Variables*.
- g. Repeat to edit more variables.
- h. In the Editing Design Variables form, click *OK*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Setting Up for the Examples

---

- i. View the new variable value in the *Design Variables* section of the Simulation window.



#	Name	Value
1	ifres	50
2	re	450
3	rc	1K
4	prf	-30
5	vlo	316.2m
6	frf	920M
7	flo	1G

---

## Modeling Transmission Lines

---

The Transmission Line Model Generator (LMG) models transmission lines in two ways, by

- Loading quadrature model subcircuits generated by LMG
- Using the LRCG matrixes, also produced by LMG, in *mtline* multi-conductor transmission line model files

LMG supports microstrip, stripline, multi-layer, and multi-transmission line systems that include substrate loss. When you use LMG with the new *mtline* external transmission line model, LMG supports the various interconnects used in IC and PCB designs.

Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) now supports a built-in 2-D field solver in its *MTLINE* transmission line model.

### The Transmission Line Models: *tline3*, *mline* and *mtline*

There are several ways to use the *tline3*, *mline*, and *mtline* external transmission line model files in the analog design environment.

#### The *tline3* Model

The *tline3* model is a 3-terminal, single-conductor transmission line model. The *tline3* model can be either a stripline or a microstrip line. The *tline3* model can either use an existing subcircuit produced by LMG or it can invoke LMG in the background during netlisting.

For details on the *tline3* model, reference the following examples.

[“Creating a \*tline3\* Macromodel in the LMG GUI”](#) on page 195.

[“Using an Existing \*tline3\* Macromodel in the Schematic Flow”](#) on page 205.

[“Creating a \*tline3\* Macromodel in the Schematic Flow”](#) on page 213.

## The *m*line Model

The *m*line model is a multi-terminal, multi-conductor transmission line model. The *m*line model can be either a stripline or a microstrip line. It can also handle multi-conductor systems. For  $n$  signal lines, the *m*line is a  $2n+1$  terminal device. It can use an existing subcircuit produced by LMG or it can invoke LMG in the background during netlisting.

For details on the *m*line model, reference the following examples.

[“Creating an \*m\*line Transmission Line Model Starting LMG From UNIX”](#) on page 218.

[“Using an \*m\*line Macromodel in the Schematic Flow”](#) on page 223

[“Resolving a Possible Error Message for an \*m\*line Model File”](#) on page 228

[“Creating an \*m\*line Transmission Line in the Schematic Flow”](#) on page 230.

## The *mt*line Model

The *mt*line model is a multi-layer, multi-conductor transmission line model. The *mt*line model can be a stripline, a microstrip line, a substrate loss interconnect, or a coplanar waveguide. The *mt*line model handles both multi-layer and multi-conductor transmission line systems. For  $n$  signal lines, the *mt*line is a  $2n+2$  terminal device; the ground plane takes 2 terminals. While the *mt*line can use the existing subcircuit produced by LMG, it usually uses an LRCG file which is also produced by LMG. From the analog design environment, an *mt*line model also can invoke the LMG GUI from its CDF.

For details on the *mt*line model, reference the following examples.

[“Looking at \*mt\*line in the Analog Design Environment”](#) on page 234.

[“Coplanar Waveguide Modeling and Analysis”](#) on page 238

## LMG Use Models

The Transmission Line Model Generator (LMG) has two use models. This chapter illustrates by example procedures for using models with various transmission line instances. You can

- Use the LMG GUI, independent of the analog design environment
- Model transmission lines from within the analog design environment by entering parameter values



## Modeling Transmission Lines Using the LMG GUI

To use the LMG GUI standalone, you first open the LMG GUI in any one of several ways

- From the UNIX command line, by entering `LMG &` at the UNIX prompt.
- By choosing *Tools—RF—Transmission Line Generator* in the Simulation window.
- For the *mtline* model only, you can open LMG from the *mtline* model's CDF.
- From the LMG GUI, you manually enter data for each unique transmission line configuration in your design. You manually create and save separate and unique model files for each transmission line segment where there is a unique combination of line type, geometry, parameters, or operating frequency.
- This use model builds a 2-D cross-section of the transmission line as you enter parameter values in the graphical window at the top of the LMG GUI as you create the transmission line. Whenever you modify a parameter value, the 2-D cross-section updates to reflect your change.

Begin by entering line-parameter information, the length of the transmission line, and the operating frequencies for the transmission line segment in the LMG GUI. LMG eventually creates a lumped-circuit macromodel for the transmission line segment. You continue by creating lumped-circuit macromodels for each unique transmission line segment in the circuit.

You can share and reuse models that you create and save manually. When you run LMG manually, you create a model file that you can apply to multiple, transmission line instances with identical characteristics. If you have a circuit with a number of identical transmission line instances, you can reuse the same model for each of these instances. You might also want to run LMG manually so you can display a cross-section of the transmission line.

You also enter unique names for the per-unit length LRCG matrix file and the Spectre netlist file that describe the subcircuit. It is helpful to use names that describe the configuration of the particular transmission line. For example, for a stripline with length 1000 and with a maximum frequency of 20GHz you might use the name `tline_s1000_20G`.

Using your input specifications, LMG automatically extracts parameters and generates a quadrature model subcircuit for each transmission line segment. The quadrature model subcircuits are only accurate for the length and frequency that you specify.

If you modify a transmission line, rerun LMG for the new configuration and update the appropriate instances in the schematic to reflect the new LMG output.

Before you simulate the circuit, you incorporate each model file into the design by specifying its filename as a property on a corresponding *tline3*, *mline*, and *mtline* instance. For *mtline* models, you can also specify an LRCG file.

For examples, see the following sections.

[“Creating a tline3 Macromodel in the LMG GUI”](#) on page 195.

[“Creating an mline Transmission Line Model Starting LMG From UNIX”](#) on page 218.

[“Looking at mtline in the Analog Design Environment”](#) on page 234.

## **Modeling Transmission Lines Without Using the LMG GUI**

This use model is easier to use than the first use model. In this use model, LMG runs within the analog design environment. The software creates every transmission line model instance separately.

This use model includes the following general steps:

1. Open a schematic.
2. Place a *tline3*, *mline*, or *mtline* transmission line instance in the Schematic window.
3. Choose *Edit—Properties—Objects* and select the transmission line instance you placed in the schematic.
4. Describe the transmission line instance by entering parameter values in the Edit Properties form fields. (Units are meters and Hertz only.)
5. For each *tline3*, *mline*, or *mtline* transmission line instance in the design, repeat steps 2, 3, and 4.

For examples, see the following sections.

[“Using an Existing tline3 Macromodel in the Schematic Flow”](#) on page 205.

[“Creating a tline3 Macromodel in the Schematic Flow”](#) on page 213.

[“Using an mline Macromodel in the Schematic Flow”](#) on page 223.

[“Creating an mline Transmission Line in the Schematic Flow”](#) on page 230.

## **Default Values and the initlmg File**

When you start LMG, the `./initlmg` file is checked for default values. If the `./initlmg` file does not exist, then LMG uses the following default values:

```
lmgLineType = Microstrip
```

```
lmgLengthUnit = um
lmgFreqUnit = GHz
lmgNumberOfConductors = 1
lmgDielectricPermittivity = 10
lmgDielectricThickness = 400
lmgConductorWidth = 300
lmgConductorDistances = For multiconductors, the width of the first conductor
lmgConductorThickness = 100
lmgConductorHeight = For multiconductors, the height of the first conductor
lmgConductorLength = 5000
lmgConductivity = 5.6e7
lmgMaxFreq = 10
lmgSubCircuitName = tline
```

## Tline3 Transmission Line Modeling Examples

This section presents a series of examples showing how to create a *tline3* transmission line model using the LMG GUI. Because the procedures for creating microstrip and stripline transmission lines are very similar, only a stripline example is shown here.

### Creating a tline3 Macromodel in the LMG GUI

This example illustrates how to create a *tline3* macromodel from the LMG GUI while you are using the analog design environment.

This example uses the circuit `tline3oscRF1mg` from your editable copy of the `rfExamples` library (*my\_rfExamples* here). You need to open this circuit to set up the analog design environment. The *my\_rfExamples* library also includes other sample circuits used in this manual. If you need assistance setting up or accessing your editable copy of the `rfExamples` library, refer to [Chapter 3, “Setting Up for the Examples.”](#)

### Opening the tline3oscRF1mg Circuit in the Schematic Window

1. In the CIW, choose *File – Open*.

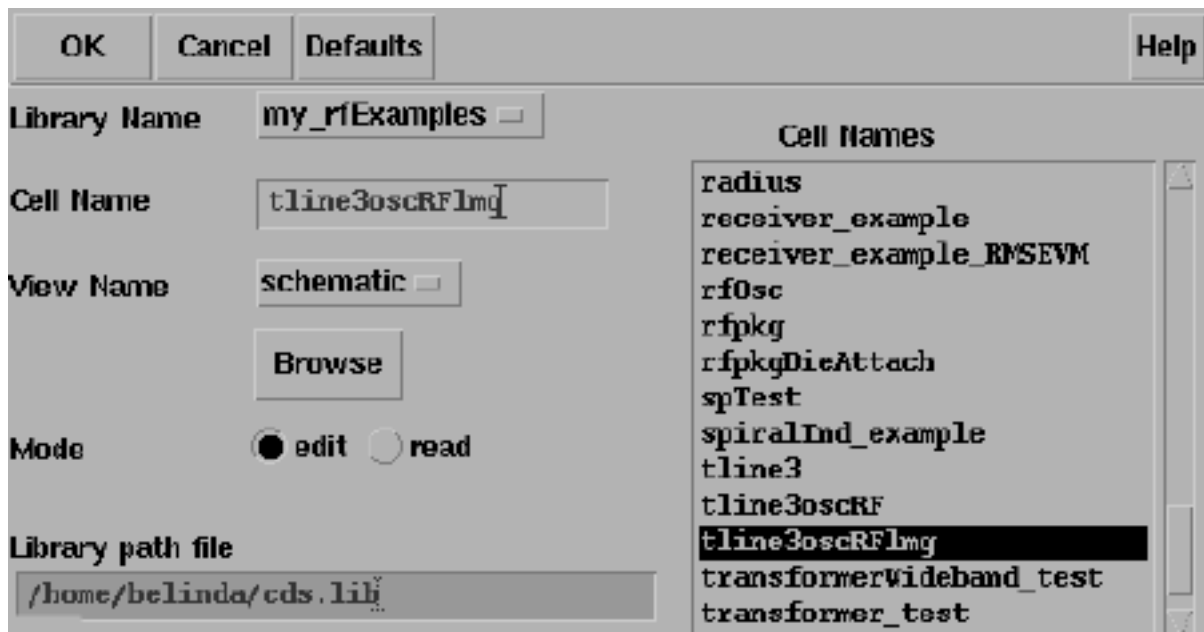
The Open File form appears.

2. In the Open File form, choose *my\_rfExamples* in the *Library Name* cyclic field and choose *tline3oscRF1mg* in the *Cell Names* list box.

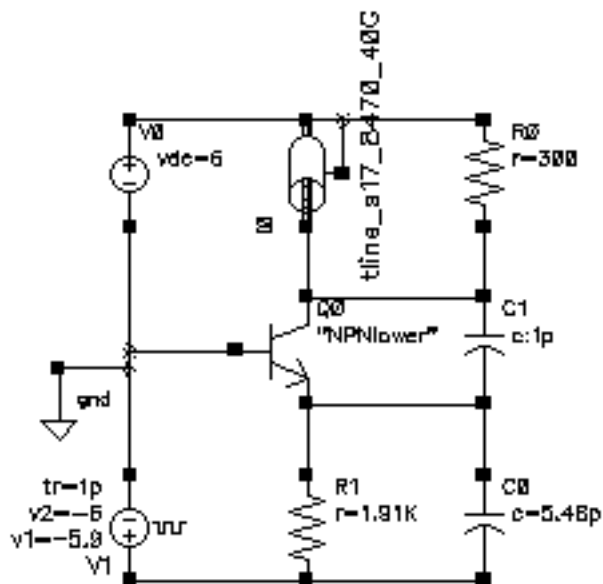
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The completed Open File form appears like the one below.

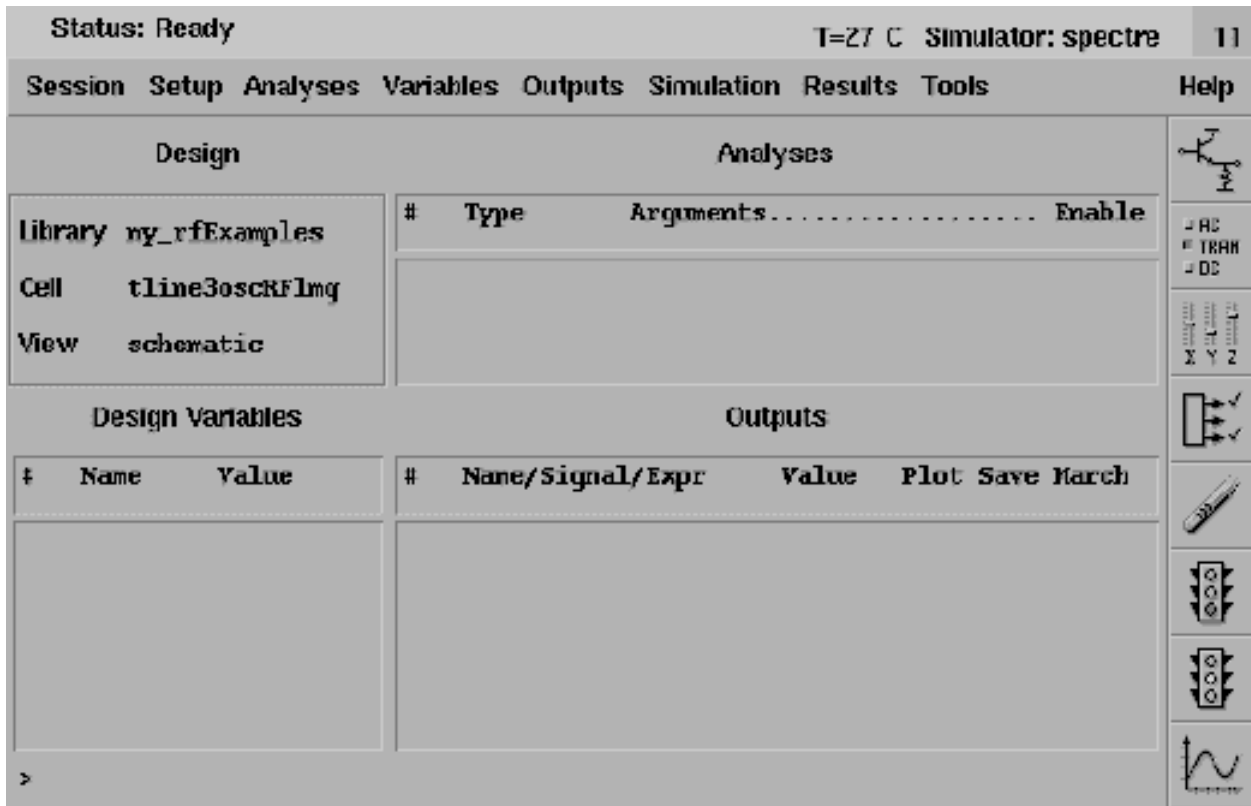


3. Click **OK**.
4. The Schematic window for the `tline3oscRF1mg` circuit appears.



5. In the Schematic window, choose *Tools—Analog Environment*.

6. The Simulation window opens.



### Opening the LMG GUI

This example assumes that this is the first run of LMG and that the file `./initlmg` does not yet exist. Consequently, LMG uses internal default values.

When you start LMG by typing `lmg &` at the UNIX prompt, the LMG GUI displays directly without first displaying the Transmission Line Modeler form and the path to the model directory as shown here in this example.

To model a `tline3` transmission line using the LMG GUI, perform the following steps:

1. Choose *Tools–RF–Transmission Line Modeler* in the Simulation window.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The Transmission Line Modeler form appears displaying the path to the directory where the model is created.



**Note:** Make a note of the location of this model as you need to enter the absolute path to this macromodel in another example, [“Using an Existing tline3 Macromodel in the Schematic Flow”](#) on page 205.

The path is similar to the following:

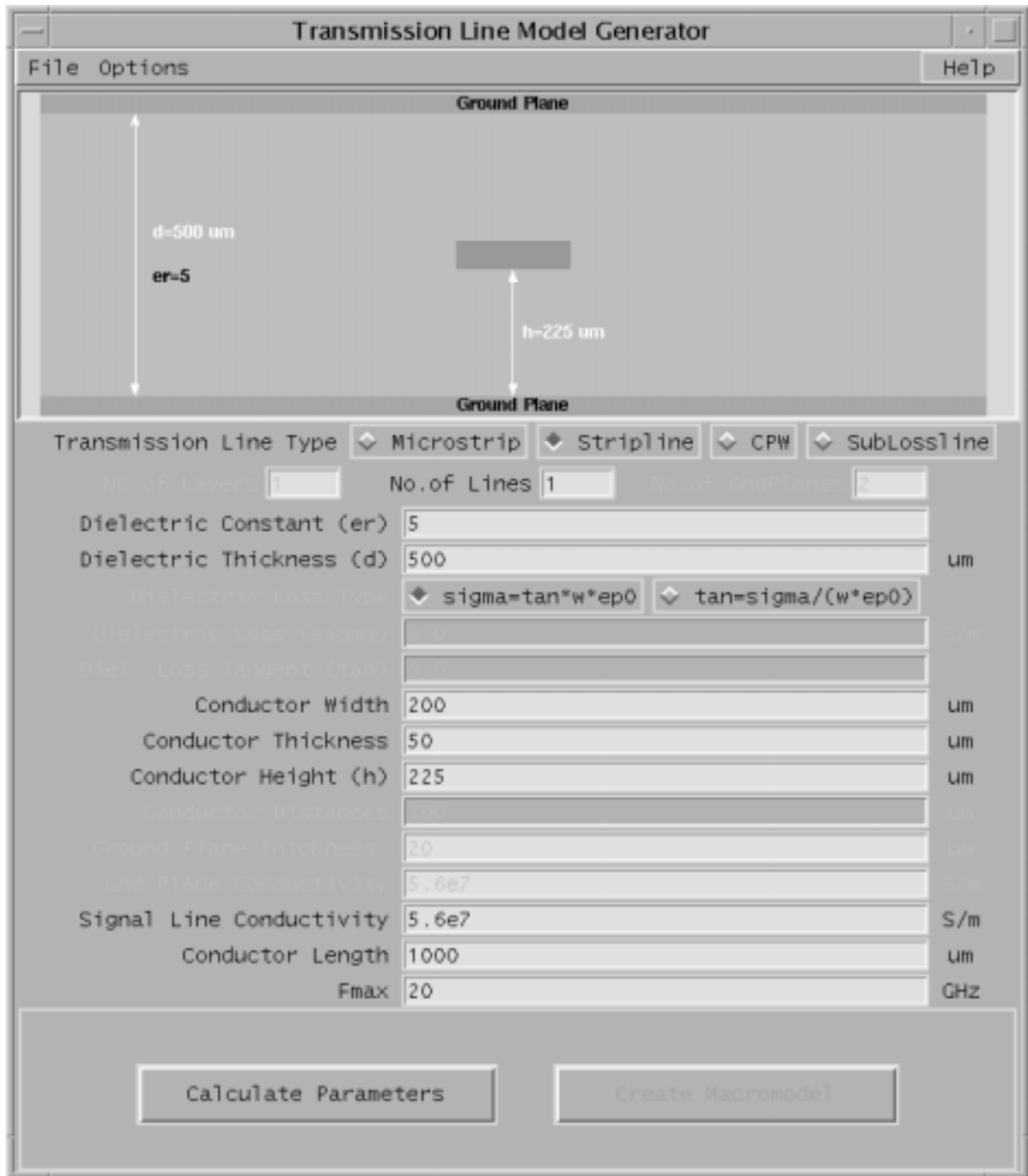
```
/home/belinda/simulation/tline3oscRF1mg/spectre/schematic/netlist
```

1. Click *OK* to display the LMG GUI.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

2. The LMG GUI appears similar to the one below.



3. Choose *Options—Model Type—Lossless* to model a lossless transmission line.

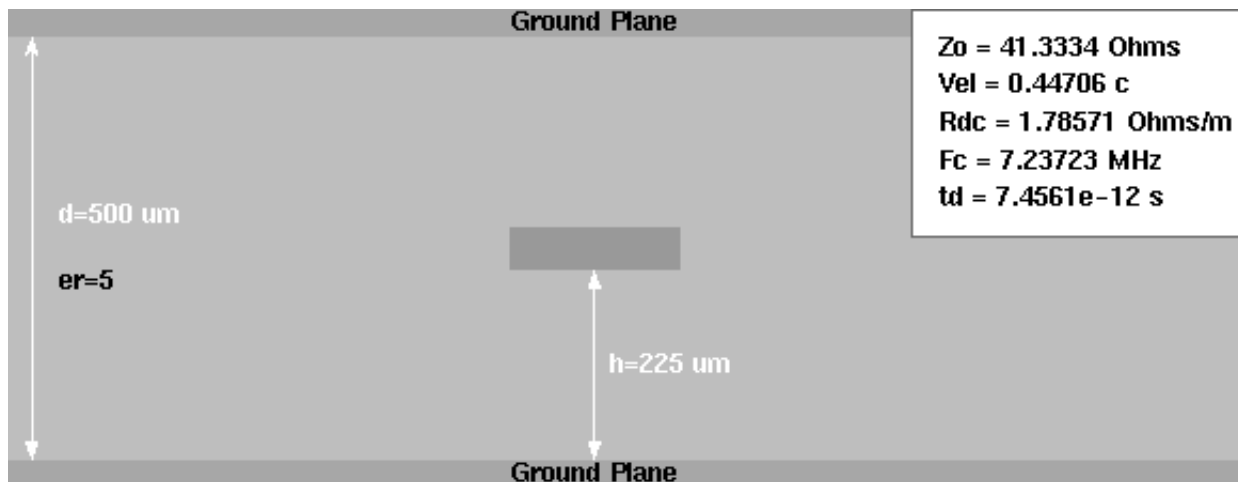
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

4. Choose *Options—Subckt Format—Spectre*.
5. This choice assumes that you are running your simulation in the analog design environment.
6. Choose *Options—Length Unit—um* to specify *um* as the length unit.
7. Choose *Options—Freq. Unit—GHz* to specify *GHz* as the frequency unit.
8. Click *Stripline* to create a stripline model.

The display section of the form now shows a stripline transmission line similar to the one below.



9. Specify the following values in the data entry fields:

*No. of Lines:* 1

*Dielectric Constant (er):* 5

*Dielectric Thickness (d):* 500 um

*sigma=tan\*w\*ep0*

*Conductor Width:* 200 um

*Conductor Thickness:* 50 um

*Conductor Height (h):* 225 um

*Signal Line Conductivity:* 5.6e7 S/m

*Conductor Length:* 1000 um

*Fmax:* 20 GHz



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The data entry fields appear like the ones below.

Transmission Line Type	Microstrip	Stripline	CPW	SubLossline	
No. of Layers	1	No. of Lines	1	No. of GndPlanes	2
Dielectric Constant ( $\epsilon_r$ )	5				
Dielectric Thickness (d)	500				$\mu\text{m}$
Dielectric Loss Type	$\sigma = \tan * w * \epsilon_0$	$\tan = \sigma / (w * \epsilon_0)$			
Dielectric Loss ( $\sigma$ )	0				S/m
Diel. Loss Tangent ( $\tan$ )	0				
Conductor Width	200				$\mu\text{m}$
Conductor Thickness	50				$\mu\text{m}$
Conductor Height (h)	225				$\mu\text{m}$
Conductor Distances	0				$\mu\text{m}$
Ground Plane Thickness	20				$\mu\text{m}$
Gnd Plane Conductivity	$5.6e7$				S/m
Signal Line Conductivity	$5.6e7$				S/m
Conductor Length	1000				$\mu\text{m}$
Fmax	20				GHz

#### 10. Click the *Calculate Parameters* button.

This calculates the per unit length LRCG matrixes and the following model parameter values:

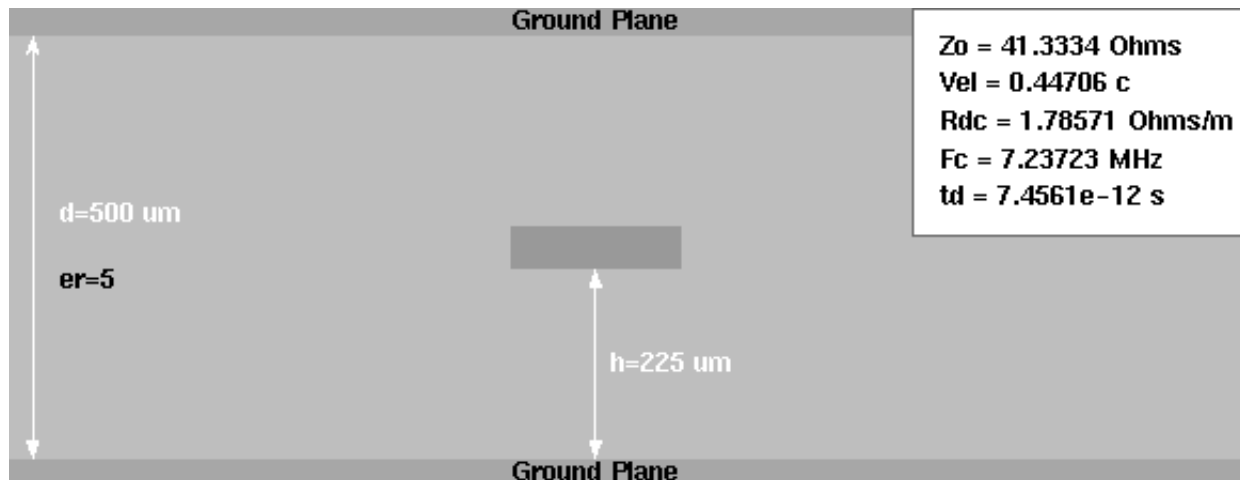
- Characteristic impedance,  $Z_0$  of 41.3334 Ohms
- Propagation velocity,  $V_{el}$  of 0.44706 c
- DC resistance per meter,  $R_{dc}$  of 1.78571 Ohms/m
- Corner frequency of the conductor,  $F_c$  of 7.23723 MHz
- Time delay of the transmission line,  $t_d$  of  $7.4561e^{-12}$  s

The display section changes to reflect the values you specified. The calculated parameter values display in the top right corner of the display section.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The calculated parameter values display in LMG only when the *No. of Lines* field is 1. (The *No. of Lines* field is in the second line of data entry fields from the top of the GUI.)



At this point, the *Create Macromodel* button at the bottom of the GUI is enabled.

#### 11. Choose *File—Subcircuit Name*.

The Subcircuit Name form appears.



#### 12. Type `tline_s1000_20G` in the *Subcircuit Name* field and click *OK*.

This model name provides descriptive information about the model type, its length, and the maximum frequency of the subcircuit.

#### 13. Choose *File—LRCG file Name*.

14. The LRCG File Name form appears.

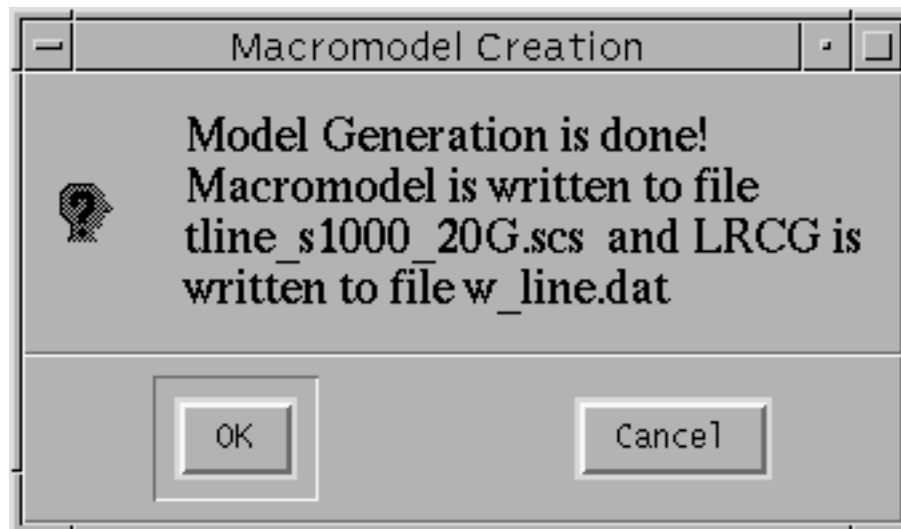


15. Type `w_line.dat` in the *LRCG Name* field and click *OK*.

This is the name of the file containing the LRCG matrixes in a format consistent with the *tline3* model.

16. Click *Create Macromodel*, to create the macromodel and the LRCG data file.

17. The Macromodel Creation form displays the names of the macromodel netlist file and the LRCG data file.



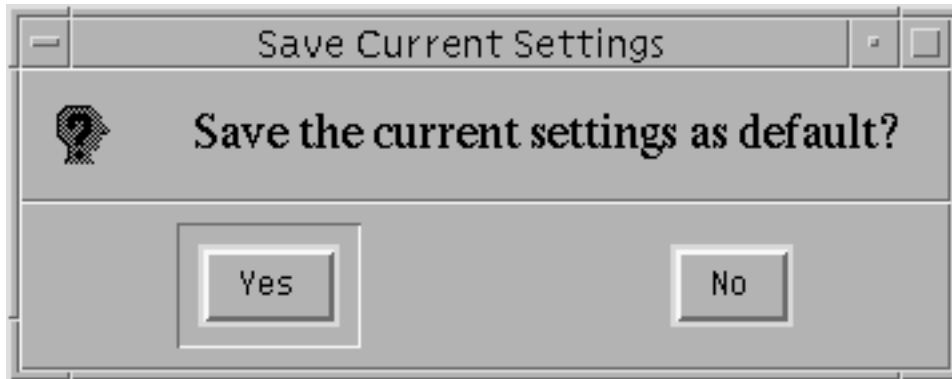
For this example, the final results are written to the following two files

- ❑ A macromodel netlist file with the name you specified and a `scs` suffix—`tline_s1000_20G.scs`.
- ❑ An LRCG file with the name `w_line.dat`.

Make a note of the name of this model as you need to enter the model name in another example, [“Using an Existing \*tline3\* Macromodel in the Schematic Flow”](#) on page 205.

Later, you use the macromodel file name as a property of a *tline3* symbol in the analog design environment.

18. Click *OK* in the Macromodel Creation form.
19. To save the current settings as default values for subsequent LMG runs, choose *File—Save Current Settings*.



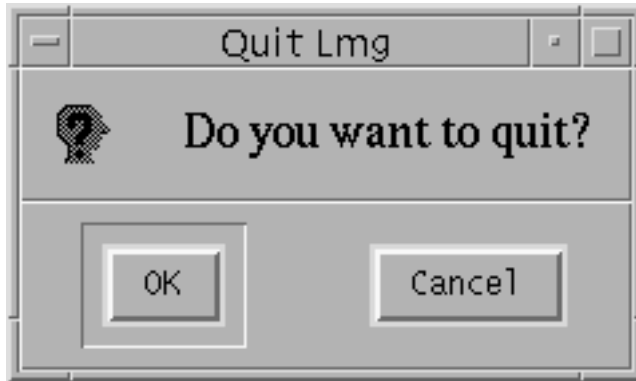
The current settings are saved in file `./initlmg`. LMG reads this file in subsequent start-ups. The `./initlmg` file contains all the quantities you can change. As long as you preserve its format, you can edit this file directly to choose the LMG startup defaults

The contents of the `./initlmg` file following this example are as follows.

```
lmgLineType = Stripline
lmgLengthUnit = um
lmgFreqUnit = GHz
lmgDielectricPermittivity = 5
lmgDielectricThickness = 500
lmgConductorWidth = 200
lmgConductorThickness = 50
lmgConductorHeight = 225
lmgConductorLength = 1000
lmgConductivity = 5.6e7
lmgMaxFreq = 20
lmgSubCircuitName = tline_s1000_20G
```

20. Click *Yes*.
21. At the end of the run, choose *File—Quit* to exit LMG.

The following form displays



22. Click *OK*.

LMG closes.

## Using an Existing *tline3* Macromodel in the Schematic Flow

This example illustrates how to call an existing *tline3* macromodel from the schematic flow while you are using the analog design environment.

This example uses the circuit `tline3oscRFImg` from your editable copy of the `rfExamples` library (*my\_rfExamples* here). This library also includes other sample circuits used in this manual. If you need assistance setting up or accessing your editable copy of the `rfExamples` library, refer to [Chapter 3, "Setting Up for the Examples."](#) You also need the absolute path to the *tline3* macromodel you created in the first example.

### Opening the *tline3oscRFImg* Circuit in the Schematic Window

1. In the CIW, choose *File – Open*.

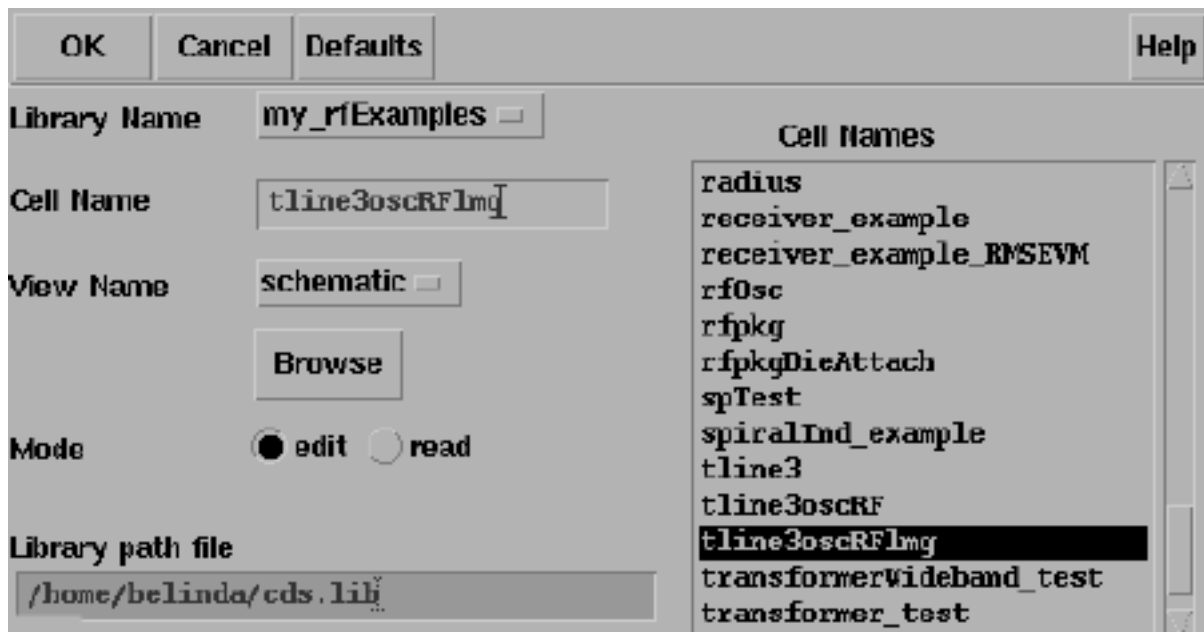
The Open File form appears.

2. In the Open File form, choose *my\_rfExamples* in the *Library Name* cyclic field and choose *tline3oscRFImg* in the *Cell Names* list box.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

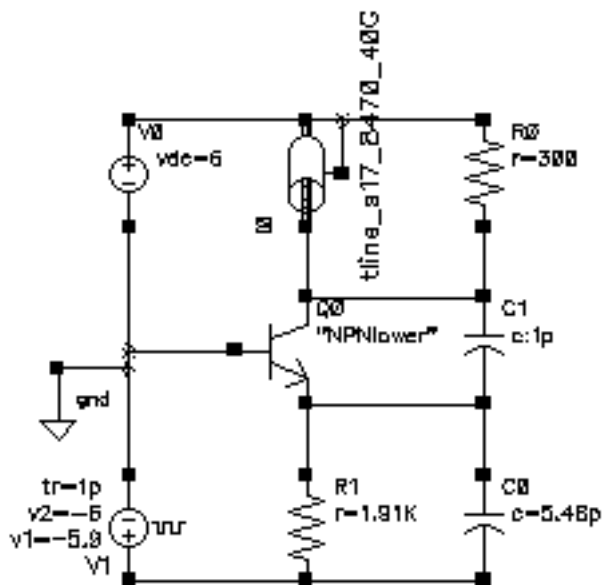
### Modeling Transmission Lines

The completed Open File form appears like the one below.



3. Click *OK*.

The Schematic window for the tline3oscRF1mg circuit appears.



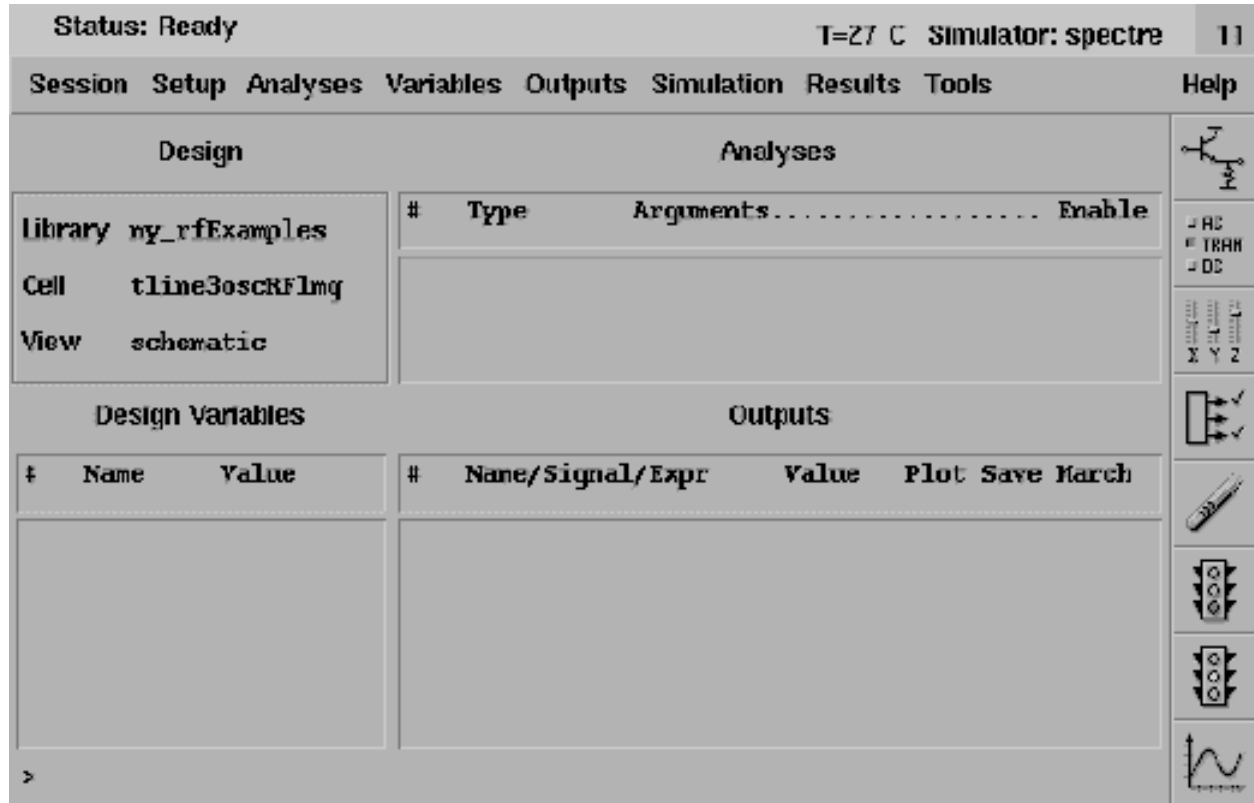
4. In the Schematic window, choose *Tools—Analog Environment*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmission Lines

---

The Simulation window opens.



### Setting Up the Model Path

1. In the Simulation window, choose *Setup—Model Libraries*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The Model Library Setup form appears.

Model Library File	Section

Model Library File:

Section (opt.):

Buttons: Add, Delete, Change, Edit File, Browse...

2. In the *Model Library File* field, type the full, absolute path to the model file including the model file name, `tline_s1000_20G.scs`.
3. The absolute path is displayed in the Transmission Line Modeler form in the first example.
4. The model name is the name of the model generated during the first example. The model name is displayed in the Subcircuit Name form in the first example.
5. For example, type the following in the *Model Library File* field:  

```
/home/belinda/simulation/tline3oscRFlmg/spectre/schematic/netlist/  
tline_s1000_20G.scs
```
6. Click *Add*.
7. Click *OK*.

### Associating the External Model File with the Transmission Line Component

1. In the Schematic window, click the transmission line component `tline_s17_8470_40G` near the top of the schematic.

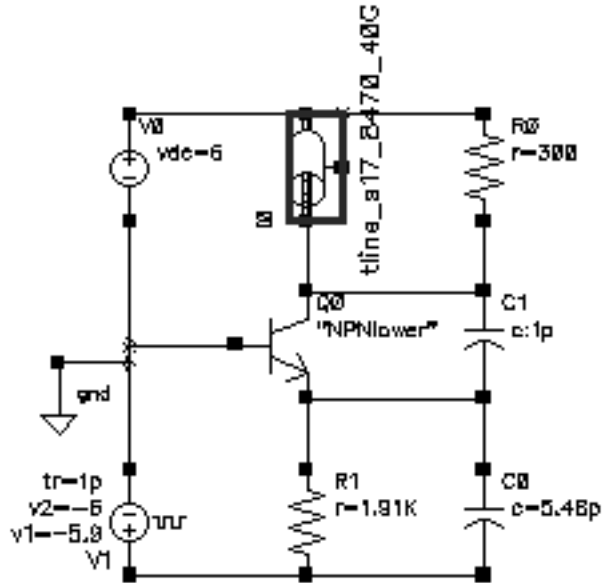


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The transmission line is now selected.



2. With the transmission line selected in the Schematic window, choose *Edit—Properties—Objects* to open the Edit Object Properties form for the selected  $tline\_s17\_8470\_40G$  transmission line component.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The *Edit—Properties—Objects* form appears similar to the following.

Apply To			<input type="text" value="only current"/>	<input type="text" value="instance"/>	
Show			<input type="checkbox"/> system	<input checked="" type="checkbox"/> user	<input checked="" type="checkbox"/> CDF
Browse			Reset Instance Labels Display		
Property	Value	Display			
Library Name	<input type="text" value="rfExamples"/>	off <input type="checkbox"/>			
Cell Name	<input type="text" value="tline3"/>	off <input type="checkbox"/>			
View Name	<input type="text" value="symbol"/>	off <input type="checkbox"/>			
Instance Name	<input type="text" value="I0"/>	off <input type="checkbox"/>			
Add			Delete	Modify	
CDF Parameter	Value	Display			
use external model file	<input type="checkbox"/>	off <input type="checkbox"/>			
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	off <input type="checkbox"/>			
er: permittivity	<input type="text" value="12.9"/>	off <input type="checkbox"/>			
d: dielectric thickness (m)	<input type="text" value="100e-6"/>	off <input type="checkbox"/>			
w: conductor width (m)	<input type="text" value="71e-6"/>	off <input type="checkbox"/>			
t: conductor thickness (m)	<input type="text" value="5e-6"/>	off <input type="checkbox"/>			
len: conductor length (m)	<input type="text" value="8.47e-3"/>	off <input type="checkbox"/>			
model type	<input type="radio"/> lossless <input checked="" type="radio"/> lossy	off <input type="checkbox"/>			
lossy type	<input checked="" type="radio"/> narrow band <input type="radio"/> wide band	off <input type="checkbox"/>			
sigma: conductivity	<input type="text" value="5.6e7"/>	off <input type="checkbox"/>			
freq: max frequency (Hz)	<input type="text" value="40e9"/>	off <input type="checkbox"/>			

- In the Edit Object Properties form, highlight *use external model file*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

- When you highlight *use external model file*, the form changes to let you specify only the *Model Name*.

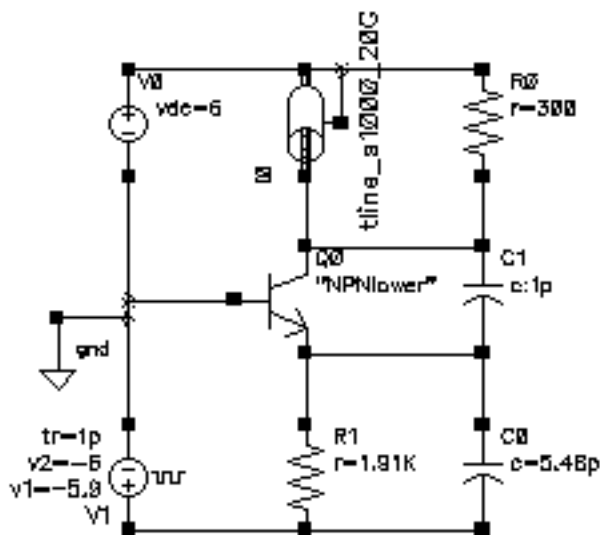
	Add	Delete	Modify
<b>CDF Parameter</b>			
<b>use external model file</b>	<input checked="" type="checkbox"/>		Display <input type="checkbox"/> off
<b>Model Name</b>	<input style="width: 100%;" type="text"/>		value <input type="checkbox"/>

- In the *Model Name* field, type `tline_s1000_20G`, the name of the existing *tline* macromodel you created using LMG.

<b>CDF Parameter</b>	<b>Value</b>
<b>use external model file</b>	<input checked="" type="checkbox"/>
<b>Model Name</b>	<input style="width: 100%;" type="text" value="tline_s1000_20G"/>

- In the Edit Object Properties form, click *OK*.

The component name in the Schematic window changes to reflect the new *tline* macromodel name.



- Check and Save the `tline3oscRF1mg` schematic.

8. You can now set up and run a simulation using the new *tline* macromodel in the analog design environment.

## Resolving A Possible Error Message for a tline3 Model File

In some circumstances the analog design environment might issue an error message stating that a `tline3` subckt has four terminals, while in a schematic it is a 3-terminal device.

To solve this problem, you can open and edit the subcircuit file with a text editor of your choice. For example, the `tline_s1000_20G.scs` file looks like the following.

```
//  
// Gaussian quadrature model to transmission line.  
// Transmission line type: Stripline  
// Number of signal conductors: 1  
subckt tline_s1000_20G (in0 out0 ref refequal)  
  10_0 (in0 n1_0) inductor l=1.445788e-11  
  11_0 (n1_0 n2_0) inductor l=5.666485e-11  
  12_0 (n2_0 n3_0) inductor l=8.297930e-11  
  13_0 (n3_0 n4_0) inductor l=8.297930e-11  
  14_0 (n4_0 n5_0) inductor l=5.666485e-11  
  15_0 (n5_0 out0) inductor l=1.445788e-11  
  c0_0 (n1_0 ref) capacitor c=2.137783e-14  
  c1_0 (n2_0 ref) capacitor c=4.318660e-14  
  c2_0 (n3_0 ref) capacitor c=5.133076e-14  
  c3_0 (n4_0 ref) capacitor c=4.318660e-14  
  c4_0 (n5_0 ref) capacitor c=2.137783e-14  
ends tline_s1000_20G
```

To edit the `tline_s1000_20G.scs` subcircuit file with the text editor, use the following steps.

Find `refequal` and delete it from the `tline_s1000_20G.scs` file.

```
//  
// Gaussian quadrature model to transmission line.  
// Transmission line type: Stripline  
// Number of signal conductors: 1  
subckt tline_s1000_20G (in0 out0 ref)  
  10_0 (in0 n1_0) inductor l=1.445788e-11  
  11_0 (n1_0 n2_0) inductor l=5.666485e-11  
  12_0 (n2_0 n3_0) inductor l=8.297930e-11
```

```
l3_0 (n3_0 n4_0) inductor l=8.297930e-11
l4_0 (n4_0 n5_0) inductor l=5.666485e-11
l5_0 (n5_0 out0) inductor l=1.445788e-11
c0_0 (n1_0 ref) capacitor c=2.137783e-14
c1_0 (n2_0 ref) capacitor c=4.318660e-14
c2_0 (n3_0 ref) capacitor c=5.133076e-14
c3_0 (n4_0 ref) capacitor c=4.318660e-14
c4_0 (n5_0 ref) capacitor c=2.137783e-14
ends tline_s1000_20G
```

1. Save the edited `tline_s1000_20G.scs` file.

After you edit the model file, you can perform a simulation using the `tline_s1000_20G.scs` model file without errors.

## Creating a `tline3` Macromodel in the Schematic Flow

This example illustrates how to create a `tline3` transmission line macromodel from the schematic flow while you are using the analog design environment.

This example follows the second use model in that it does not use the GUI. You can use this procedure if you do not need to see the two-dimensional cross-section display that the LMG GUI provides. This example creates a transmission line that is identical to the one created in the previous example [“Using an Existing `tline3` Macromodel in the Schematic Flow”](#) on page 205.

This example uses the circuit `tline3oscRFImg` from your editable copy of the `rfExamples` library, `my_rfExamples`. This library also includes other sample circuits used in this manual. If you need assistance setting up or accessing the `rfExamples` library, refer to [Chapter 3, “Setting Up for the Examples.”](#)

## Opening the `tline3oscRFImg` Circuit in the Schematic Window

1. In the CIW, choose *File—Open*.

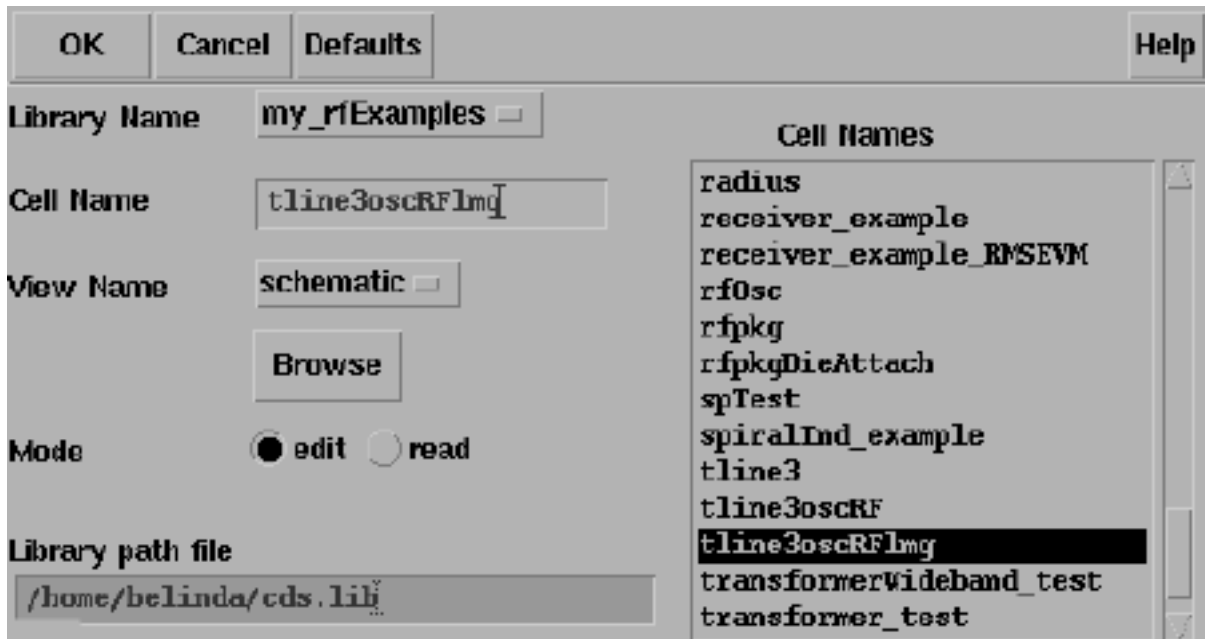
The Open File form appears.

2. In the Open File form, choose your editable copy of the `rfExamples` library, in this case `my_rfExamples` in the *Library Name* cyclic field and choose `tline3oscRFImg` in the *Cell Names* list box.

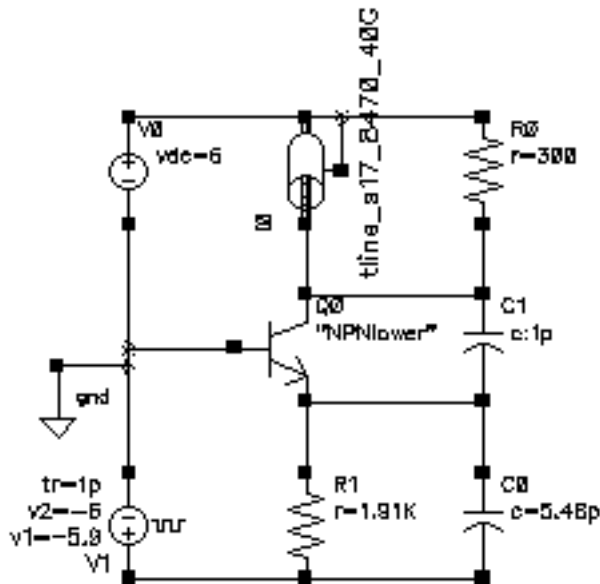
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The completed Open File form appears like the one below.



3. Click **OK**.
4. The Schematic window for the `tline3oscRF1mg` circuit appears.



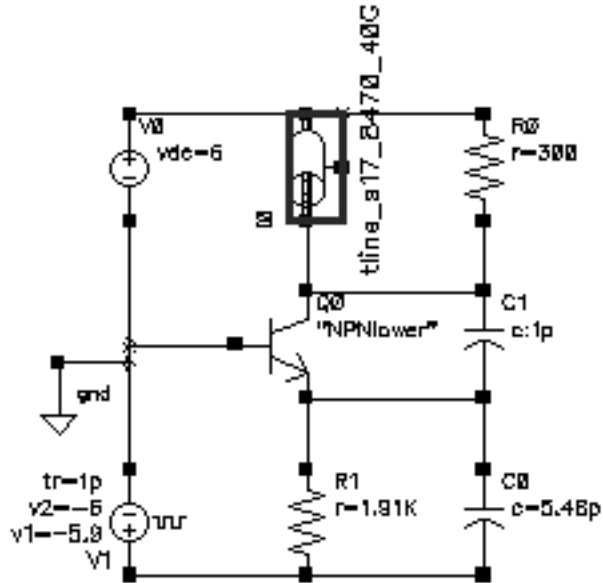
5. In the Schematic window, click the transmission line `tline_s17_8470_40G` near the top of the schematic.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The transmission line is now selected.



6. In the Schematic window, choose *Edit—Properties—Objects* with the transmission line selected to open the Edit Object Properties form for the transmission line component.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The *Edit – Properties – Objects* form for *tline\_s17\_8470\_40G* appears.

Apply To		
<input type="button" value="only current"/>	<input type="button" value="instance"/>	
Show		
<input type="checkbox"/> system	<input checked="" type="checkbox"/> user	<input checked="" type="checkbox"/> CDF
<input type="button" value="Browse"/> <input type="button" value="Reset Instance Labels Display"/>		
Property	Value	Display
Library Name	<input type="text" value="rfExamples"/>	<input type="checkbox"/> off
Cell Name	<input type="text" value="tline3"/>	<input type="checkbox"/> off
View Name	<input type="text" value="symbol"/>	<input type="checkbox"/> off
Instance Name	<input type="text" value="I0"/>	<input type="checkbox"/> off
<input type="button" value="Add"/> <input type="button" value="Delete"/> <input type="button" value="Modify"/>		
CDF Parameter	Value	Display
use external model file	<input type="checkbox"/>	<input type="checkbox"/> off
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	<input type="checkbox"/> off
er: permittivity	<input type="text" value="12.9"/>	<input type="checkbox"/> off
d: dielectric thickness (m)	<input type="text" value="100e-6"/>	<input type="checkbox"/> off
w: conductor width (m)	<input type="text" value="71e-6"/>	<input type="checkbox"/> off
t: conductor thickness (m)	<input type="text" value="5e-6"/>	<input type="checkbox"/> off
len: conductor length (m)	<input type="text" value="8.47e-3"/>	<input type="checkbox"/> off
model type	<input type="radio"/> lossless <input checked="" type="radio"/> lossy	<input type="checkbox"/> off
lossy type	<input checked="" type="radio"/> narrow band <input type="radio"/> wide band	<input type="checkbox"/> off
sigma: conductivity	<input type="text" value="5.6e7"/>	<input type="checkbox"/> off
freq: max frequency (Hz)	<input type="text" value="40e9"/>	<input type="checkbox"/> off



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

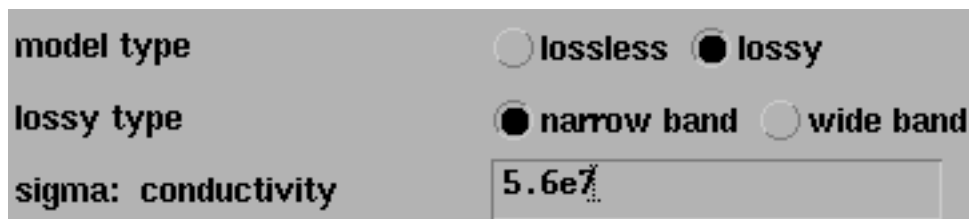
7. In the Edit Object Properties form, perform the following edits in the *CDF Parameter* section in the bottom half of the form:

- Highlight *stripline* for *line type*.
- Type 5 for *er: permittivity*.
- Type  $500e-6$  for *d: dielectric thickness (m)*.

**Note:** Notice that in the Edit Object Properties form, the values you enter are in units of meters and Hertz. These units are probably different than the units you used in LMG.

- Type  $200e-6$  for *w: conductor width (m)*.
- Type  $5e-6$  for *t: conductor thickness (m)*.
- Type  $5e-6$  for *h: conductor height (m)*.
- Type  $1e-3$  for *len: conductor length (m)*.
- Highlight *lossless* for the *model type*.

**Note:** If you choose *lossy* for the *model type*, two additional fields open to let you specify the *lossy type* and *conductivity*.



model type  lossless  lossy

lossy type  narrow band  wide band

sigma: conductivity

- Type  $40e9$  for *freq: max frequency (Hz)*.

After edits, the CDF parameters appear as follows

CDF Parameter	Value
use external model file	<input type="checkbox"/>
line type	<input type="radio"/> microstrip <input checked="" type="radio"/> stripline
er: permittivity	5
d: dielectric thickness (m)	500e-6
w: conductor width (m)	200e-6
t: conductor thickness (m)	5e-6
h: conductor height (m)	5e-6
len: conductor length (m)	1e-3
model type	<input checked="" type="radio"/> lossless <input type="radio"/> lossy
freq: max frequency (Hz)	40e9

8. In the Edit Object Properties form, click *OK*.
9. Check and Save the *tline3oscRFImg* schematic.
10. You can now set up and run a simulation using the analog design environment.

## Mline Transmission Line Modeling Examples

This section presents a series of examples showing how to create an *mline* transmission line model. Because the procedures for creating microstrip and stripline transmission lines are very similar, a microstrip example is shown here.

### Creating an mline Transmission Line Model Starting LMG From UNIX

This example starts LMG from the UNIX prompt. It illustrates how to create an *mline* model for 2 microstrip transmission lines using the LMG GUI.

At the UNIX prompt, type `lmg &` and press `return` to start the LMG GUI.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

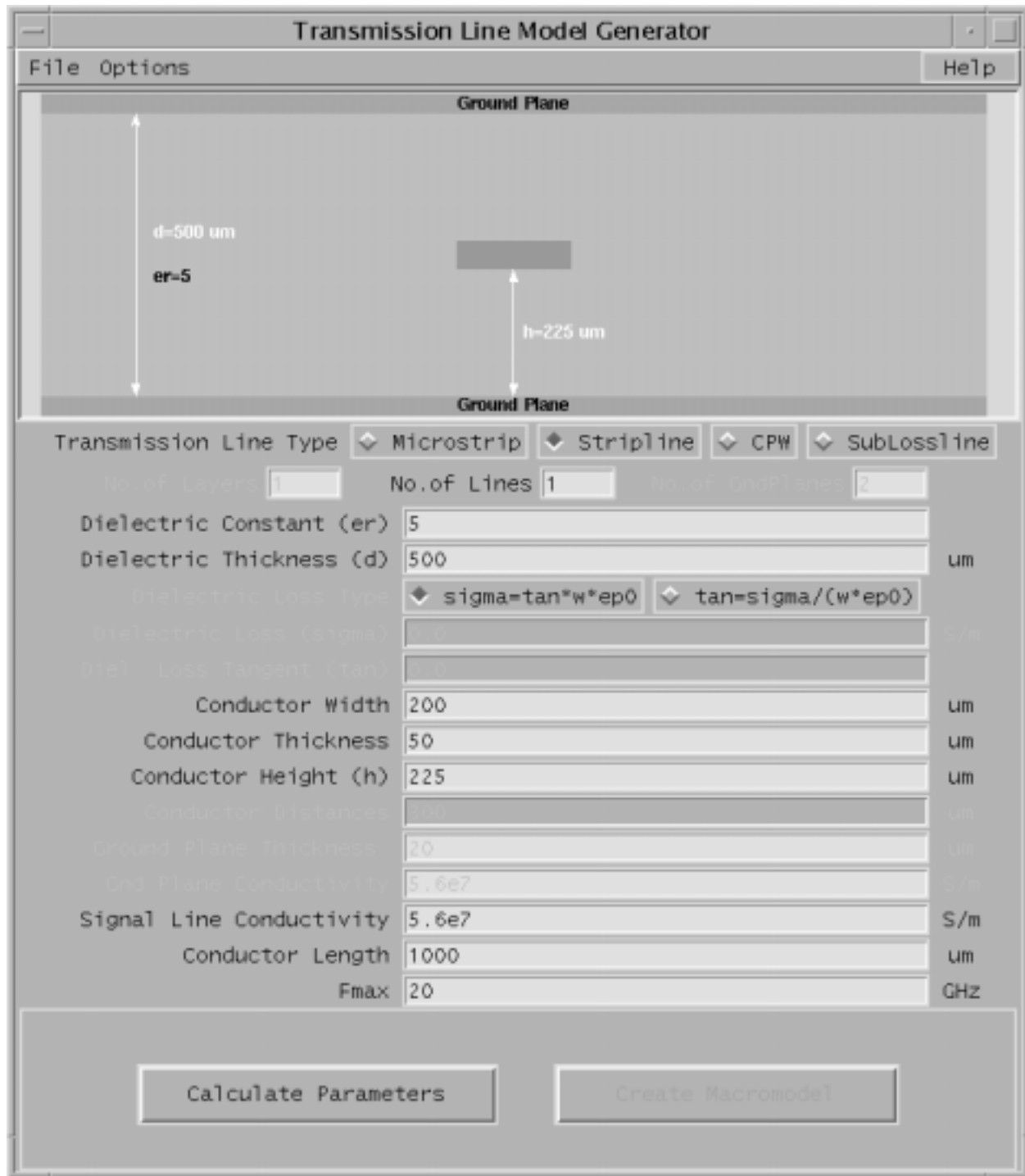
% lmg &

**Note:** When you start LMG from the UNIX prompt, LMG does not display the Transmission Line Modeler form and thus does not display the path to the directory where the models are saved.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmission Lines

The LMG GUI displays similar to the one below.



1. Choose *Options—Model Type—Lossless* to model a lossless transmission line.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

2. In the LMG GUI, choose *Options—Subckt Format—Spectre*.

This choice assumes that you are running your simulation in the analog design environment.

3. Choose *Options—Length Unit—um* to specify *um* as the length unit.
4. Choose *Options—Freq. Unit—GHz* to specify *GHz* as the frequency unit.
5. Choose *File—Subcircuit Name*.

The Subcircuit Name form appears.



6. Type `tline_s1` in the *Subcircuit Name* field and click *OK*.
7. Choose *File—LRCG file Name*.

The LRCG file Name form appears.



8. Type `w_line1.dat` in the *LRCG Name* field and click *OK*.
9. Specify the following values in the data entry fields:

*Transmission Line Type*: Microstrip

*No. of Lines*: 2

*Dielectric Constant (er)*: 12.9

*Dielectric Thickness (d)*: 200 *um*

$\sigma = \tan * w * \epsilon_0$

*Conductor Width*: 300 *um*

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

*Conductor Thickness:* 10  $\mu\text{m}$

*Conductor Distances:* 150  $\mu\text{m}$

*Signal Line Conductivity:* 5.6e7 S/m

*Conductor Length:* 5000  $\mu\text{m}$

*Fmax:* 30 GHz

10. Choose *File—Save Current Settings* to save the current settings in the LMG GUI as default values for subsequent LMG runs.

The current settings are saved in the file `./initlmg`.

When LMG starts, it reads the `./initlmg` file which contains all the quantities that can be modified. As long as you preserve the `./initlmg` file format, you can edit this file directly to choose the LMG startup defaults.

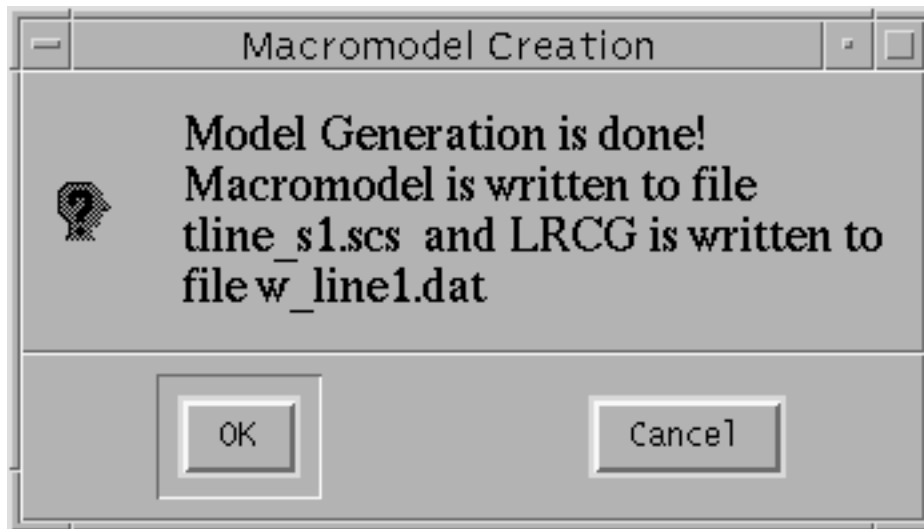
11. Click the *Calculate Parameters* button to extract the microstrip model parameters.

The *Create Macromodel* button is enabled when the parameters are extracted.



12. Click the *Create Macromodel* button when it is enabled.

The Macromodel Creation form displays when the macromodel and data files are created.



The macromodel subckt is written to file *tline\_s1.scs* and the LRCG data file is written to file *w\_line1.dat*.

13. Click *OK* to create both files.
14. Choose *File—Quit* to exit LMG and close the LMG GUI.

## Using an *mline* Macromodel in the Schematic Flow

The example presented in this section illustrates how to use an *mline* macromodel you created in the previous example from the schematic flow while you are using the analog design environment. This example uses the *mlineoscRFImg* circuit from your editable copy of the *rfExamples* library.

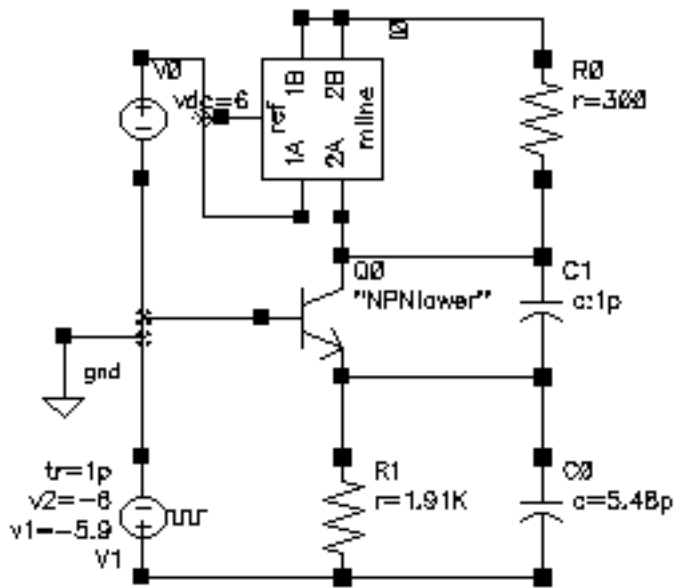
### Opening the *mlineoscRFImg* Circuit in the Schematic Window

1. In the *CIW*, choose *File—Open*.

The Open File form appears.

2. In the Open File form, choose the editable copy of the *rfExamples* library, *my\_rfExamples* in the *Library Name* cyclic field and choose *mlineoscRFImg* in the *Cell Names* list box and click *OK*.

The Schematic window for the *mlineoscRFImg* circuit appears.



### Setting Up the Model Path

1. In the Schematic window, choose *Tools—Analog Environment* to open the Simulation window.

The Simulation window appears.

2. In the Simulation window, choose *Setup—Model Libraries*.

The Model Library Setup form appears.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The Model Library Setup form appears.

3. In the *Model Library File* field, type the full, absolute path to the model file plus the file name `tline_sl.scs`.

The absolute path is the path to the directory where you started LMG in the previous example, “[Creating an mline Transmission Line Model Starting LMG From UNIX](#)” on page 218.

The model name is the name of the *mline* model displayed in the [Subcircuit Name form](#) in the previous example.

For example, type the following in the *Model Library File* field:

```
/home/belinda/tline_sl.scs
```

4. Click *Add*.
5. Click *OK*.

### Associating the External Model File with the Transmission Line Component

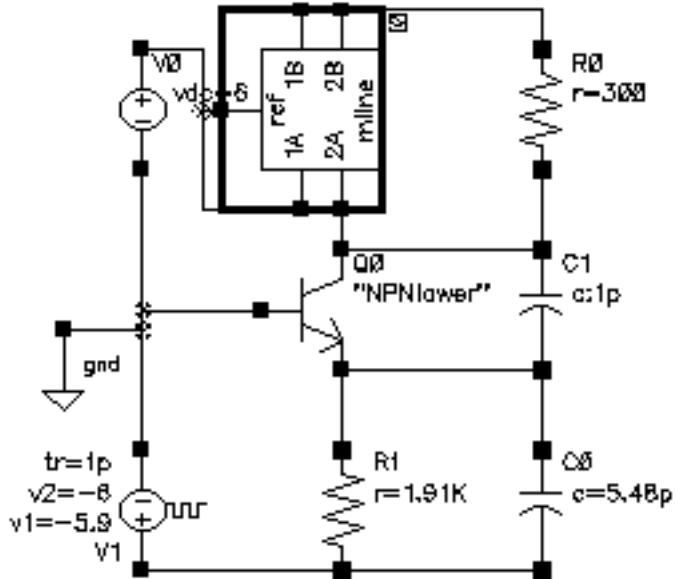
1. In the Schematic window, click the *mline* symbol near the top of the schematic.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The *mline* transmission line is now selected.



2. Choose *Edit—Properties—Objects* in the Schematic window to open the Edit Object Properties form for the selected *mline* transmission line component. The CDF parameters display at the bottom of the form.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The CDF parameters for the *mline* component appear below.

CDF Parameter	Value
Number of Conductors	2
use external model file	<input type="checkbox"/>
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline
er: permittivity	12.9
d: dielectric thickness (m)	100e-6
w: conductor widths (m)	300e-6
cd: conductor distances (m)	50e-6
t: conductor thickness (m)	5e-6
len: conductor length (m)	8.47e-3
sigma: signal conductivity	5.6e7
model type	<input checked="" type="radio"/> lossless <input type="radio"/> lossy
freq: max frequency (Hz)	40e9

3. In the CDF Parameters area, highlight *use external model file*, so the form changes to let you specify the model name.

CDF Parameter	Value
Number of Conductors	2
use external model file	<input checked="" type="checkbox"/>
Model Name	tline_s17_8470_40G

4. In the Model Name field, type the name of the mline model you created, `tline_s1`.

CDF Parameter	Value
Number of Conductors	2
use external model file	<input type="checkbox"/>
Model Name	tline_s1

5. In the Edit Object Properties form, click *OK*.
6. Check and Save the *mlineoscRFImg* schematic.

You can now set up and run a simulation using the new *mline* macromodel in the analog design environment.

## Resolving a Possible Error Message for an mline Model File

In some circumstances the analog design environment might issue an error message stating that an `mline` subckt has six terminals, while in the schematic it is a 5-terminal device.

To solve this problem, you can open and edit the subcircuit file with the text editor of your choice. The `tline_s1.scs` file looks like the following.

```
//  
// Gaussian quadrature model to transmission line.  
// Transmission line type: Microstrip  
// Number of signal conductors: 2  
subckt tline_s1 (in0 out0 in1 out1 ref refequal)  
  10_0 (in0 n1_0) inductor l=2.655224e-11  
  11_0 (n1_0 n2_0) inductor l=1.101868e-10  
  12_0 (n2_0 n3_0) inductor l=1.856834e-10  
  13_0 (n3_0 n4_0) inductor l=2.411049e-10  
  14_0 (n4_0 n5_0) inductor l=2.704059e-10  
  15_0 (n5_0 n6_0) inductor l=2.704059e-10  
  16_0 (n6_0 n7_0) inductor l=2.411049e-10  
  17_0 (n7_0 n8_0) inductor l=1.856834e-10  
  18_0 (n8_0 n9_0) inductor l=1.101868e-10  
  19_0 (n9_0 out0) inductor l=2.655224e-11  
  10_1 (in1 n1_1) inductor l=2.655224e-11  
  11_1 (n1_1 n2_1) inductor l=1.101868e-10  
  12_1 (n2_1 n3_1) inductor l=1.856834e-10  
  13_1 (n3_1 n4_1) inductor l=2.411049e-10  
  14_1 (n4_1 n5_1) inductor l=2.704059e-10  
  15_1 (n5_1 n6_1) inductor l=2.704059e-10  
  16_1 (n6_1 n7_1) inductor l=2.411049e-10  
  17_1 (n7_1 n8_1) inductor l=1.856834e-10  
  18_1 (n8_1 n9_1) inductor l=1.101868e-10  
  19_1 (n9_1 out1) inductor l=2.655224e-11
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

```
m10_1_0 mutual_inductor coupling=2.207763e-01 ind1=10_1 ind2=10_0
m10_1_1 mutual_inductor coupling=2.207763e-01 ind1=11_1 ind2=11_0
m10_1_2 mutual_inductor coupling=2.207763e-01 ind1=12_1 ind2=12_0
m10_1_3 mutual_inductor coupling=2.207763e-01 ind1=13_1 ind2=13_0
m10_1_4 mutual_inductor coupling=2.207763e-01 ind1=14_1 ind2=14_0
m10_1_5 mutual_inductor coupling=2.207763e-01 ind1=15_1 ind2=15_0
m10_1_6 mutual_inductor coupling=2.207763e-01 ind1=16_1 ind2=16_0
m10_1_7 mutual_inductor coupling=2.207763e-01 ind1=17_1 ind2=17_0
m10_1_8 mutual_inductor coupling=2.207763e-01 ind1=18_1 ind2=18_0
m10_1_9 mutual_inductor coupling=2.207763e-01 ind1=19_1 ind2=19_0
c0_0 (n1_0 ref) capacitor c=5.315752e-14
c1_0 (n2_0 ref) capacitor c=1.181530e-13
c2_0 (n3_0 ref) capacitor c=1.704520e-13
c3_0 (n4_0 ref) capacitor c=2.042912e-13
c4_0 (n5_0 ref) capacitor c=2.159935e-13
c5_0 (n6_0 ref) capacitor c=2.042912e-13
c6_0 (n7_0 ref) capacitor c=1.704520e-13
c7_0 (n8_0 ref) capacitor c=1.181530e-13
c8_0 (n9_0 ref) capacitor c=5.315752e-14
c0_1 (n1_1 ref) capacitor c=5.315752e-14
c1_1 (n2_1 ref) capacitor c=1.181530e-13
c2_1 (n3_1 ref) capacitor c=1.704520e-13
c3_1 (n4_1 ref) capacitor c=2.042912e-13
c4_1 (n5_1 ref) capacitor c=2.159935e-13
c5_1 (n6_1 ref) capacitor c=2.042912e-13
c6_1 (n7_1 ref) capacitor c=1.704520e-13
c7_1 (n8_1 ref) capacitor c=1.181530e-13
c8_1 (n9_1 ref) capacitor c=5.315752e-14
cm0_1_0 (n1_1 n1_0) capacitor c=5.821279e-15
cm1_1_0 (n2_1 n2_0) capacitor c=1.293893e-14
cm2_1_0 (n3_1 n3_0) capacitor c=1.866619e-14
cm3_1_0 (n4_1 n4_0) capacitor c=2.237193e-14
cm4_1_0 (n5_1 n5_0) capacitor c=2.365344e-14
cm5_1_0 (n6_1 n6_0) capacitor c=2.237193e-14
cm6_1_0 (n7_1 n7_0) capacitor c=1.866619e-14
cm7_1_0 (n8_1 n8_0) capacitor c=1.293893e-14
cm8_1_0 (n9_1 n9_0) capacitor c=5.821279e-15
ends tline_s1
```

To edit the `tline_s1.scs` subcircuit file with the text editor, use the following steps.

#### 1. Find `refequal` and delete it from the `tline_s1.scs` file.

```
//
// Gaussian quadrature model to transmission line.
// Transmission line type: Microstrip
// Number of signal conductors: 2
subckt tline_s1 (in0 out0 in1 out1 ref)
  l0_0 (in0 n1_0) inductor l=2.655224e-11
  l1_0 (n1_0 n2_0) inductor l=1.101868e-10
  l2_0 (n2_0 n3_0) inductor l=1.856834e-10
  l3_0 (n3_0 n4_0) inductor l=2.411049e-10
  l4_0 (n4_0 n5_0) inductor l=2.704059e-10
  l5_0 (n5_0 n6_0) inductor l=2.704059e-10
  l6_0 (n6_0 n7_0) inductor l=2.411049e-10
  l7_0 (n7_0 n8_0) inductor l=1.856834e-10
  .
  .
  .
  cm6_1_0 (n7_1 n7_0) capacitor c=1.866619e-14
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

```
cm7_1_0 (n8_1 n8_0) capacitor c=1.293893e-14
cm8_1_0 (n9_1 n9_0) capacitor c=5.821279e-15
ends tline_s1
```

2. Save the edited `tline_s1.scs` file.

After you edit the model file, you can perform a simulation using the `tline_s1.scs` model file without errors.

## Creating an *mline* Transmission Line in the Schematic Flow

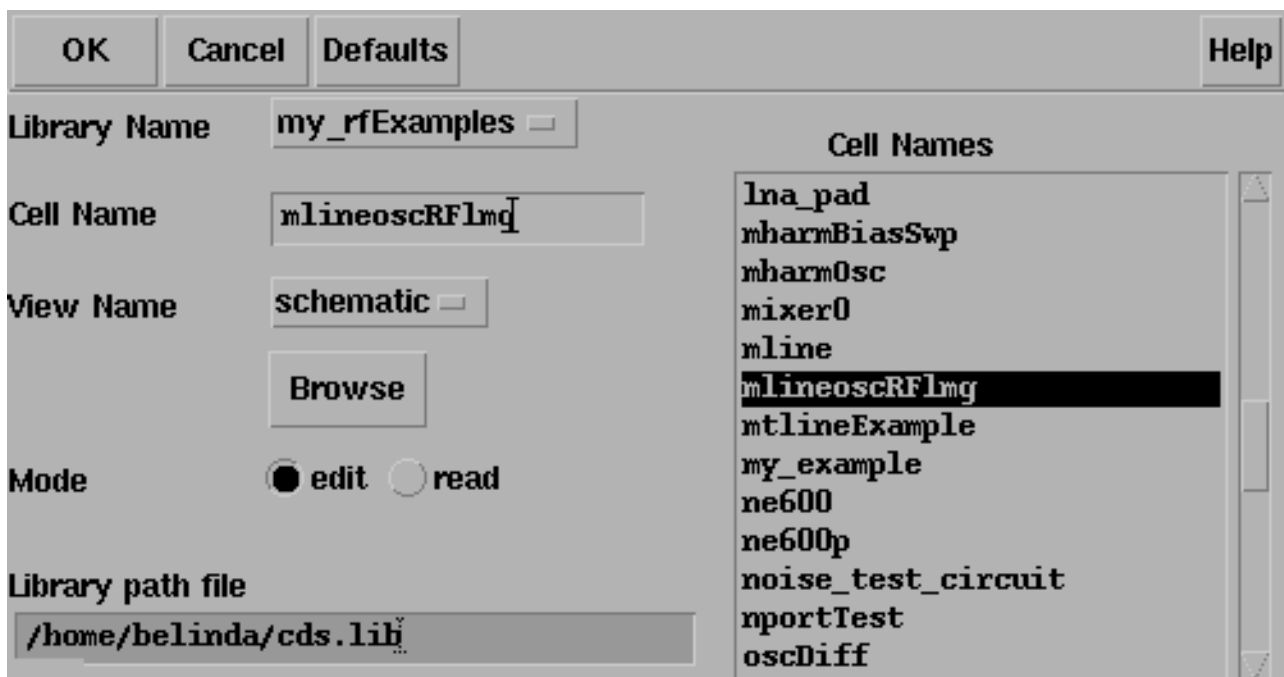
This example illustrates how to create an *mline* transmission line macromodel from the schematic flow while you are using the analog design environment. The *mline* CDF is used to invoke LMG to produce the subcircuit for the transmission line when the netlist is generated.

This example uses the circuit *mlineoscRFImg* from your editable copy of the *rfExamples* library, *my\_rfExamples*.

## Opening the *mlineoscRFImg* Circuit in the Schematic Window

1. In the CIW, choose *File—Open*.

The Open File form appears.

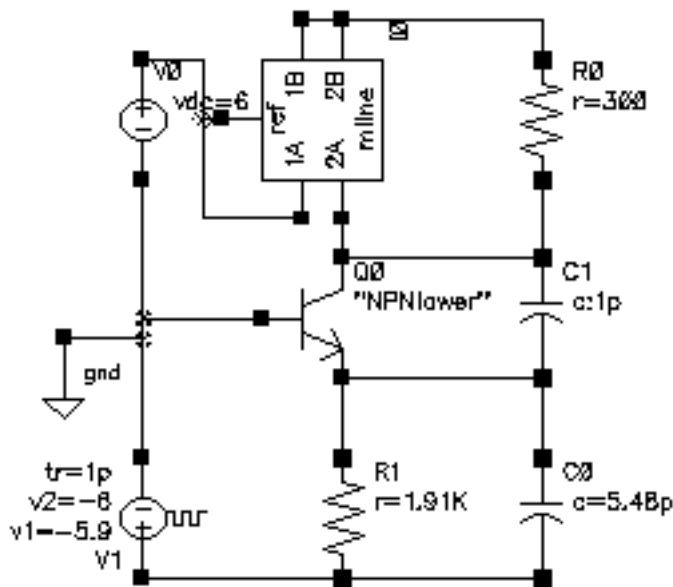


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

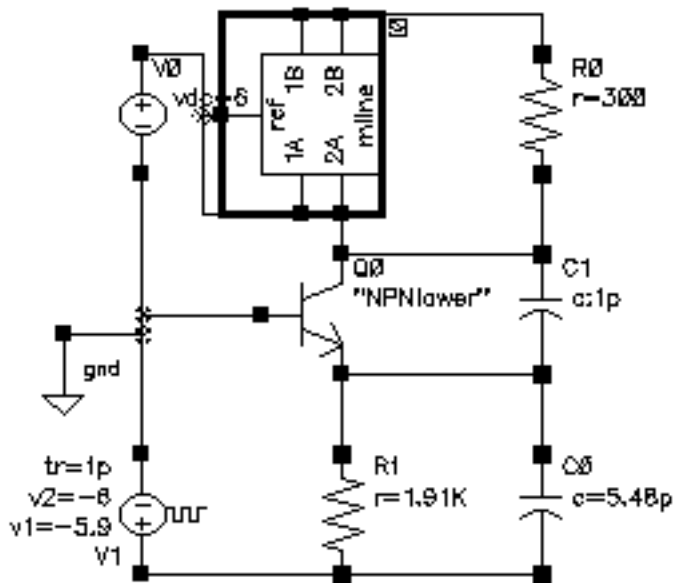
### Modeling Transmission Lines

2. In the Open File form, choose the editable copy of the *rfExamples* library, *my\_rfExamples*, in the *Library Name* cyclic field.
3. Choose *mlineoscRFImg* in the *Cell Names* list box.

The Schematic window for the *mlineoscRFImg* circuit appears.



4. In the Schematic window, select the *mline* symbol near the top of the schematic.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

5. Choose *Edit—Properties—Objects* in the Schematic window to open the Edit Object Properties form for the selected *mline* transmission line component.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

6. The Edit—Properties—Objects form for *mline* component appears.

OK
Cancel
Apply
Defaults
Previous
Next
Help

Apply To    only current ...
instance ...

Show         system    user    CDF

Browse
Reset Instance Labels Display

Property	Value	Display
Library Name	<input style="width: 80%;" type="text" value="rfExamples"/>	<input type="checkbox"/> off
Cell Name	<input style="width: 80%;" type="text" value="mline"/>	<input type="checkbox"/> off
View Name	<input style="width: 80%;" type="text" value="symbol"/>	<input type="checkbox"/> off
Instance Name	<input style="width: 80%;" type="text" value="I0"/>	<input type="checkbox"/> off

Add
Delete
Modify

CDF Parameter	Value	Display
Number of Conductors	<input style="width: 80%;" type="text" value="2"/>	<input type="checkbox"/> off
use external model file	<input type="checkbox"/>	<input type="checkbox"/> off
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	<input type="checkbox"/> off
er: permittivity	<input style="width: 80%;" type="text" value="12.9"/>	<input type="checkbox"/> off
d: dielectric thickness (m)	<input style="width: 80%;" type="text" value="100e-6"/>	<input type="checkbox"/> off
w: conductor widths (m)	<input style="width: 80%;" type="text" value="300e-6"/>	<input type="checkbox"/> off
cd: conductor distances (m)	<input style="width: 80%;" type="text" value="50e-6"/>	<input type="checkbox"/> off
t: conductor thickness (m)	<input style="width: 80%;" type="text" value="5e-6"/>	<input type="checkbox"/> off
len: conductor length (m)	<input style="width: 80%;" type="text" value="8.47e-3"/>	<input type="checkbox"/> off
sigma: signal conductivity	<input style="width: 80%;" type="text" value="5.6e7"/>	<input type="checkbox"/> off
model type	<input checked="" type="radio"/> lossless <input type="radio"/> lossy	<input type="checkbox"/> off
freq: max frequency (Hz)	<input style="width: 80%;" type="text" value="40e9"/>	<input type="checkbox"/> off

7. In the CDF Parameters section at the bottom of the Edit Object Properties form, edit the CDF parameter values for the *mtline* component.

The edited CDF parameters look similar to the following:

CDF Parameter	Value
Number of Conductors	3
use external model file	<input type="checkbox"/>
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline
er: permittivity	12.9
d: dielectric thickness (m)	200e-6
w: conductor widths (m)	300e-6
cd: conductor distances (m)	150e-6
t: conductor thickness (m)	10e-6
len: conductor length (m)	5e-3
sigma: signal conductivity	5.6e7
model type	<input checked="" type="radio"/> lossless <input type="radio"/> lossy
freq: max frequency (Hz)	40e9

8. In the Edit Object Properties form, click *OK*.
9. Close the *mlineoscRFImg* schematic.

## Looking at *mtline* in the Analog Design Environment

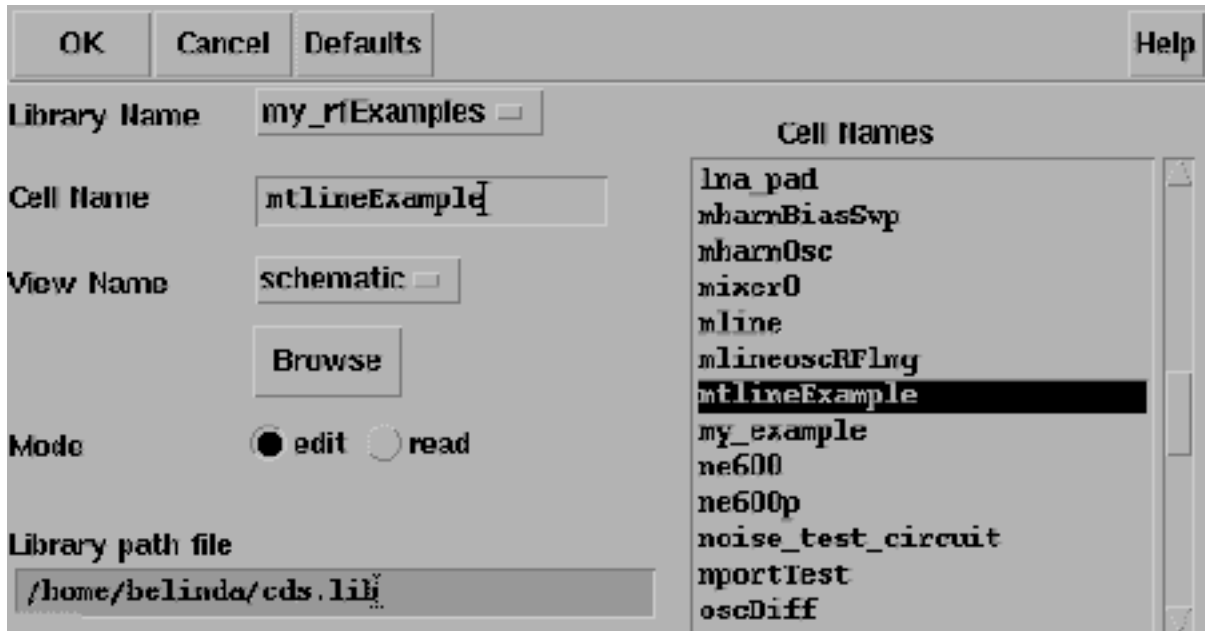
This example illustrates *mtline* parameters viewed in the analog design environment.

This example uses the *mtlineExample* circuit from your editable copy of the *rfExamples* library, *my\_rfExamples*. This library also includes other sample circuits used in this manual. If you need assistance setting up or accessing the *my\_rfExamples* library, refer to [Chapter 3, "Setting Up for the Examples."](#)

### Opening the `mtlineExample` Circuit in the Schematic Window

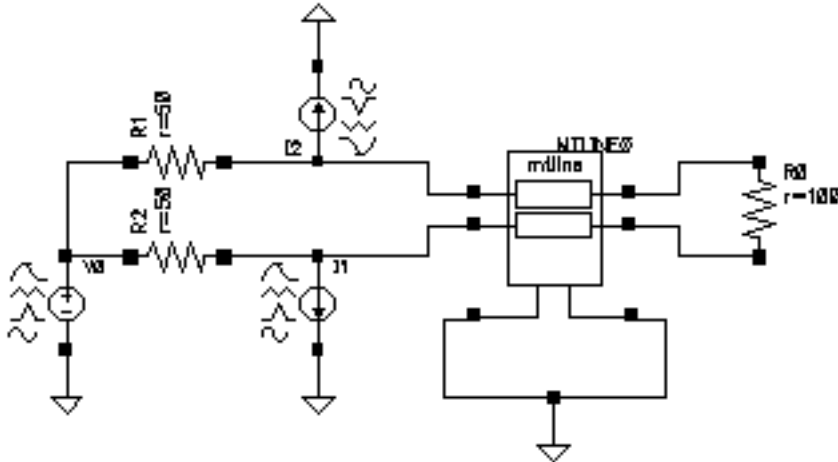
1. In the CIW, choose *File—Open*.

The Open File form appears.



2. In the Open file form, make the following selections.
3. In the *Library Name* cyclic field choose *my\_rfExamples*, your editable copy of the *rfExamples* library.
4. In the *Cell Names* list choose *mtlineExample*.
5. In the *View Name* cyclic field choose *schematic*.
6. For *Mode* highlight *edit*.
7. Click *OK*.

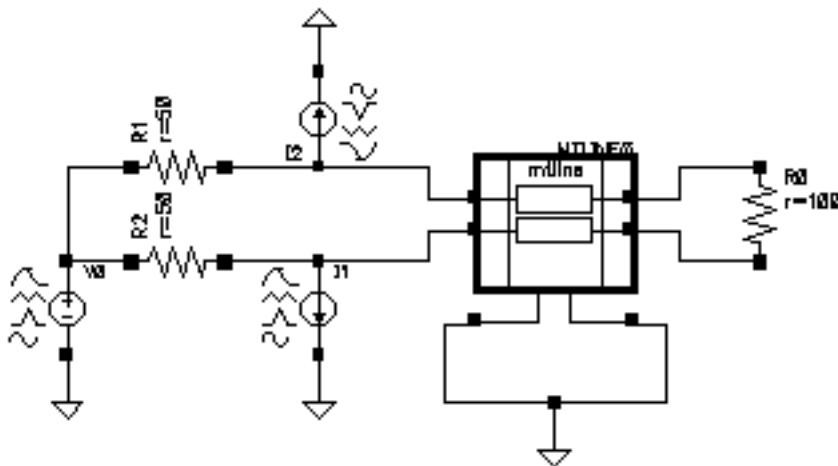
The *mtlineExample* schematic displays.



### Examining the mtline Component

1. In the Schematic window, select the transmission line component *mtline* at the center of the schematic.

The transmission line component is now highlighted.



2. With the transmission line selected in the Schematic window, choose *Edit – Properties – Objects* to open the Edit Object Properties form for the *mtline* component.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The CDF Parameter section of the *Edit – Properties – Objects* form for the *mtline* transmission line model appears below.

CDF Parameter	Value
Num of lines (excluding ref.)	2
Physical length	76.2m
Multiplicity factor	1
Max signal frequency	
Type of Input	<input checked="" type="radio"/> RLGC <input type="radio"/> FieldSolver <input type="radio"/> Tline
RLCG data file	rlgc.dat
use lmg subckt	<input type="checkbox"/>
Enter RLCG etc. matrices	<input checked="" type="checkbox"/>
R matrix per unit length	
L matrix per unit length	
G matrix per unit length	
C matrix per unit length	0 -1.9226e-11 1.213e-10
Skin effect res matrix per uni	
Dielectric loss cond matrix pe	
Frequency scale factor	

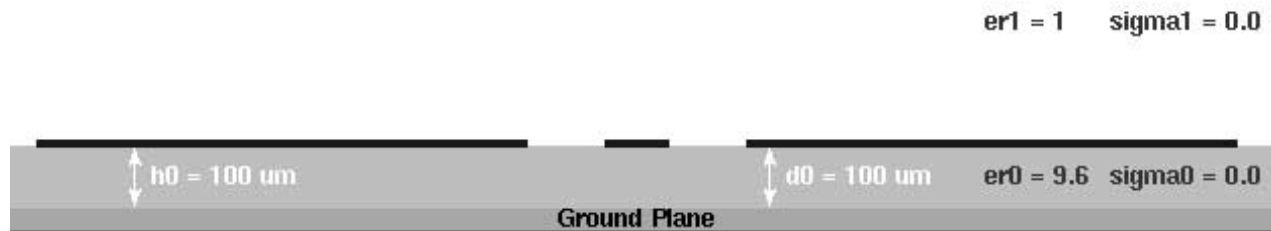
For the *Type of Input* parameter you can choose

- RLGC* - Uses the RLGC matrix
- FieldSolver* - Uses the built in 2-D field solver
- Tline* - Uses tline characteristics

When you select *FieldSolver*, all LMG properties are displayed.

## Coplanar Waveguide Modeling and Analysis

A single-signal coplanar waveguide is a transmission line made up of three conductors separated by gaps and lying on the surface of a dielectric substrate.



From left to right, the first and third conductors are ground planes which are 8 to 12 times wider than the middle conductor, which is the signal line. When the signal line in the middle is too narrow, the first and third conductors are 5 to 7 times the dielectric thickness of the narrow signal line in the center.

The characteristic impedance ( $Z_c$ ) of the CPW is a function of the ratio of the width of the gap between the signal line and the ground planes to the thickness of the dielectric substrate and its dielectric constant.

The first example, [Using LMG to Obtain Subcircuit Macromodel and LRCG Files](#), creates a CPW lossy, narrow-band model with one signal line. The signal line is 100 um wide with 130 um gaps on either side between the signal line and the 800 um wide ground planes. The characteristic impedance of the CPW is  $Z_c=50$  Ohm.

### Using LMG to Obtain Subcircuit Macromodel and LRCG Files

When you use LMG to model and analyze coplanar waveguide (CPW) transmission lines, you begin by verifying the content of the `.initlmg` file. You then open the LMG GUI, select the CPW transmission line type, and LMG extracts parameters and generates a CPW model.

### Verifying and Editing the Content of the `initlmg` File

Many features of the CPW model are displayed in both the `.initlmg` file illustrated in [Example 4-1](#) and in the LMG GUI displayed in [Figure 4-1](#) on page 240.

1. In the directory where you plan to invoke LMG, use the text editor of your choice to create the `.initlmg` file shown in [Example 4-1](#).

### Example 4-1 Sample `initlmg` file for a Coplanar Waveguide

```
lmgLineType = CPW
lmgModelType = LossyNarrow
lmgSubcktFormat = 0
lmgNumLayers = 2
lmgNumLines = 3
lmgNumGndPlanes = 1
lmgLengthUnit = um
lmgFreqUnit = GHz
lmgDielectricPermittivity = 9.6 1
lmgDielectricThickness = 100 400
lmgLossSigmaLossTang = VSIGMA
lmgDielectricConductivity = 0.0
lmgGndThickness = 20
lmgGndSigma = 5.6e7
lmgConductorWidth = 800 100 800
lmgConductorGaps = 130
lmgConductorThickness = 10
lmgConductorHeight = 100
lmgConductorLength = 2000
_lmgConductivity = 5.6e7
lmgMaxFreq = 1
lmgSubCircuitName = cpw1
lmgLRCHandle = w_linecpw1.dat
lmgOutputID = BOTH
```

2. Save the edited `.initlmg` file in the directory where you plan to invoke LMG.
3. The edited `.initlmg` file defines the properties of the CPW. Once you open the LMG GUI, you can modify or define the properties of the CPW by entering values in the GUI. When you exit LMG, you can save the values in the LMG GUI to create a new `.initlmg` file.

### Opening the LMG GUI

1. From the directory where you created and saved the `.initlmg` file, invoke the LMG GUI from the UNIX command line by typing:

```
% lmg &
```

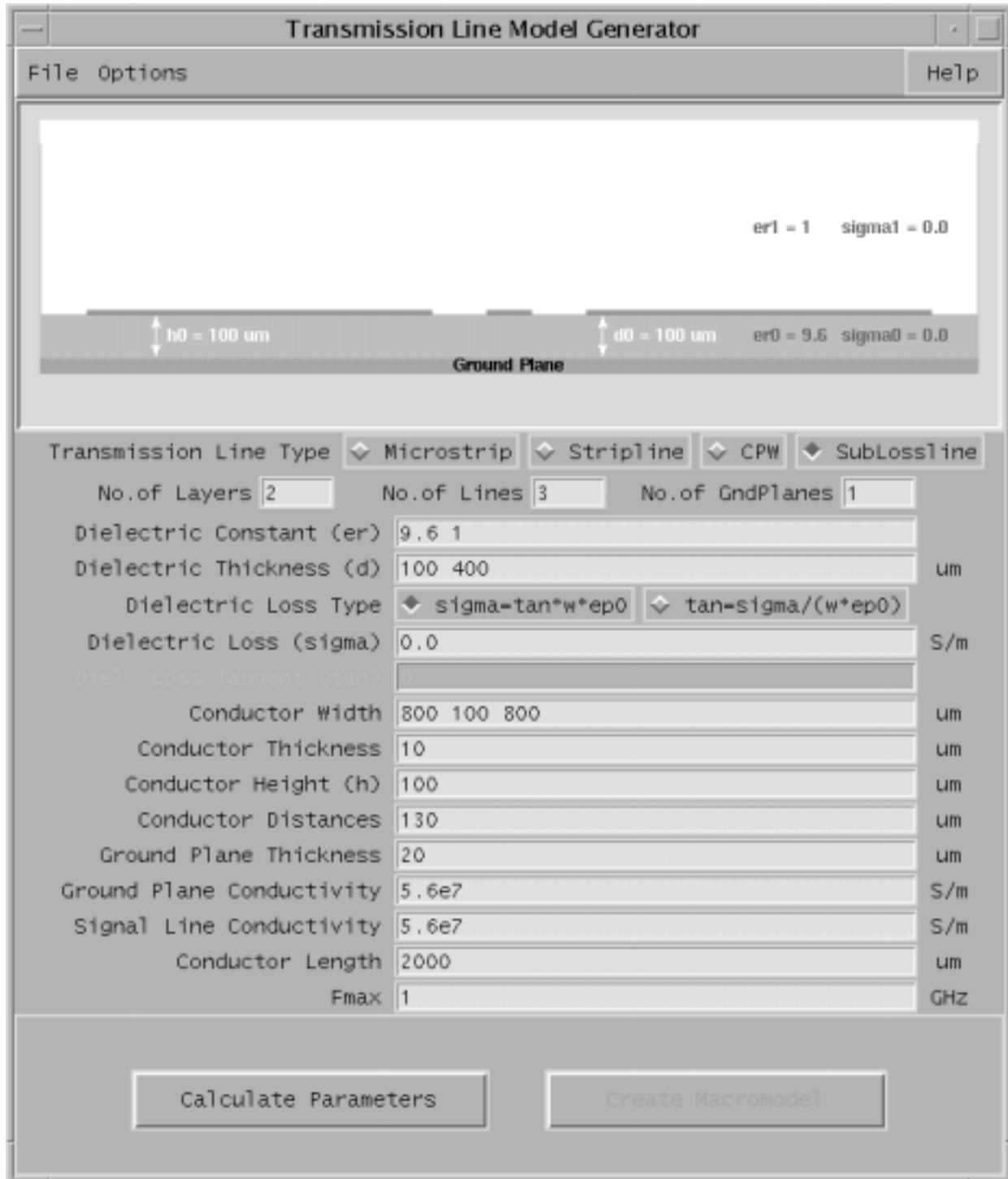
**Note:** When you start LMG from the UNIX prompt, LMG saves the model files in the directory where you start it. The LMG GUI displays directly without first displaying the Transmission Line Modeler form which shows the path to the model directory (the directory where LMG saves model files).

The LMG GUI opens reflecting the CPW parameter values from the `.initlmg` file.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmission Lines

Figure 4-1 LMG UI set up to Model a Coplanar Waveguide



Notice the following *File* menu choices made in the .initlmg file.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

<b>initlmg File Entry</b>	<b>File Menu Choice</b>
<code>lmgSubCircuitName = cpw1</code>	File—Subcircuit Name
<code>lmgLRCGName = w_linecpw1.dat</code>	File—LRCG file Name

Notice the following *Options* menu choices made in the `.initlmg` file.

<b>initlmg File Entry</b>	<b>Options Menu Choice</b>
<code>lmgModelType = LossyNarrow</code>	Options—Model Type—Lossy, Narrow Band
<code>lmgSubcktFormat = 0</code>	Options—Subckt Format—Spectre
<code>lmgLengthUnit = LossyNarrow</code>	Options—Length Unit—um
<code>lmgFreqUnit = LossyNarrow</code>	Options—Freq. Unit—GHz

Notice the following GUI field choices made in the `.initlmg` file and displayed in the LMG GUI.

<b>initlmg File Entry</b>	<b>GUI Choice</b>
<code>lmgLineType = CPW</code>	Transmission Line Type is CPW.
<code>lmgNumLines = 3</code>	Number of Lines is 3. The 2 wide ground planes and the 1 narrow signal line.
<code>lmgDielectricPermittivity = 9.6 1</code>	Dielectric Constants are 9.6 and 1. The values are separated by a space.
<code>lmgConductorWidth = 800 100 800</code>	Conductor Widths are 800, 100 and 800 um. The values are separated by a space.
<code>lmgConductorGaps = 130</code>	Conductor Distances are both 130 um.
<code>lmgConductorLength = 2000</code>	Conductor length is 2000 um.

Whenever you enter multiple values in the LMG GUI for a parameter that accepts multiple values, separate the values with a space.

The Conductor Width values have a ratio of 8 to 1. Usually the larger the ratio between the two numbers, the more accurate the result but the slower the speed.

Since both Conductor Distances are the same, only one entry is required.

### Generating the Subcircuit and LRCG Files

1. To enter a name for the subcircuit file, choose *File—Subcircuit Name*.

The Subcircuit Name form appears.



The subcircuit name, CPW1, from the `initlmg` file appears in the *Subcircuit Name* field.

2. Click *OK*.

This name gives information about the type of model subcircuit created. The suffix `scs` is attached to the subcircuit model file.

3. To enter a name for the LRCG file, choose *File—LRCG file Name*.

The LRCG File Name form appears.



The LRCG file name, `w_linecpw1.dat` from the `initlmg` file appears in the *LRCG Name* field.

4. Click *OK*.

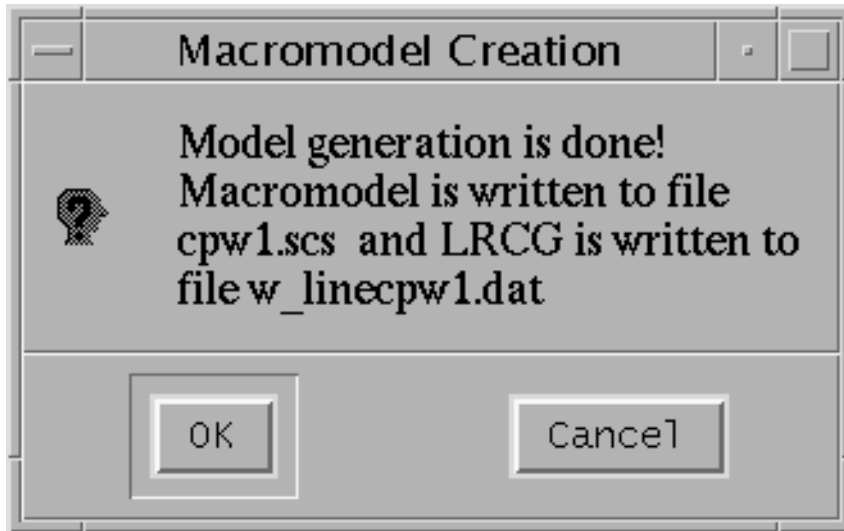
This name gives information about the type of model this LRCG file is associated with.

5. Click *Calculate Parameters* at the bottom of the LMG GUI.

The *Create Macromodel* button is active after the parameters are calculated.

6. Click *Create Macromodel* at the bottom of the LMG GUI to create the CPW macromodel netlist file and the LRCG data file.

The Macromodel Creation form displays the names of the two final result files.



The two result files are created in the directory where you started LMG:

- An LRCG file with the name `w_linecpw1.dat`. (See [step 7](#)).
- A macromodel netlist file with the name `cpw1.scs`. (See [“Resolving a Possible Error Message in the mtline Model File”](#) on page 244).

**Note:** Be sure to record the absolute path to the subckt model file `cpw1.scs` and the LRCG file `w_linecpw1.dat`. You need the absolute path and filenames when you use the files in CPW simulations later in this section.

7. Click *OK* in the Macromodel Creation form.

The contents of the LRCG (`w_linecpw1.dat`) file are

```
; RLCG Matrices produced by LMG
; It's Coplanar waveguide (CPW) line, first and last lines are regarded as
ground
; Size Reduction is done automatically, reference FAQs in the Documents for
details
; The Inputs are:
; lmgLineType = CPW
; lmgModelType = LossyNarrow
; lmgSubcktFormat = 0
; lmgNumLayers = 2
; lmgNumLines = 3
; lmgNumGndPlanes = 1
; lmgLengthUnit = um
; lmgFreqUnit = GHz
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

```
; lmgDielectricPermittivity = 9.6 1
; lmgDielectricThickness = 100 400
; lmgLossSigmaLossTang = VSIGMA
; lmgDielectricConductivity = 0.0
; lmgGndThickness = 20
; lmgGndSigma = 5.6e7
; lmgConductorWidth = 800 100 800
; lmgConductorGaps = 130
; lmgConductorThickness = 10
; lmgConductorHeight = 100
; lmgConductorLength = 2000
; _lmgConductivity = 5.6e7
; lmgMaxFreq = 1
; lmgSubCircuitName = cpw1
; lmgLRCGName = w_linecpw1.dat
; lmgOutputID = BOTH
```

```
FORMAT Freq:  L1:1
               R1:1
               C1:1
               G1:1
```

```
1.000000e+09 : 3.967294e-07
                6.935151e+01
                1.574700e-10
                0.000000e+00
```

#### 8. Choose *File—Quit* to exit LMG. Click *OK* in the Quit form.

You use the LRCG file `w_linecpw1.dat` and the subckt model file `cpw1.scs` you just created in different simulations later in this section.

## Resolving a Possible Error Message in the mtline Model File

In some circumstances the analog design environment issues an error message for the `cpw1.scs` subcircuit file.

To solve this problem, you can open and edit the subcircuit file with the text editor of your choice. The unedited `cpw1.scs` file looks like the following.

```
//
// quadrature model to transmission line.
// Transmission line type: CPW
// Number of signal conductors: 1
subckt cpw1 (in0 out0 ref)
  10_0 (in0 n1_0) inductor l=3.722123e-11 r=6.506573e-03
  11_0 (n1_0 n2_0) inductor l=1.458814e-10 r=2.550124e-02
  12_0 (n2_0 n3_0) inductor l=2.136269e-10 r=3.734371e-02
  13_0 (n3_0 n4_0) inductor l=2.136269e-10 r=3.734371e-02
  14_0 (n4_0 n5_0) inductor l=1.458814e-10 r=2.550124e-02
  15_0 (n5_0 out0) inductor l=3.722123e-11 r=6.506573e-03
  c0_0 (n1_0 ref) capacitor c=3.730874e-14
  c1_0 (n2_0 ref) capacitor c=7.536955e-14
  c2_0 (n3_0 ref) capacitor c=8.958279e-14
  c3_0 (n4_0 ref) capacitor c=7.536955e-14
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

```
c4_0 (n5_0 ref) capacitor c=3.730874e-14
ends cpw1
```

To edit the `cpw1.scs` subcircuit file with the text editor, use the following steps.

1. Find the following text in the `cpw1.scs` file:

```
(in0 out0 ref)
```

2. Replace it with the following text.

```
(in0 out0 gnd refgnd)
```

The edited contents of the `cpw1.scs` file looks like this.

```
//
// quadrature model to transmission line.
// Transmission line type: CPW
// Number of signal conductors: 1
subckt cpw1 (in0 out0 gnd refgnd)
  l0_0 (in0 n1_0) inductor l=3.722123e-11 r=6.506573e-03
  l1_0 (n1_0 n2_0) inductor l=1.458814e-10 r=2.550124e-02
  l2_0 (n2_0 n3_0) inductor l=2.136269e-10 r=3.734371e-02
  l3_0 (n3_0 n4_0) inductor l=2.136269e-10 r=3.734371e-02
  l4_0 (n4_0 n5_0) inductor l=1.458814e-10 r=2.550124e-02
  l5_0 (n5_0 out0) inductor l=3.722123e-11 r=6.506573e-03
  c0_0 (n1_0 ref) capacitor c=3.730874e-14
  c1_0 (n2_0 ref) capacitor c=7.536955e-14
  c2_0 (n3_0 ref) capacitor c=8.958279e-14
  c3_0 (n4_0 ref) capacitor c=7.536955e-14
  c4_0 (n5_0 ref) capacitor c=3.730874e-14
ends cpw1
```

3. Save the edited `cpw1.scs` file.

After you edit and save the `cpw1.scs` file, you can perform a simulation using the edited copy of the `cpw1.scs` file in [“Simulating Coplanar Waveguides with the Generated Subcircuit Macromodel File”](#) on page 273.

## General Theory of Coplanar Waveguide Analysis

You can see from [Example 4-1](#) on page 243 that the size of the LRCG matrixes in file `w_linecpw1.dat` is 1 \* 1 matrix even though the *No. of Lines* value in the LMG GUI is 3. This happens because you selected *CPW* for *Transmission Line Type*. For a single-line coplanar waveguide system, as shown in [Figure 4-2](#) on page 246, the first and last lines are regarded as ground planes and the single, narrow line in the middle is regarded as the signal line.

Generally, a strict coplanar analysis is quite complicated and the parameter extraction requires information about the equivalent magnetic source solver. LMG does not perform this type of analysis. Rather, LMG first processes a coplanar waveguide as a system of regular transmission lines and then performs the size reduction.

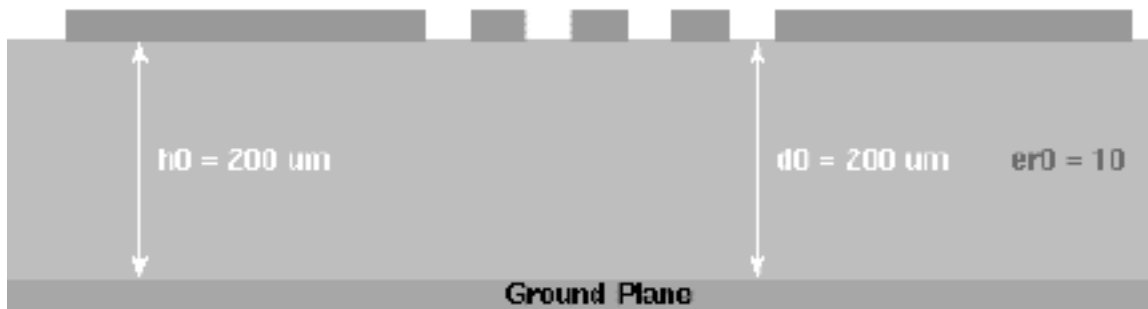
When processing the single-line CPW system shown in [Figure 4-2](#) on page 246, LMG first treats this CPW as a regular 3-line system. It then sets the voltages of the first and third lines to zero ( $v_1=0$  and  $v_3=0$ ) and performs the parameter extraction and model generation.

**Figure 4-2 Single Line Coplanar Waveguide System**



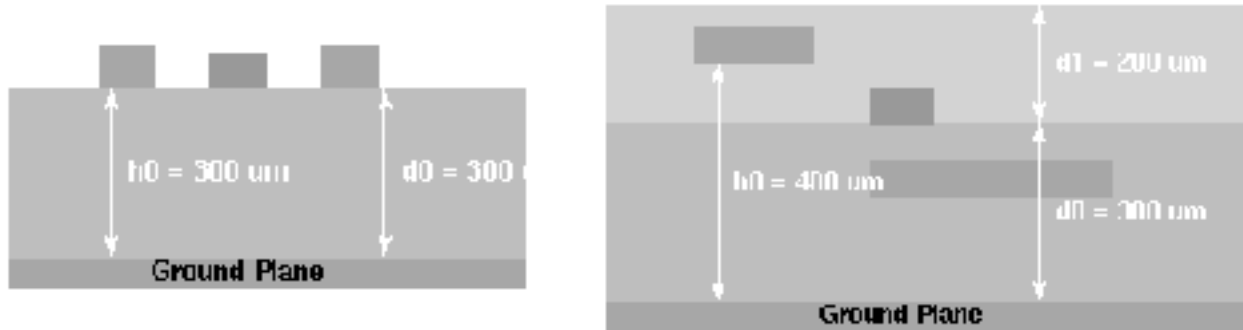
LMG can also handle a multi-line coplanar waveguide system such as the one shown in [Figure 4-3](#).

**Figure 4-3 Multi-Line Coplanar Waveguide System**



In a general sense, you can use a CPW transmission line to process various side-shielding lines such as the two side-shielding line systems shown in [Figure 4-3](#).

### Side-Shielding Lines Handled by the CPW Line Type



In the example systems shown in Figure 4-3, the first and last lines are always considered to be grounded. In the LMG GUI, they display in the ground plane color.

In principle, for a CPW, an  $n$ -line system produces  $m = n - 2$  order modeling. This means that the LRCG file size is  $m$  times  $m$ . LMG performs the size reduction automatically and bases lumped models on the reduced-size LRCG file.

## Simulating Coplanar Waveguides with the Generated LRCG File

To simulate a Coplanar Waveguide, or CPW, use the LRCG file, `w_linecpw1.dat`, you created, in Section [“Using LMG to Obtain Subcircuit Macromodel and LRCG Files”](#) on page 238. Then add the LRCG file to the `mtline` macromodel in the `CPW_tlineSimple` schematic. To do this you need the absolute path to the LRCG file you noted in Section [“Generating the Subcircuit and LRCG Files”](#) on page 242.

This example uses the circuit `CPW_tlineSimple` from your editable copy of the `rfExamples` library (`my_rfExamples` here). This library also includes other sample circuits used in this manual. If you need assistance setting up or accessing your editable copy of the `rfExamples` library, refer to [Chapter 3, “Setting Up for the Examples.”](#)

### Opening the CPW\_tlineSimple Circuit in the Schematic Window

1. In the CIW, choose *File – Open*.

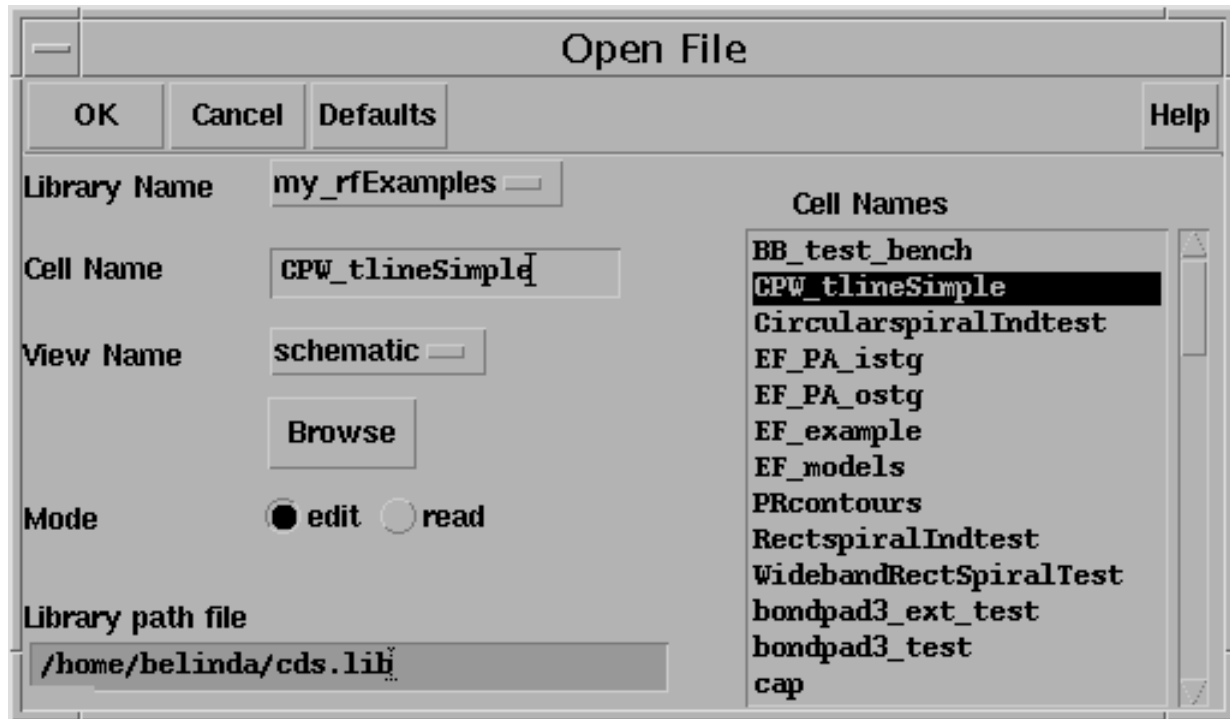
The Open File form appears.

2. In the Open File form, choose `my_rfExamples` in the *Library Name* cyclic field and choose `CPW_tlineSimple` in the *Cell Names* list box.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The completed Open File form appears like the one below.



3. Click **OK**.

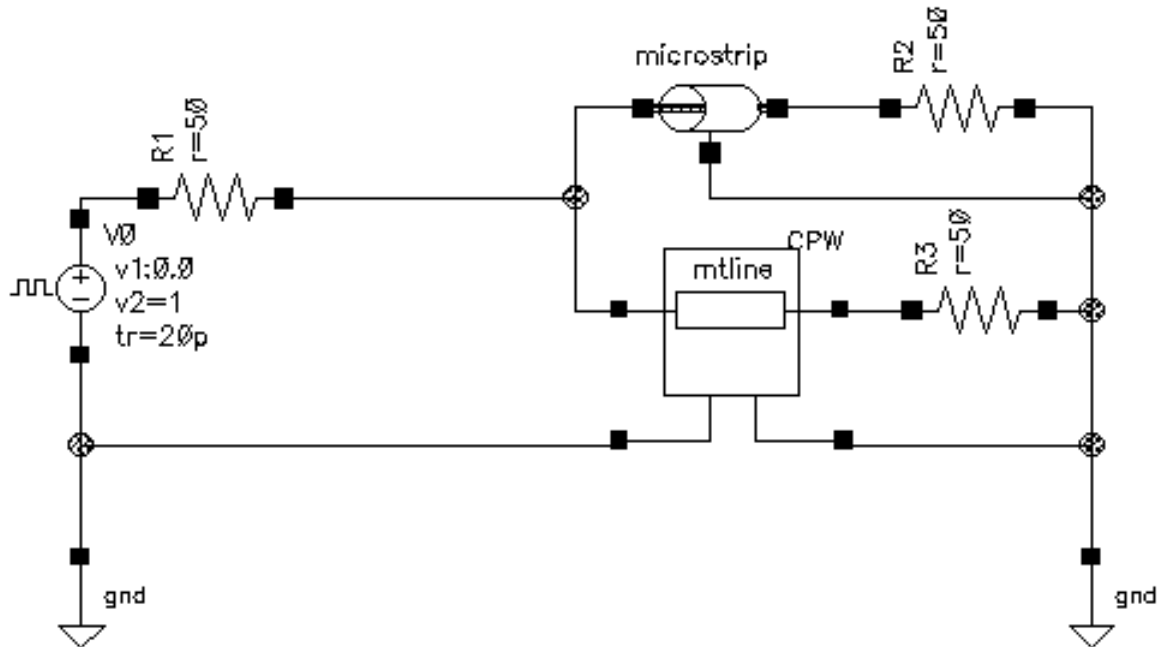


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The Schematic window for the CPW\_tlineSimple circuit appears.

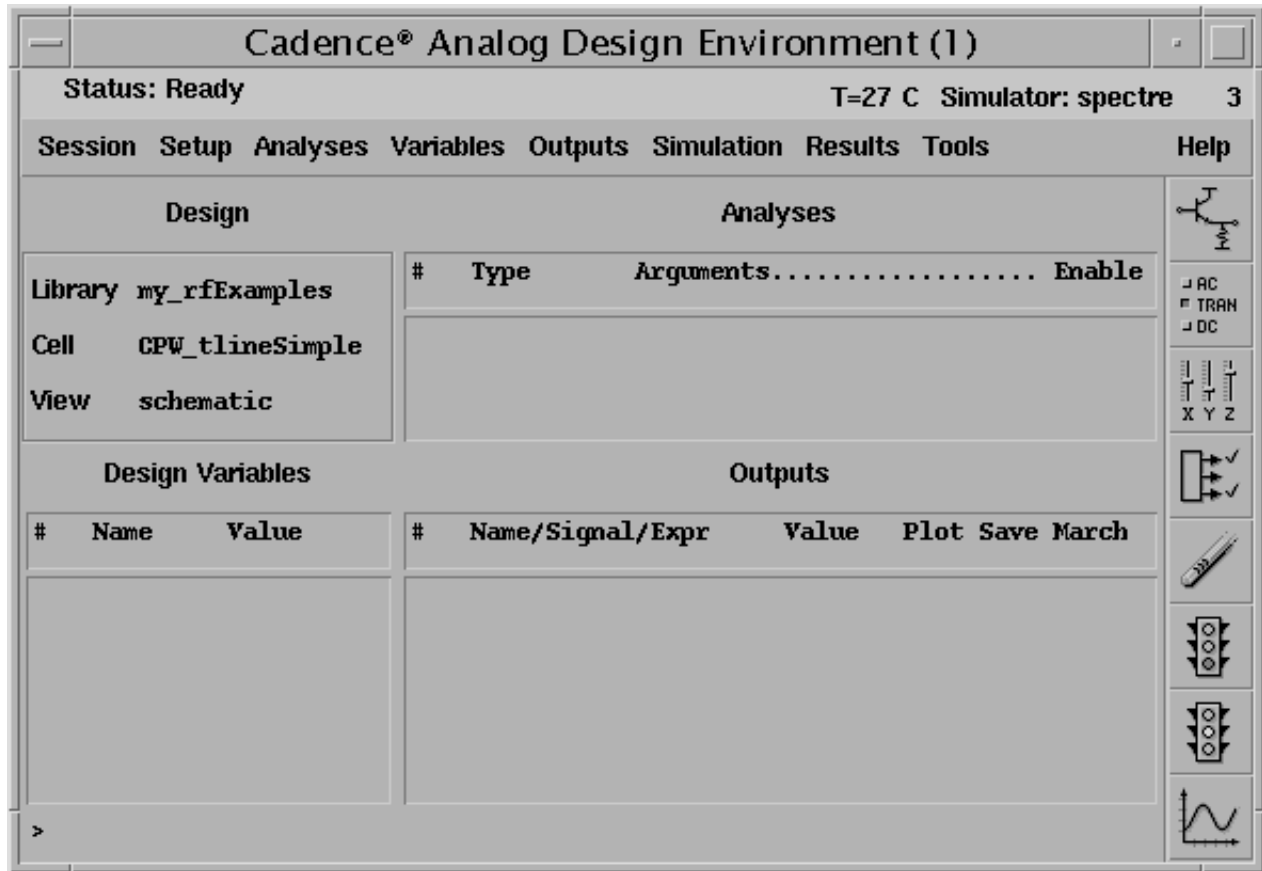


4. In the Schematic window, choose *Tools—Analog Environment*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmission Lines

The Simulation window for the CPW\_tlineSimple circuit opens.



### Viewing the Properties of the Microstrip Component

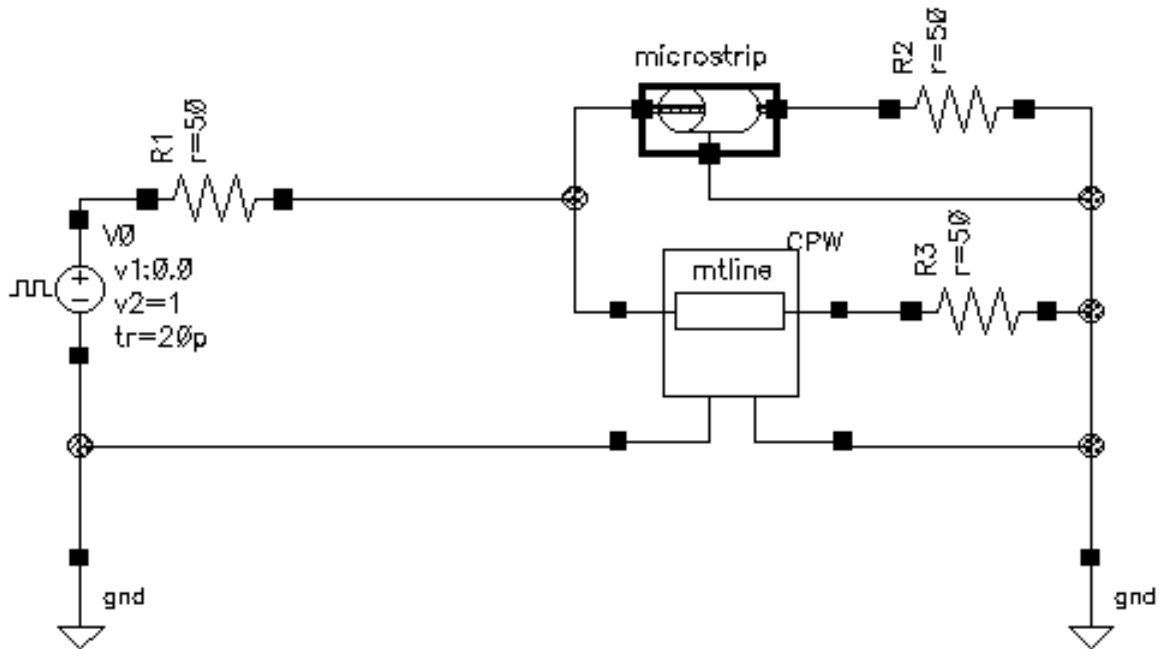
1. In the Schematic window, click the `tline3` component symbol labeled *microstrip* located at the top of the schematic.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The *microstrip* symbol is now selected.



2. With the *microstrip* (`tline3`) transmission line symbol selected in the Schematic window, choose *Edit—Properties—Objects* to open the Edit Object Properties form for the *microstrip*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The *Edit—Properties—Objects* form for the *microstrip* appears similar to the following.

**Edit Object Properties**

OK
Cancel
Apply
Defaults
Previous
Next
Help

Apply To   

Show         system  user  CDF

Browse
Reset Instance Labels Display

Property	Value	Display
Library Name	<input type="text" value="rfExamples"/>	<input type="checkbox"/> off
Cell Name	<input type="text" value="tline3"/>	<input type="checkbox"/> off
View Name	<input type="text" value="symbol"/>	<input type="checkbox"/> off
Instance Name	<input type="text" value="microstrip"/>	<input type="checkbox"/> off

Add
Delete
Modify

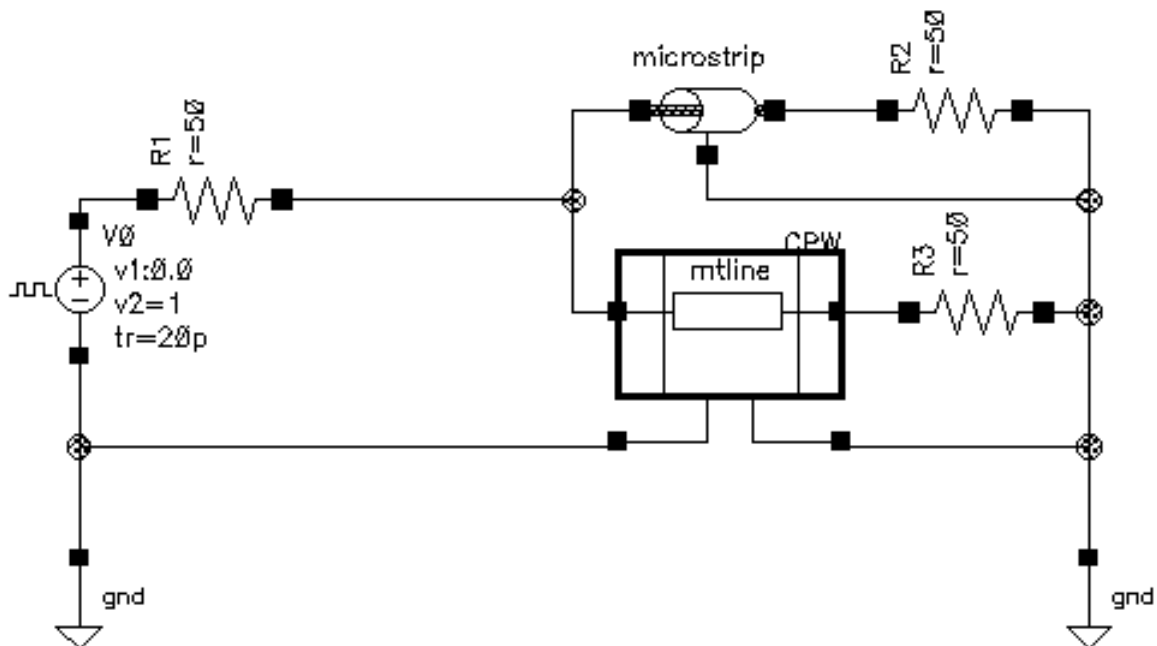
CDF Parameter	Value	Display
use external model file	<input type="checkbox"/>	<input type="checkbox"/> off
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	<input type="checkbox"/> off
er: permittivity	<input type="text" value="9.6"/>	<input type="checkbox"/> off
d: dielectric thickness (m)	<input type="text" value="100u"/>	<input type="checkbox"/> off
w: conductor width (m)	<input type="text" value="100u"/>	<input type="checkbox"/> off
t: conductor thickness (m)	<input type="text" value="10u"/>	<input type="checkbox"/> off
len: conductor length (m)	<input type="text" value="2000u"/>	<input type="checkbox"/> off
model type	<input type="radio"/> lossless <input checked="" type="radio"/> lossy	<input type="checkbox"/> off
lossy type	<input checked="" type="radio"/> narrow band <input type="radio"/> wide band	<input type="checkbox"/> off
sigma: conductivity	<input type="text" value="5.6e7"/>	<input type="checkbox"/> off
freq: max frequency (Hz)	<input type="text" value="1G"/>	<input type="checkbox"/> off

3. In the Edit Object Properties form for the *microstrip*, click *OK*.

### Associating the LRCG File with the Mline Component

1. In the Schematic window, select the *CPW* (*mtline*) transmission line symbol at the center of the schematic.

The *CPW* symbol is now selected.



2. With the *CPW* symbol selected, choose *Edit—Properties—Objects* to open the Edit Object Properties form for the *CPW*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The CDF parameter section of the CPW's Edit Object Properties form appears similar to the following.

CDF Parameter	Value	Display
Num of lines (excluding ref.)	1	off <input type="checkbox"/>
RLCG data file	/home/belinda/w_linecpw	off <input type="checkbox"/>
use lmg subckt	<input type="checkbox"/>	off <input type="checkbox"/>
Invoke 'LMG' parameter extraction tool		
Physical length	2.0000m	off <input type="checkbox"/>
Enter RLCG etc. matrices	<input type="checkbox"/>	off <input type="checkbox"/>
Frequency scale factor		off <input type="checkbox"/>
ROM data file		off <input type="checkbox"/>
Multiplicity factor	1	off <input type="checkbox"/>

3. Edit the *CDF Parameter* data field value for *RLCG data file* by replacing the current contents of the field with the full, absolute path to the LRCG data file, `w_linecpw1.dat`, you generated with LMG in [“Generating the Subcircuit and LRCG Files”](#) on page 242.

For example, enter `/home/belinda/w_linecpw1.dat`

CDF Parameter	Value	Display
Num of lines (excluding ref.)	1	off <input type="checkbox"/>
RLCG data file	/belinda/w_linecpw1.dat	off <input type="checkbox"/>
use lmg subckt	<input type="checkbox"/>	off <input type="checkbox"/>

4. In the *Edit—Properties—Objects* form, click *OK*.

When you click *OK*, the form closes and the `w_linecpw1.dat` file is saved in the schematic editor. When you simulate with the *CPW* component, the `w_linecpw1.dat` LRCG data file is used.

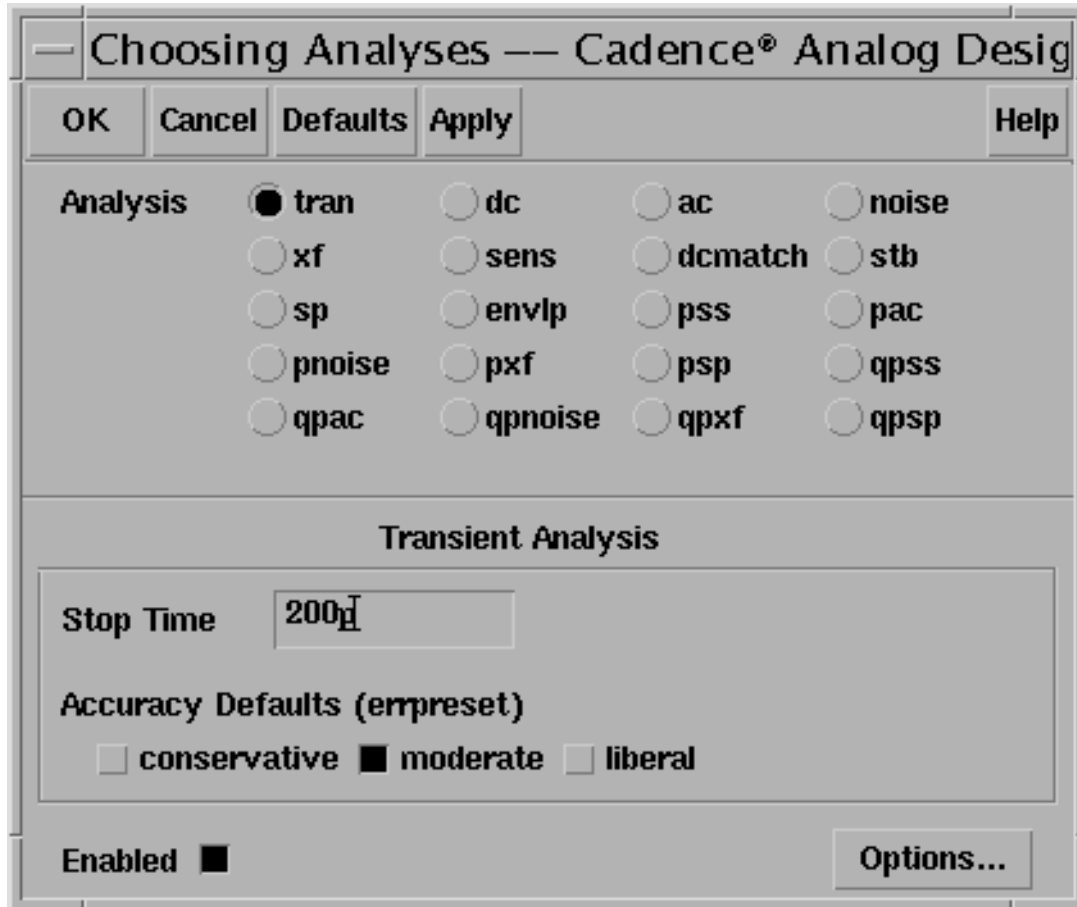
5. In the schematic window, choose *Design—Check and Save* for the `CPW_tlineSimple` schematic.

You can now set up and run a simulation using the new `mtline` macromodel for the *CPW* in the analog design environment.

### Simulating the CPW

1. In the Simulation window, choose *Analyses—Choose*.  
The Choosing Analyses form appears.
2. In the Choosing Analyses form, click *tran*.  
The form displays options needed for transient analysis.
3. In the *Stop Time* field, type your best estimate.  
Type `200p` in this example.
4. Highlight *moderate* for the *Accuracy Defaults (errpreset)*.
5. Verify that *Enabled* is highlighted.

The Choosing Analyses form appears below.



6. Click *Apply* to check for setup errors for the Transient analysis.
7. Click *OK*.

The *Transient* analysis options you choose appear in the *Analyses* list box in the Simulation window.

Analyses				
#	Type	Arguments.....		Enable
1	tran	0	200p	yes



### **Running the Simulation**

1. In the Simulation window, choose *Simulation – Netlist and Run* to run the simulation.  
The output log file appears and displays information about the simulation as it runs.
2. Look in the CIW for a message that says the simulation completed successfully.

### **Plotting the Coplanar Waveguide Signals**

1. To open the Direct Plot form, choose *Results—Direct Plot—Main Form* in the Simulation window.  
The Waveform window and the Direct Plot form appear.
2. In the Direct Plot form do the following.
  - a. Highlight *Append* for *Plot Mode*.
  - b. Highlight *Tran* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The filled-in form looks like this.



The image shows a dialog box titled "Direct Plot Form". It contains several controls:

- Buttons: "OK", "Cancel", and "Help".
- Plot Mode: Radio buttons for "Append" (selected) and "Replace".
- Analysis: A list box containing "tran" (selected).
- Function: Radio buttons for "Voltage" (selected) and "Current".
- Select: A text box containing "Net".
- Prepend Waveform from Reference Directory: An unchecked checkbox.
- Add To Outputs: An unchecked checkbox.
- Bottom prompt: "> Select Net on schematic...".

3. Follow the prompt at the bottom of the Direct Plot form.

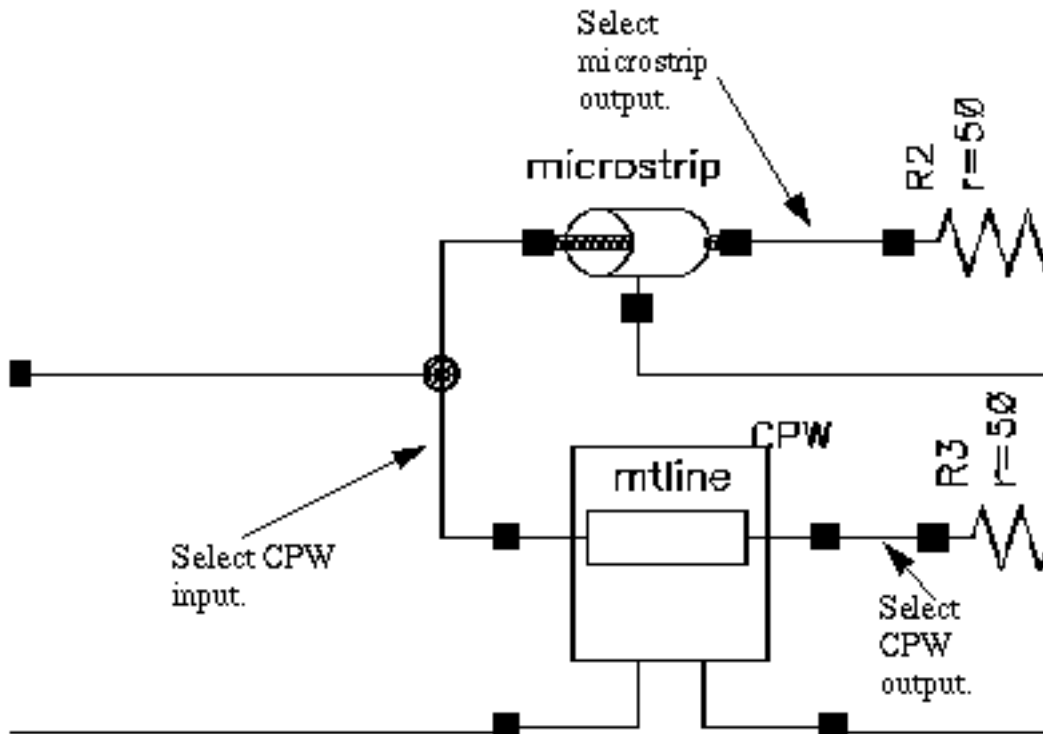
> Select Net on Schematic...

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

4. Select nets in the CPW\_tlineSimple schematic window.



- a. Click the CPW input.
  - b. Click the microstrip output.
  - c. Click the CPW output.
5. Press *Esc* to stop selecting nets.

The plot in Figure 4-4 appears in the Waveform window.

Figure 4-4 Signals Passing Through the CPW and the 200u Microstrip

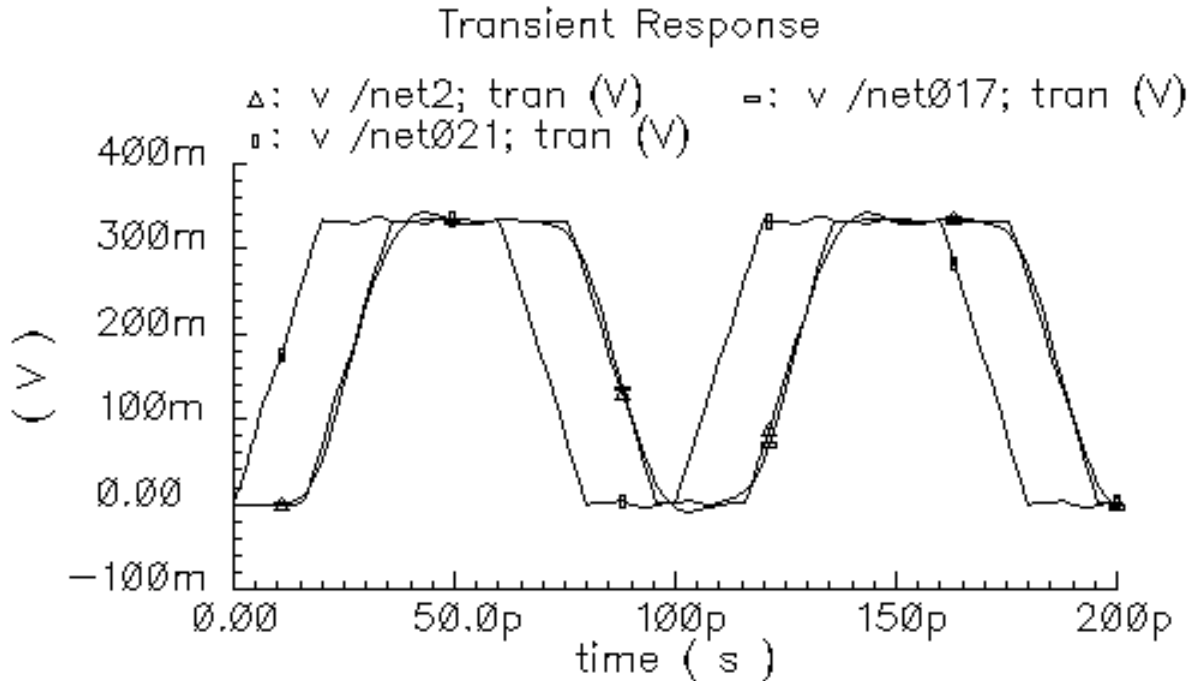


Figure 4-4 shows that frequency dispersion for the CPW is better than it is for the microstrip line. From Figure 4-4, the waveform shape of *net2* (the CPW output) is better than the shape of *net017* (the microstrip output). The phase delays for both lines are correct. You can also see that the input signal is slightly corrupted by the reflection from the microstrip line. This is evident from the ripple on *net021*.

### Verifying Input Signal Corruption by Microstrip Line Reflection

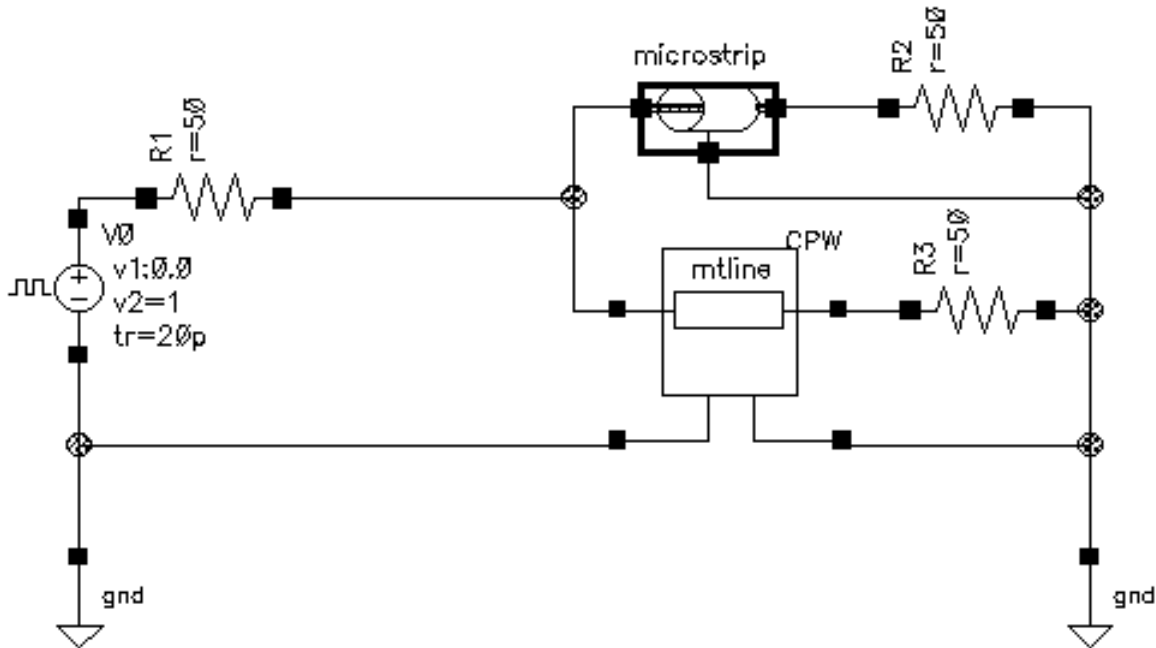
Use the following procedures to verify that the input signal is slightly corrupted by reflection from the microstrip line.

1. In [Shortening the Length of the Microstrip Component](#), decrease the microstrip conductor length from 200u to 10u.
2. In [Increasing the Length of the Microstrip Component](#) on page 266, increase the microstrip conductor length from 200u to 100000u.
3. In both cases, simulate and compare the transient responses in Figures 4-4, 4-5 and 4-6.

### Shortening the Length of the Microstrip Component

1. In the Schematic window, click the *microstrip* component symbol located at the top of the schematic.

The `tline3` symbol is now selected.



2. With the `tline3` transmission line symbol selected, choose *Edit—Properties—Objects* to open the Edit Object Properties form for the *microstrip*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The *CDF Parameter* section of the Edit Objects Properties form for the *microstrip* appears similar to the following.

CDF Parameter	Value	Display
use external model file	<input type="checkbox"/>	off <input type="checkbox"/>
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	off <input type="checkbox"/>
er: permittivity	9.6	off <input type="checkbox"/>
d: dielectric thickness (m)	100u	off <input type="checkbox"/>
w: conductor width (m)	100u	off <input type="checkbox"/>
t: conductor thickness (m)	10u	off <input type="checkbox"/>
len: conductor length (m)	10u	off <input type="checkbox"/>
model type	<input type="radio"/> lossless <input checked="" type="radio"/> lossy	off <input type="checkbox"/>
lossy type	<input checked="" type="radio"/> narrow band <input type="radio"/> wide band	off <input type="checkbox"/>
sigma: conductivity	5.6e7	off <input type="checkbox"/>
freq: max frequency (Hz)	1G	off <input type="checkbox"/>

3. Type 10u in the *len: conductor length (m)* field.
4. In the Edit Object Properties form for the *microstrip*, click *OK*.
5. In the schematic window, choose *Design—Check and Save* to check and save the CPW\_tlineSimple schematic.

You can now set up and run a simulation in the analog design environment using the new *tline3* macromodel line length for the *microstrip* component.

### Simulating the CPW

1. in the Simulation window, verify that the *transient* analysis is still enabled.

The transient analysis appear in the *Analyses* list box.

Analyses				
#	Type	Arguments.....		Enable
1	tran	0	200p	yes

### Running the Simulation

1. In the Simulation window, choose *Simulation—Netlist and Run* to run the simulation.  
The output log file appears and displays information about the simulation as it runs.
2. Look in the CIW for a message that says the simulation completed successfully.

### Plotting the Coplanar Waveguide Signals

1. To open the Direct Plot form, choose *Results—Direct Plot—Main Form* in the Simulation window.  
The Waveform window and the Direct Plot form appear.
2. In the Direct Plot form do the following.
  - a. Highlight *Append* for *Plot Mode*.
  - b. Highlight *Tran* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The filled-in form looks like this.



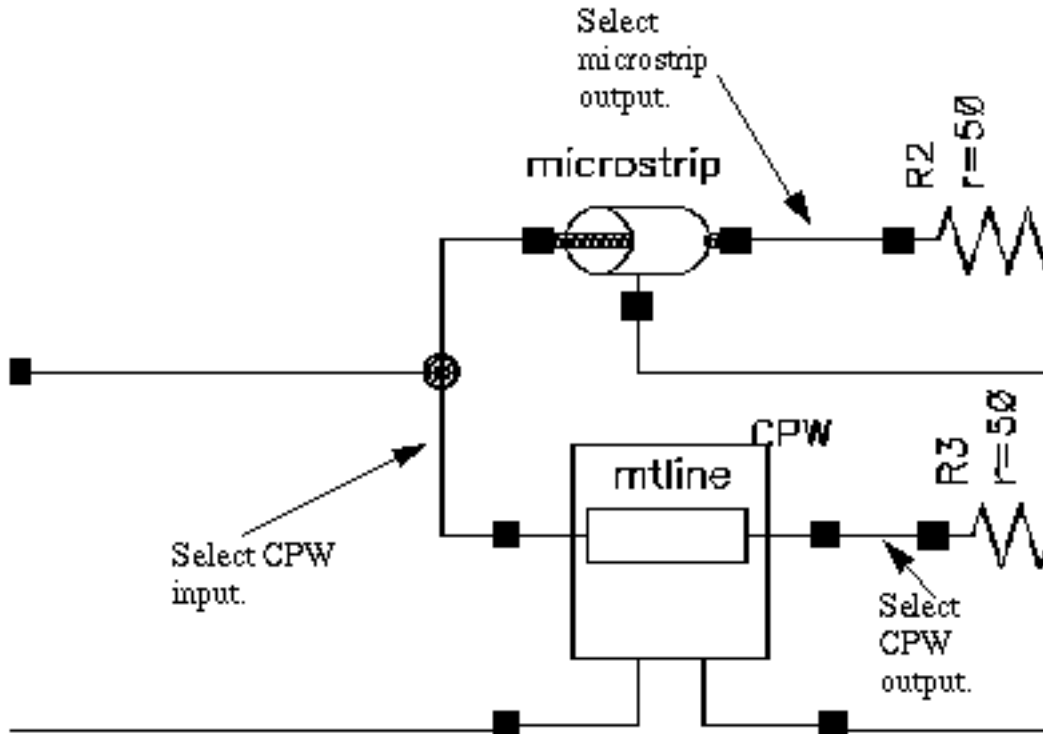
The image shows a dialog box titled "Direct Plot Form". It contains several controls: "OK", "Cancel", and "Help" buttons at the top. Below these are radio buttons for "Plot Mode" with "Append" selected and "Replace" unselected. The "Analysis" section has a radio button for "tran" selected. The "Function" section has radio buttons for "Voltage" selected and "Current" unselected. There is a "Select" label followed by a text box containing "Net". Below that are two checkboxes: "Prepend Waveform from Reference Directory" (unchecked) and "Add To Outputs" (unchecked). At the bottom, there is a prompt: "> Select Net on schematic...".

3. Follow the prompt at the bottom of the Direct Plot form.

> Select Net on Schematic...



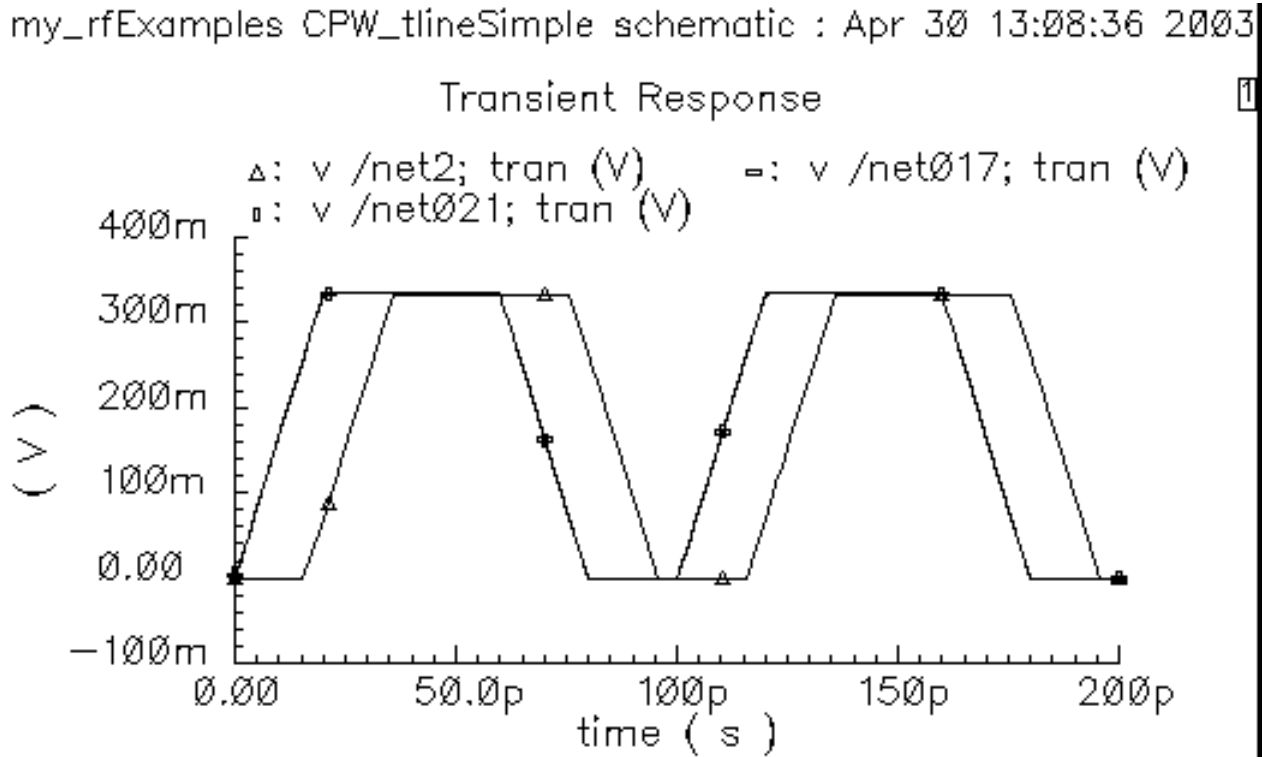
4. Select the following nets in the CPW\_tlineSimple schematic.window



- a. Click the CPW input.
  - b. Click the microstrip output.
  - c. Click the CPW output.
5. Press *Esc* to stop selecting nets.

The plot in Figure 4-4 appears in the Waveform window.

Figure 4-5 Signals Passing Through the CPW and the 10u Microstrip



When you compare [Figure 4-4](#) on page 260 with [Figure 4-5](#), you can see that the shapes in [Figure 4-5](#) are much cleaner than the shapes in [Figure 4-4](#).

In [Figure 4-5](#), the microstrip line is only 10  $\mu\text{m}$  long. Since it is so short, in fact almost a shunt interconnect between *net021* and *net017*, there is no reflection from the *microstrip* to corrupt the input signal. Because the *CPW* is a very low frequency dispersion line, the *CPW* output (*net2*) is almost identical to its input (*net021*) except for the phase shift.

### Increasing the Length of the Microstrip Component

1. In the Schematic window, click the *microstrip* component symbol located at the top of the schematic.
2. With the *tline3* transmission line symbol selected, choose *Edit—Properties—Objects* to open the Edit Object Properties form for the *microstrip*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The *CDF Parameter* section of the Edit Objects Properties form for the *microstrip* appears similar to the following.

CDF Parameter	Value	Display
use external model file	<input type="checkbox"/>	off <input type="button" value=""/>
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	off <input type="button" value=""/>
er: permittivity	<input type="text" value="9.6"/>	off <input type="button" value=""/>
d: dielectric thickness (m)	<input type="text" value="100u"/>	off <input type="button" value=""/>
w: conductor width (m)	<input type="text" value="100u"/>	off <input type="button" value=""/>
t: conductor thickness (m)	<input type="text" value="10u"/>	off <input type="button" value=""/>
len: conductor length (m)	<input type="text" value="100000u"/>	off <input type="button" value=""/>
model type	<input type="radio"/> lossless <input checked="" type="radio"/> lossy	off <input type="button" value=""/>
lossy type	<input checked="" type="radio"/> narrow band <input type="radio"/> wide band	off <input type="button" value=""/>
sigma: conductivity	<input type="text" value="5.6e7"/>	off <input type="button" value=""/>
freq: max frequency (Hz)	<input type="text" value="1G"/>	off <input type="button" value=""/>

3. This time type 100000u in the *len: conductor length (m)* field.
4. Click *OK*.
5. In the schematic window, choose *Design—Check and Save* to check and save the CPW\_tlineSimple schematic.

You can now set up and run a simulation in the analog design environment using the new tline3 macromodel line length for the *microstrip* component.

### Simulating the CPW

1. In the Simulation window, choose *Analyses – Choose*.  
The Choosing Analyses form appears.
2. In the Choosing Analyses form, click *tran*.  
The form displays options needed for tran analysis.

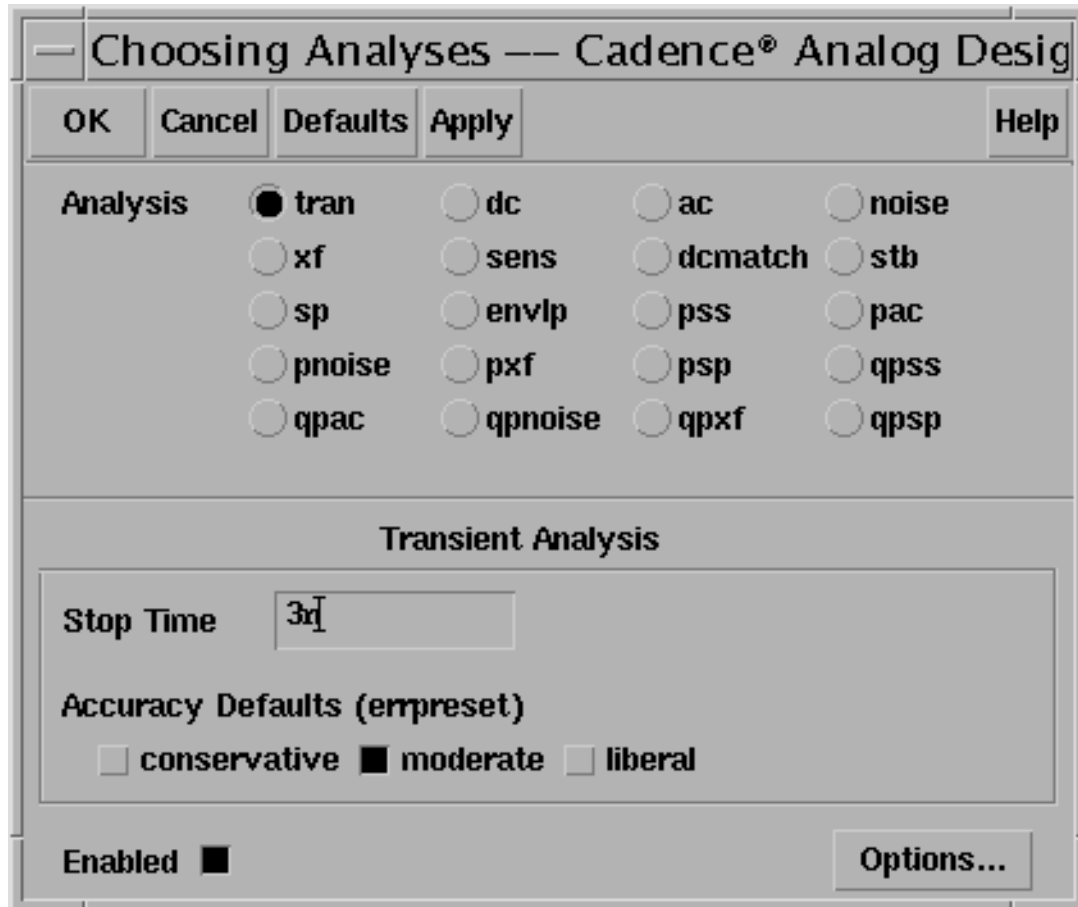
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

3. In the *Stop Time* field, type 3n.
4. Highlight *moderate* for the *Accuracy Defaults (errpreset)*.
5. Verify that *Enabled* is highlighted.

The Choosing Analyses form appears below.



6. Click *Apply* to check for setup errors for the Transient analysis.
7. Click *OK*.

The `Transient` analysis options you choose appear in the *Analyses* list box in the Simulation window.

<b>Analyses</b>					
#	Type	Arguments.....			Enable
1	tran	0	3n	mode..	yes

### Running the Simulation

1. In the Simulation window, choose *Simulation – Netlist and Run* to run the simulation.  
The output log file appears and displays information about the simulation as it runs.
2. Look in the CIW for a message that says the simulation completed successfully.

### Plotting the Coplanar Waveguide Signals

1. To open the Direct Plot form, choose *Results – Direct Plot – Direct Plot* in the Simulation window.  
The Waveform window and the Direct Plot form appear.
2. In the Direct Plot form do the following.
  - a. Highlight *Append* for *Plot Mode*.
  - b. Highlight *Tran* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The filled-in form looks like this.



The image shows a dialog box titled "Direct Plot Form". It contains several sections: "Plot Mode" with radio buttons for "Append" (selected) and "Replace"; "Analysis" with a radio button for "tran" (selected); "Function" with radio buttons for "Voltage" (selected) and "Current"; a "Select" dropdown menu showing "Net"; "Prepend Waveform from Reference Directory" checkbox (unchecked); "Add To Outputs" checkbox (unchecked); and a prompt at the bottom: "> Select Net on schematic...". Buttons for "OK", "Cancel", and "Help" are located at the top.

3. Follow the prompt at the bottom of the Direct Plot form.

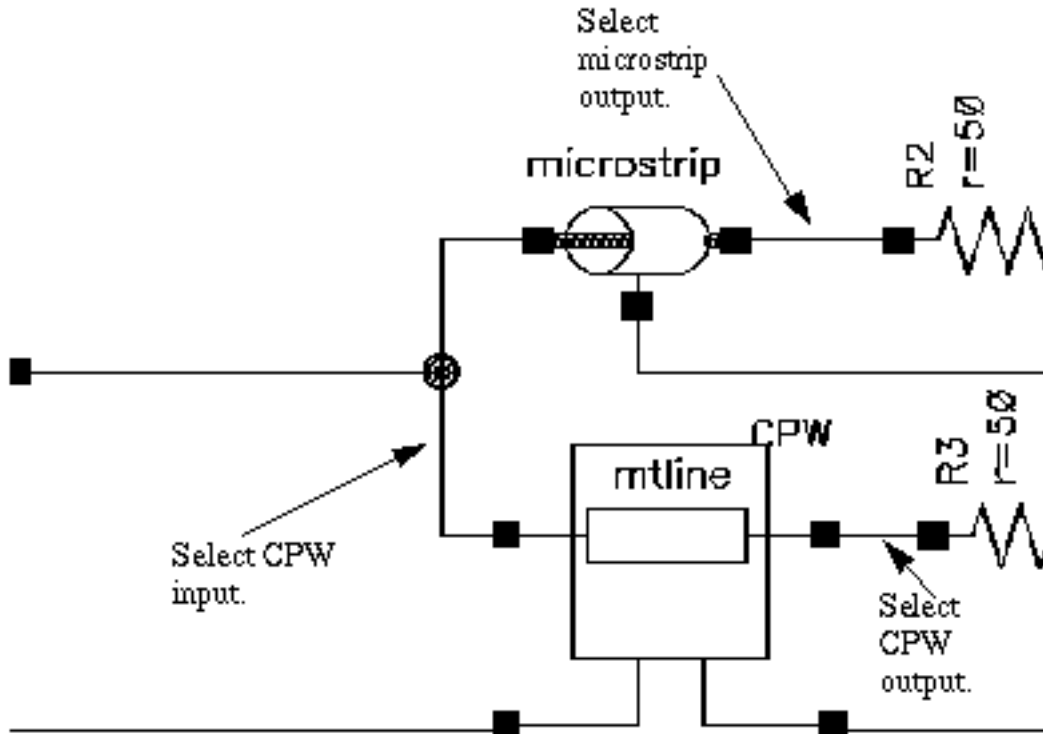
> Select Net on Schematic...

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

Select the following nets in the CPW\_tlineSimple schematic.window.

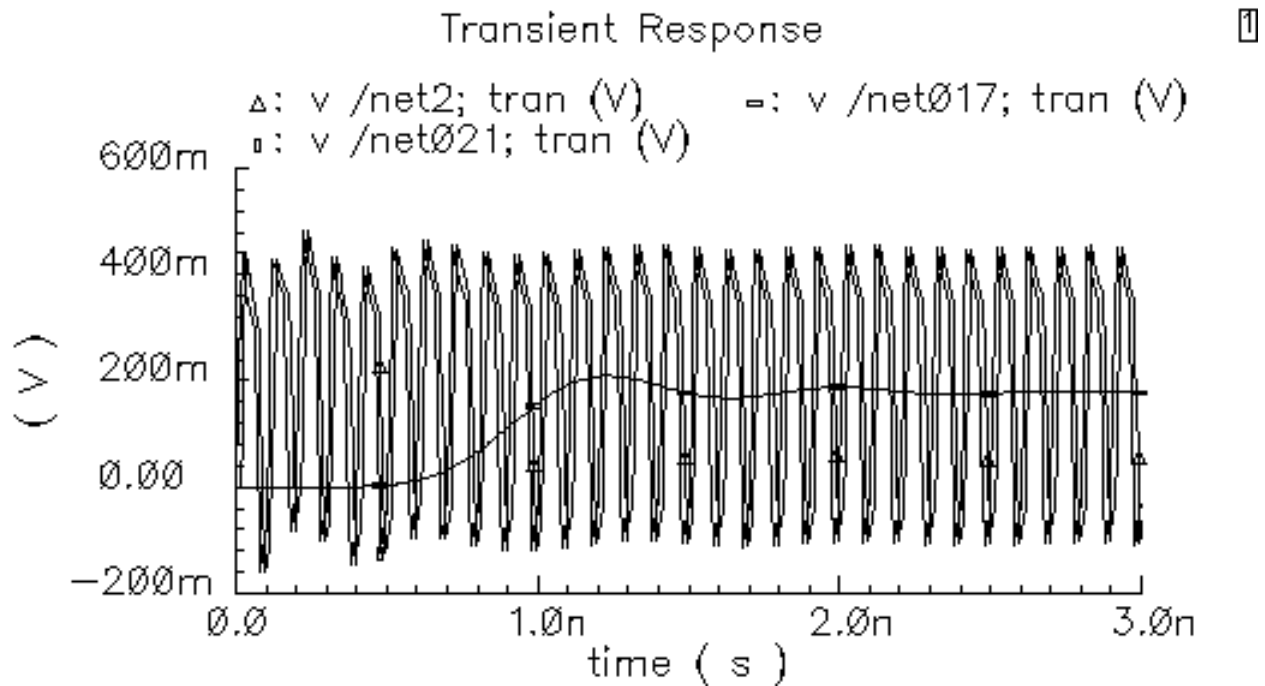


- a. Click the CPW input.
  - b. Click the microstrip output.
  - c. Click the CPW output.
4. Press Esc to stop selecting nets.

The plot in Figure 4-4 appears in the Waveform window.

**Figure 4-6 Signals Passing Through the CPW and the 10000u Microstrip Line with a 3n Stop Time**

my\_rfExamples CPW\_tlineSimple schematic : Apr 30 14:50:16 2003



From Figure 4-6, you can see that now that the microstrip is longer (100mm) its output also has a longer time delay. Also due to the longer microstrip, its resistance is greater and its output amplitude is less. The input signal is corrupt due to reflection of the *CPW* and *microstrip* signals, but the *CPW* output is almost identical to its input except for some phase delay.

You can see this more clearly by zooming in on an area of Figure 4-6. In the Waveform window, choose *Zoom – Zoom In*.

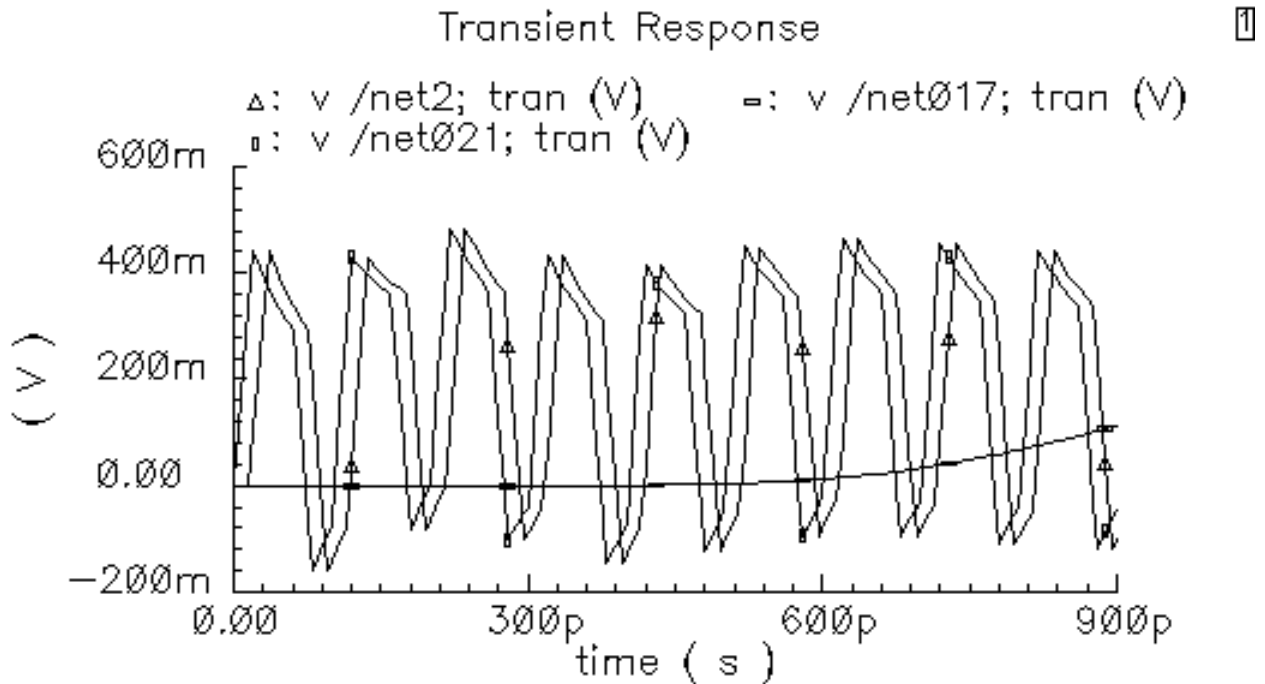
Select an area to enlarge by clicking at the top left corner of the area to enlarge. Then draw a rectangle around the area and click at the bottom right corner.

The enlarged area displays in Figure 4-6.



## Magnified View of Transient Response

my\_rfExamples CPW\_tlineSimple schematic : Apr 30 14:50:16 2003



## Simulating Coplanar Waveguides with the Generated Subcircuit Macromodel File

Simulate a Coplanar Waveguide, or CPW, using the macromodel subcircuit file, `cpw1.scs`, you created in Section “Using LMG to Obtain Subcircuit Macromodel and LRCG Files” on page 238. Add the `cpw1.scs` file to the CPW (`mtline`) component in the `CPW_tlineSimple` schematic. To do this you need the absolute path to the `cpw1.scs` file you noted in Section “Generating the Subcircuit and LRCG Files” on page 242.

This example uses the circuit `CPW_tlineSimple` from your editable copy of the `rfExamples` library (`my_rfExamples` here). This library also includes other sample circuits used in this manual. If you need assistance setting up or accessing your editable copy of the `rfExamples` library, refer to Chapter 3, “Setting Up for the Examples.”

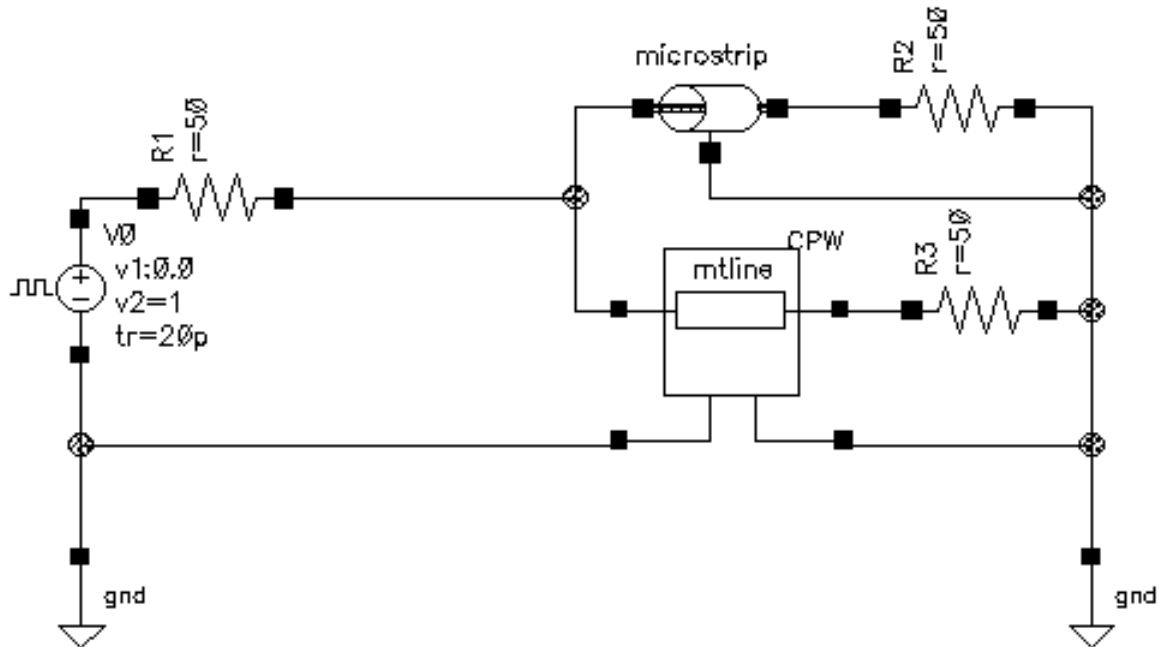
1. Open the `CPW_tlineSimple` circuit in the Schematic window as described in “Opening the CPW\_tlineSimple Circuit in the Schematic Window” on page 247.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The Schematic window for the CPW\_tlineSimple circuit appears.



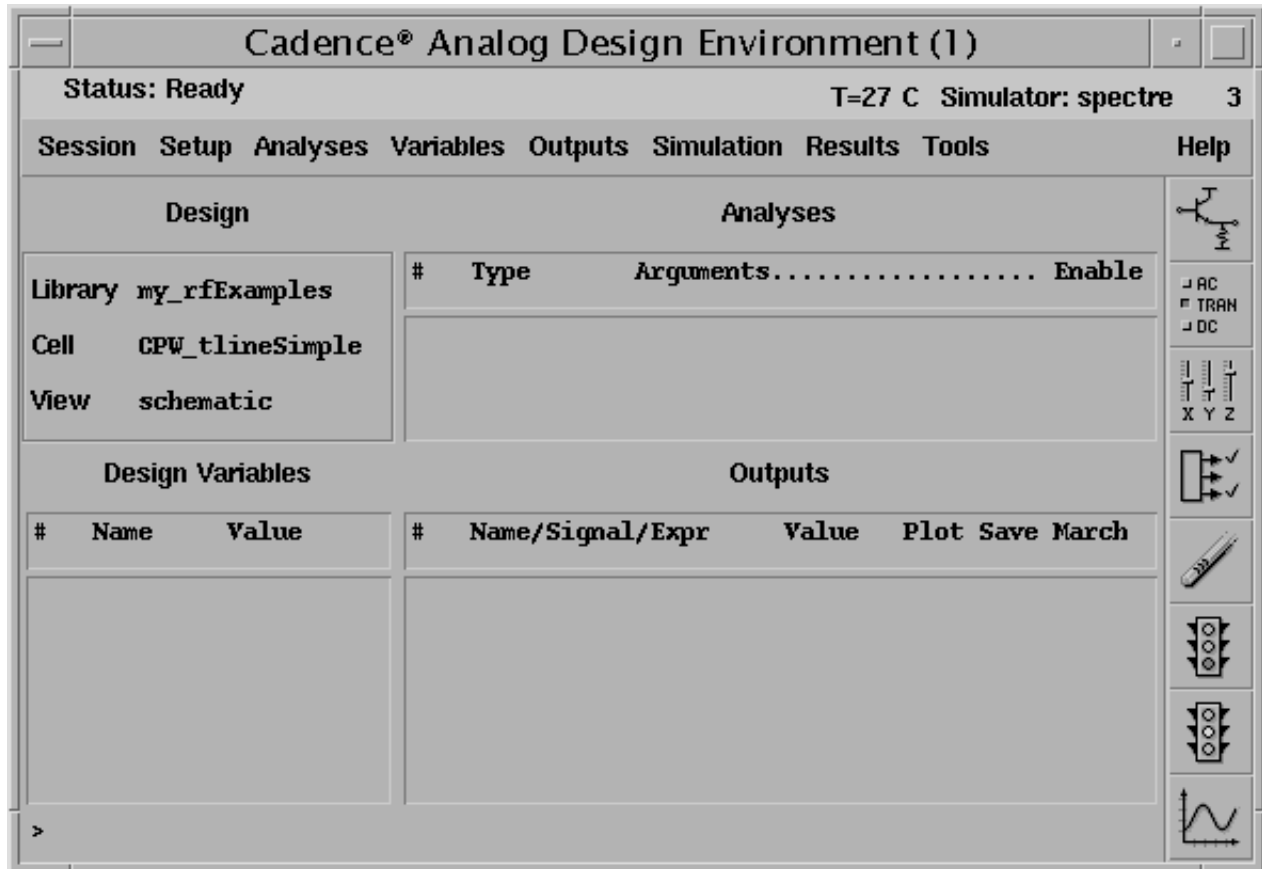
2. In the Schematic window, choose *Tools—Analog Environment*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The Simulation window for the CPW\_tlineSimple circuit opens.



3. In the CPW\_tlineSimple schematic, verify that the length of the microstrip component is 2000  $\mu\text{m}$ , as described in, [“Viewing the Properties of the Microstrip Component”](#) on page 250. If necessary, edit the conductor length CDF parameter.

len: conductor length (m) 2000u

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

The *CDF Parameter* section of the Edit Objects Properties form for the *microstrip* appears similar to the following.

CDF Parameter	Value	Display
use external model file	<input type="checkbox"/>	off <input type="checkbox"/>
line type	<input checked="" type="radio"/> microstrip <input type="radio"/> stripline	off <input type="checkbox"/>
er: permittivity	9.6	off <input type="checkbox"/>
d: dielectric thickness (m)	100 $\mu$	off <input type="checkbox"/>
w: conductor width (m)	100 $\mu$	off <input type="checkbox"/>
t: conductor thickness (m)	10 $\mu$	off <input type="checkbox"/>
len: conductor length (m)	200 $\mu$	off <input type="checkbox"/>
model type	<input type="radio"/> lossless <input checked="" type="radio"/> lossy	off <input type="checkbox"/>
lossy type	<input checked="" type="radio"/> narrow band <input type="radio"/> wide band	off <input type="checkbox"/>
sigma: conductivity	5.6e7	off <input type="checkbox"/>
freq: max frequency (Hz)	1G	off <input type="checkbox"/>

You can now set up and run a simulation in the analog design environment using the new `tline3` macromodel line length for the *microstrip* component.

### Associating the Lumped Model File with the Mtlne Component

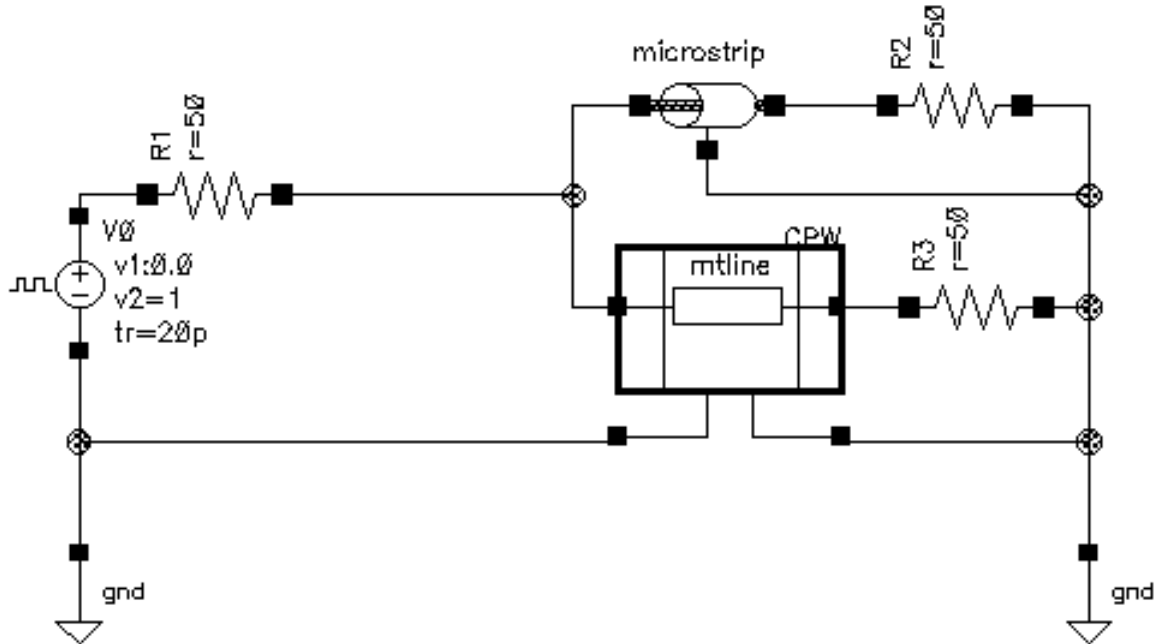
1. In the Schematic window, select the `mtline` transmission line component at the center of the schematic. This `mtline` component represents the *CPW*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The `mtline` transmission line symbol is now selected.



2. With the `mtline` symbol selected, choose *Edit—Properties—Objects* to open the Edit Object Properties form for the `CPW`.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The CFD parameter section of the Edit Object Properties form appears similar to the following.

CDF Parameter	Value	Display
Num of lines (excluding ref.)	<input type="text" value="1"/>	off <input type="checkbox"/>
RLCG data file	<input type="text" value="/belinda/w_linecpw1.dat"/>	off <input type="checkbox"/>
use lmg subckt	<input type="checkbox"/>	off <input type="checkbox"/>
<input type="button" value="Invoke 'LMG' parameter extraction tool"/>		
Physical length	<input type="text" value="2.0000m M"/>	off <input type="checkbox"/>
Enter RLCG etc. matrices	<input type="checkbox"/>	off <input type="checkbox"/>
Frequency scale factor	<input type="text" value=""/>	off <input type="checkbox"/>
ROM data file	<input type="text" value=""/>	off <input type="checkbox"/>
Multiplicity factor	<input type="text" value="1"/>	off <input type="checkbox"/>

**3.** Select *use lmg subckt*.

The CFD parameter section changes to allow you to enter a lumped model file name.

CDF Parameter	Value	Display
Num of lines (excluding ref.)	<input type="text" value="1"/>	off <input type="checkbox"/>
LMG subcircuit file	<input type="text" value="R397042/397042/cpw1.scs"/>	off <input type="checkbox"/>
use lmg subckt	<input checked="" type="checkbox"/>	off <input type="checkbox"/>
<input type="button" value="Invoke 'LMG' parameter extraction tool"/>		

**4.** In the *LMG subcircuit file* field type the full, absolute path to the lumped model file you generated in “Generating the Subcircuit and LRCG Files” on page 242.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

For example, enter `/home/belinda/cpw1.scs`.

CDF Parameter	Value	Display
Num of lines (excluding ref.)	<input type="text" value="1"/>	off <input type="checkbox"/>
LMG subcircuit file	<input type="text" value="/home/belinda/cpw1.scs"/>	off <input type="checkbox"/>
use lmg subckt	<input checked="" type="checkbox"/>	off <input type="checkbox"/>
<input type="button" value="Invoke 'LMG' parameter extraction tool"/>		

**Note:** If you receive an error message later when you simulate with this model, see [“Resolving a Possible Error Message in the mtline Model File”](#) on page 244 for information on correcting this error.

5. In the *Edit – Properties – Objects* form, click *OK*.

When you click *OK*, the form closes and the `cpw1.scs` file is saved in the schematic editor. When you simulate with the *mtline* component, the `cpw1.scs` lumped model file is used.

6. In the schematic window, choose *Design—Check and Save* check and save for the `CPW_tlineSimple` schematic.

You can now set up and run a simulation using the new `mtline` macromodel in the analog design environment.

### Simulating the CPW

1. In the Simulation window, choose *Analyses – Choose*.

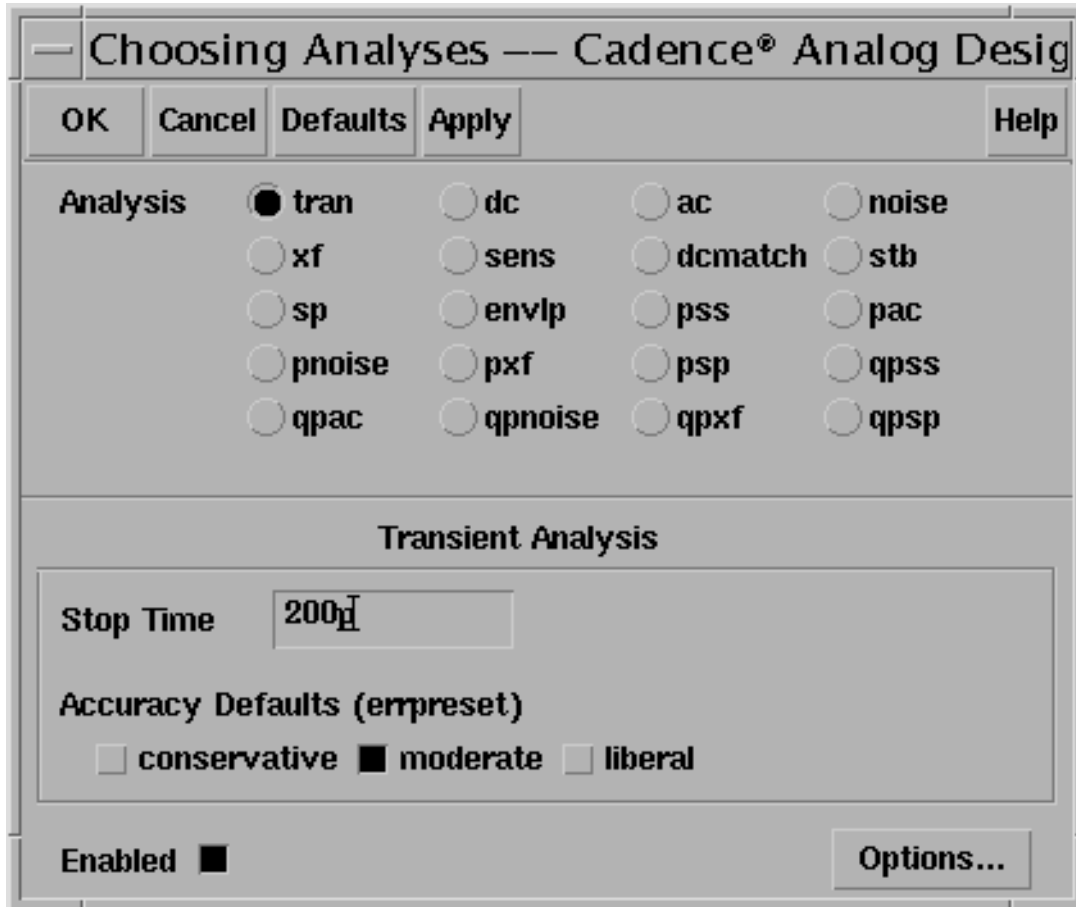
The Choosing Analyses form appears.

2. In the Choosing Analyses form, click *tran*.

The form displays options needed for `tran` analysis.

3. In the *Stop Time* field, type `200p`.
4. Highlight *moderate* for the *Accuracy Defaults (errpreset)*.
5. Verify that *Enabled* is highlighted.

The Choosing Analyses form appears below.



6. Click *Apply* to check for setup errors for the Transient analysis.
7. Click *OK*.

The *Transient* analysis options you choose appear in the *Analyses* list box in the Simulation window.

<b>Analyses</b>				
#	Type	Arguments.....		Enable
1	tran	0	200p	yes



### **Running the Simulation**

1. In the Simulation window, choose *Simulation – Netlist and Run* to run the simulation.  
The output log file appears and displays information about the simulation as it runs.
2. Look in the CIW for a message that says the simulation completed successfully.

### **Plotting the Coplanar Waveguide Signals**

1. To open the Direct Plot form, choose *Results – Direct Plot – Direct Plot* in the Simulation window.  
The Waveform window and the Direct Plot form appear.
2. In the Direct Plot form do the following.
  - a. Highlight *Append* for *Plot Mode*.
  - b. Highlight *Tran* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

The filled-in form looks like this.



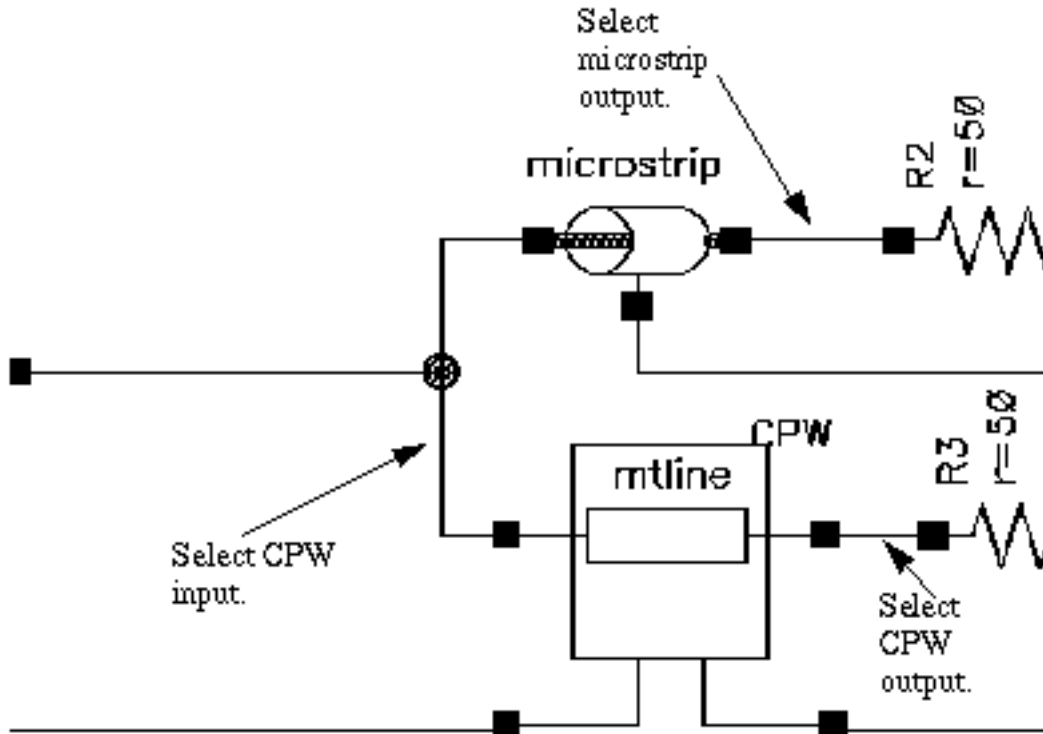
The image shows a dialog box titled "Direct Plot Form". It contains several controls:

- Buttons: "OK", "Cancel", and "Help".
- Plot Mode: Radio buttons for "Append" (selected) and "Replace".
- Analysis: A list box containing "tran" (selected).
- Function: Radio buttons for "Voltage" (selected) and "Current".
- Select: A text box containing "Net".
- Prepend Waveform from Reference Directory: An unchecked checkbox.
- Add To Outputs: An unchecked checkbox.
- Bottom prompt: "> Select Net on schematic...".

3. Follow the prompt at the bottom of the Direct Plot form.

> Select Net on Schematic...

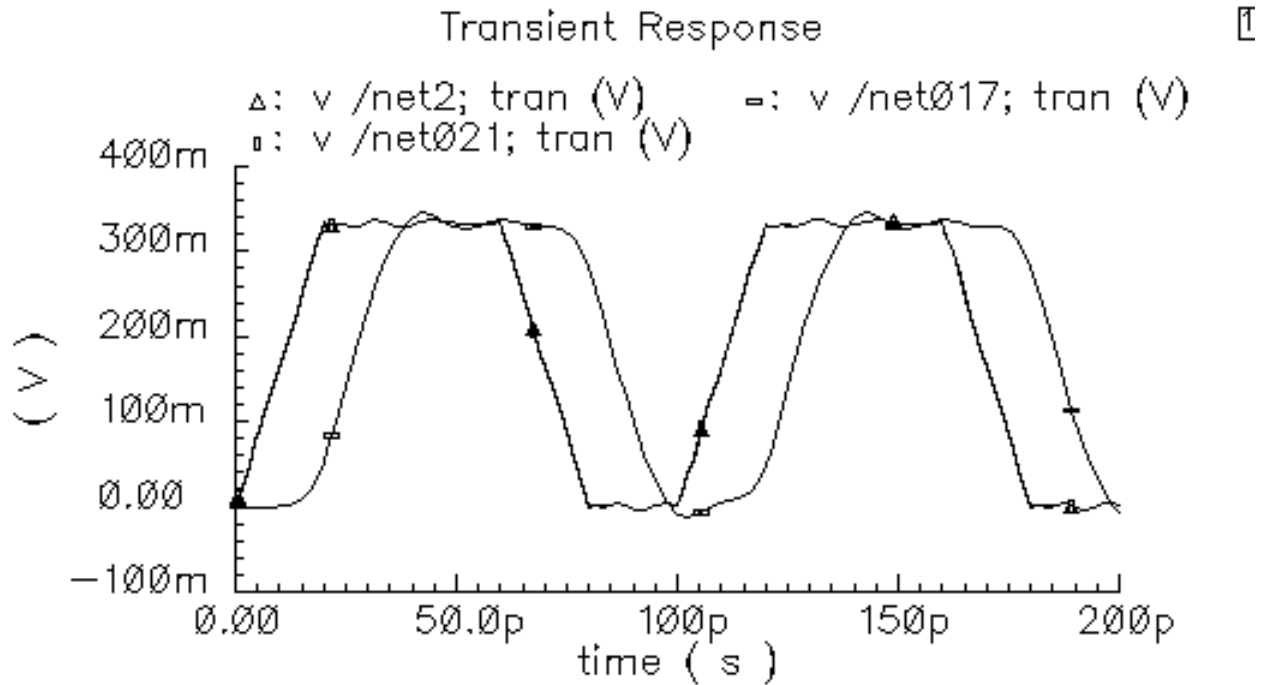
4. Select the following nets in the CPW\_tlineSimple schematic.



- Click the CPW input.
  - Click the microstrip output.
  - Click the CPW output.
5. Press Esc to stop selecting nets.

The plot in Figure 4-4 appears in the Waveform window.

**Figure 4-7 Signals Passing Through the CPW and the Microstrip**



Comparing Figure 4-7 to Figure 4-5, you can see that the corruption of the input signal is mainly due to the reflection that comes from the *microstrip*. Because the length of CPW is now very short, 20  $\mu\text{m}$ , the output signal from the CPW is almost overlapping its input signal. From the Figure 4-7 you can also see that the lumped CPW model LMG has created is very accurate.

### Internal Techniques in LMG and Model Accuracy

In the current release of LMG, both the LMG GUI and its engine have been rewritten. Both the boundary element method (BEM) and the boundary spectral method (BSM) provide efficient parameter extraction of per-unit length LRCG matrixes for multi-layer multi-conductor transmission lines. Special discretization and basis functions efficiently capture both the skin effect and the proximity effect. The substrate coupling effects that impact on the CG matrixes are extracted by enforcing continuity conditions across different dielectric interfaces and the voltage condition on conductor surfaces. After frequency-dependent LRCG matrixes are extracted, a robust rational function fitting algorithm generates lumped interconnect models via Gaussian quadrature.

For the frequency-dependent per-unit length LRCG matrixes, we have made many comparisons between the current LMG tool and other measurements, other commercial tools

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

and other publications. On both IC and PCB levels, the parameter extraction of the current LMG is efficient and accurate.

The LMG Gaussian quadrature lumped model is a simple yet accurate model. But since we also provide the *mtline* transmission line model and *mtline* focuses on simulation, we suggest that for long transmission lines, you use LMG to do parameter extraction and to provide the LRCG file to *mtline* to do the simulation. The current LMG combined with the *mtline* transmission line model provides a complete solution for transmission line modeling.

## The LMG GUI Reference

In General, LMG does the following

- LMG models 4 transmission line types:
  - Microstrip
  - Stripline
  - CPW (The coplanar wave guide line is new in 5.0.0.)
  - SubLossline (Includes stripline, microstrip line, lossy multilayer and multiconductor lines.)
- LMG generates 3 model types:
  - Lossless (All metal layers are PEC with no internal L.)
  - Lossy Narrow Band (Model operates at 1 frequency set with Fmax.)
  - Lossy Wide Band (Operates from DC to the frequency set with Fmax.)
- You can invoke LMG in 4 ways:
  - As a standalone GUI, type `lmg` at the operating system prompt.
  - In the analog design environment schematic flow, edit the `mtline` model's object properties during simulation setup.
  - From the *RF* Submenu, open the LMG GUI.
  - From the command line without the LMG GUI, type `lmg - help` at the operating system prompt to display the LMG command and options.

LMG extracts parameters, produces per-unit-length LRCG matrixes, and generates models and subcircuits. You can input existing, frequency-dependent LRCG matrixes directly into the `mtline` model by editing the model's object properties during simulation setup.

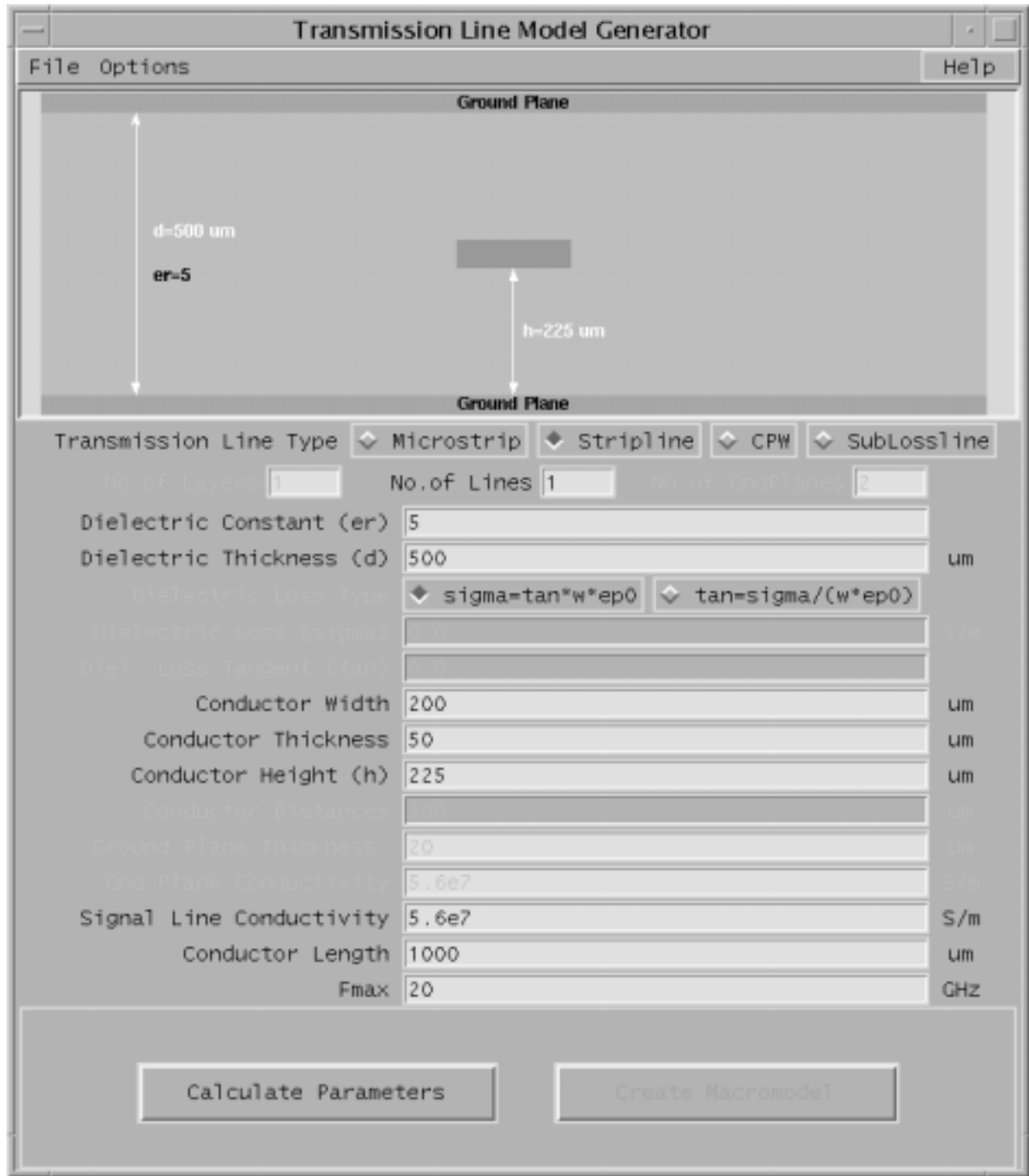
New in 5.0.0, LMG models transmission lines without ground planes. (Input 0 in the LMG GUI's "No. of GndPlanes" field.) LMG uses the last line as the reference return for both current and charge conservation. LMG performs size reduction for the lumped model and RLCG file.

The Transmission Line Generator (LMG) GUI is described in the following sections.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmission Lines

### LMG GUI



The LMG GUI, is divided into four sections.

- Menu
- Display Section
- Data Entry Section
- Function Buttons

## Menus for the LMG GUI

The two pull-down menus at the top of the GUI.

Use the two menus, *File* and *Option*, to set up LMG and to begin creating and saving models.

### File Menu

The *File* menu has the following four options:

- *Subcircuit Name*
- *LRCG File Name*
- *Save Current Settings*
- *Quit*

### *Subcircuit Name*

Choose *File—Subcircuit Name* to bring up the Subcircuit Name form in which you type the name of the output subcircuit. In this case, the default subcircuit name is `tline`. LMG creates the output subcircuit file name by appending `.scs` to the name you enter.



Enter a name that describes the model you are creating.



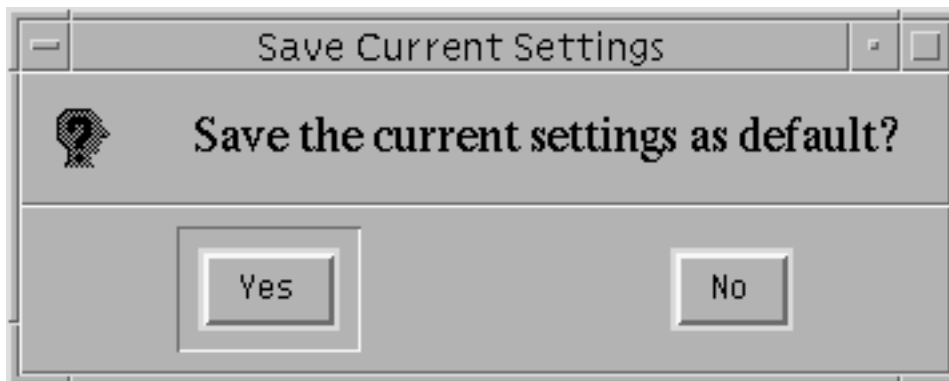
### ***LRCG File Name***

Choose *File—LRCG File Name* to bring up the LRCG File Name form in which you type the name of the output LRCG file name where the frequency dependent per-unit length LRCG matrixes are stored in the format consistent with the mtline macromodel. In the case illustrated here, the default file name is `w_line1.dat`.



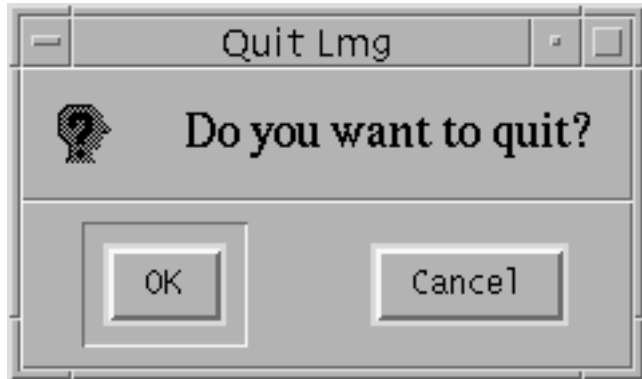
### ***Save Current Settings***

Saves the current transmission line information in the file `./initlmg` in the current directory as a defaults for the next LMG session if you click *Yes*.



## Quit

Exits `lmg` if you click *OK*.



## Options Menu

The *Options* menu has five main options.

- *Model Type*
- *Subckt Format*
- *Length Format*
- *Freq. Unit*
- *Output file Control*

Each option has a submenu in which you choose a value. After you choose a value, the geometry of the diagram is updated automatically to implement your choice.

### ***Model Type***

The three *Model Type* options you choose from are

- *Lossy, Wide Band*

Parameter extraction includes metal loss from both signal lines and ground planes. The frequency range is from DC to  $F_{max}$ .

- *Lossy, Narrow Band*

Parameter extraction includes metal loss from both signal lines and ground planes. The working frequency is set to one frequency:  $F_{max}$ .

■ *Lossless*

Parameter extraction does not include the metal loss from signal lines and ground planes. This means that all the metals are regarded as perfect electrical conductors (PEC), and, at this point, Fmax is meaningless, for the code is internally setup to work at the frequency of 50 GHz to force the current to flow on the surface of the metals.

***Subckt Format***

Select the simulator to use one of two subcircuit formats.

- *Spectre*
- *SpectreS*

***Length Unit***

Selects the length unit to use.

- *um*
- *mm*
- *m*
- *mil*

The currently selected length unit is displayed to the right of the data entry fields in the LMG GUI. All the dimension quantities use the same unit for consistency.

***Freq. Unit***

Selects the frequency unit to use.

- *Hz*
- *MHz*
- *GHz*

The currently selected frequency unit is displayed to the right of the data entry fields in the LMG GUI.

### **Output file Control**

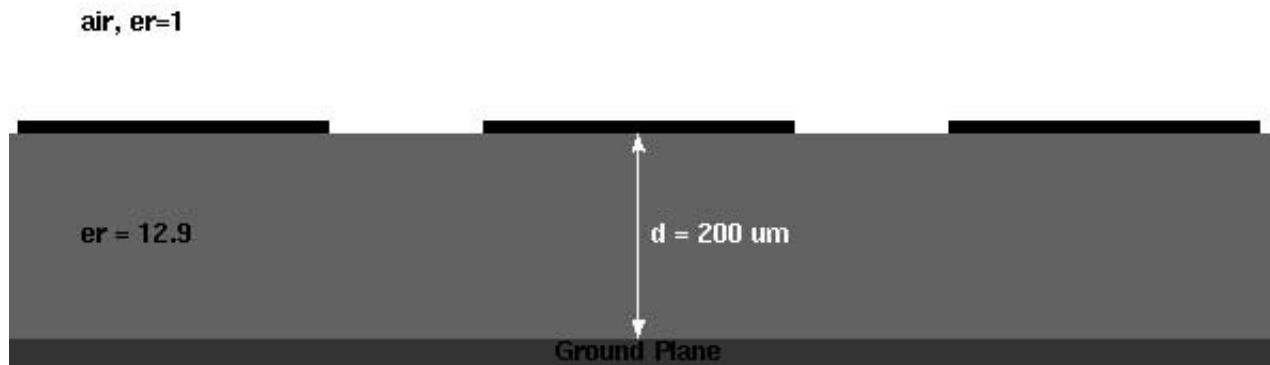
Selects which output files to create.

- *RLCG file*
- *Lumped Model*
- *Both*

The selected files are created when you press the *Create Macromodel* button.

### **Display Section**

Displays the two dimensional cross section of the transmission line. Inside the cross section also displays the dielectric thickness, the dielectric constant, and the conductor height. The display is to scale and applies the same scale for both the horizontal and vertical dimensions.



Any time you specify new parameter information in the data entry section, the display changes immediately to reflect your change.

### **Data Entry Section**

In the data entry section, you type information describing the transmission line you are modeling. If a field has a label, you can leave the default value unmodified, but you cannot leave a field blank.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmission Lines

---

Do not put values in the fields that are grayed out. Grayed out fields represent quantities that do not accept data for the model you are using. The meanings of the data entries are explained in the following sections.

Some data entry fields accept more than one data entry. To enter multiple data items, separate them with a single space.

When you enter data incorrectly, the field is flooded in red until you correct your mistake.

#### **Transmission Line Type: *Microstrip*, *Stripline*, *CPW*, *SubLossline***

The 4 radio buttons at the top of the data entry section let you choose between the *Microstrip*, *Stripline*, *CPW* or *SubLossline* transmission line types.

*CPW* models a coplanar waveguide.

*SubLossline* includes transmission line systems with multi-layer multi-conductors. It also includes cases with or without substrate loss. In fact, *SubLossline* includes both *Microstrip* and *Stripline*. For reasons of speed we keep *Microstrip* and *Stripline* as transmission line types.

#### **Number of Layers**

Type an integer number of layers for the transmission line up to 12.

#### **Number of Lines**

Type an integer number of conductors for the transmission line up to 30.

#### **Number of GndPlanes**

Type an integer number of ground planes for the transmission line up to two.

#### **Dielectric Constant ( $\epsilon_r$ )**

For *Microstrip* transmission lines,  $\epsilon_r$  is the relative dielectric permittivity of the substrate with air above the substrate.

For *Stripline* transmission lines,  $\epsilon_r$  is the relative permittivity of the dielectric between the upper and lower ground planes.

For *SubLossline* transmission lines, *er* is accounted for from bottom to top, each layer's *er* is separated by blank space. If the number of *er* values is less than the number of layers, then the last *er* value is automatically repeated.

### **Dielectric Thickness (d)**

For *Microstrip* transmission lines, *d* is the thickness of the substrate.

For *Stripline* transmission lines, *d* is the distance between the upper and lower ground planes.

For *SubLossline* transmission lines, *d* is accounted for from bottom to top, each layer's thickness *d* data is separated by blank space. If the number *d* is less than the number of layers, the last *d* is repeated.

### **Dielectric Loss Type: ( $\sigma = \tan * w * \epsilon_0$ ) or ( $\tan = \sigma / (w * \epsilon_0)$ )**

The 2 radio buttons allow you to select the appropriate Dielectric loss type *sigma* (for IC users) or *loss tangent* (for PCB users).

*sigma = tan \* w \* ep0* (the *Dielectric Loss (sigma)* field is activated)

■ *tan = sigma / (w \* ep0)* (the *Dielectric Loss Tangent* field is activated)

For a wide band model,

- when you select tangent, tangent is constant over the frequency range.
- when you select sigma, sigma is constant over the frequency range.

**Note:** When you activate one field, for example, *Dielectric Loss (sigma)*, you deactivate the other field, *Dielectric Loss Tangent*.

### **Dielectric Loss Sigma**

For the *SubLossline* transmission line, each dielectric can have substrate loss (for  $\sigma \geq 1.0e^{-6}$  S/m) or not have substrate loss (for  $\sigma < 1.0e^{-6}$  S/m). We ignored  $\sigma < 1.0e^{-6}$  and treat it as 0.

### **Conductor Width**

Specifies the width of each conductor. The conductors are accounted for from left to right. Separate each conductor line width with a blank space. If the number of conductor widths is less than the number of conductor lines, then the last width is repeated.

### **Conductor Thickness**

Specifies the thickness of each conductor. The conductors are accounted for from left to right. Each line's thickness is separated by blank space. If the number of thickness values is less than the number of conductor lines, then the last thickness is repeated.

### **Conductor Height (h)**

*Conductor Height* is for the *Stripline* and *SubLossline* transmission lines only. It specifies the distance between the lower face of the conductor wire and the upper face of the bottom ground plane.

### **Conductor Distances**

Either

- There are multiple conductors (You entered a number greater than 1 in the *No. of Lines* field), and the *Conductor Distances* field is active.
- There is a BoardSide coupled transmission line
  - For two conductors, enter the distance between the conductors.
  - For three or more conductors, enter the distances between the conductors separated by a space.
  - For a boardside coupled transmission line, enter a negative number. Conductor Distance is the only entry in the LMG GUI that you can input a negative number.
  - Ground Plane Thickness

Specifies ground plane thickness. You can specify one or two ground plane thicknesses. Two ground planes can have different thicknesses. The first is for the bottom ground plane, the second is for the top ground plane. They are separated by a space.

### **Gnd Plane Conductivity**

Specifies ground plane conductivity. If you do not want extracted parameters including ground plane loss effects, input a very larger number, say the ground plane conductivity is  $1.0e^{14}$ .

### Signal Line Conductivity

Specifies the conductivity of the conductor wire. Relative to the per-unit length R and L matrixes.

### Conductor Length

Specifies the length of the transmission line in the unit shown to the right of the data field.

### Fmax

The maximum frequency at which the macromodel must work properly. You usually set  $F_{max}$  to the frequency of the highest harmonic that has significant energy in the transmission line simulations. LMG generates a macromodel of the transmission line that is good from DC to the value chosen for  $F_{max}$  if the model type is chosen as *Lossy, Wide Band*. If the model type is chosen as *Lossy, Narrow Band*, then LMG just do parameter extraction and model generation at frequency defined by  $F_{max}$ . However, if model type is chosen as *Lossless*, then  $F_{max}$  is ignored because the tool sets a very high working frequency internally to avoid the internal inductance. Looking at the LRCG file, you can find what is the real frequency you used.

### Function Buttons for the LMG GUI

The function buttons

- Update the figure display
- Calculate the transmission line parameters
- Create the macromodels.
- There are two function buttons at the bottom of the Transmission Line Modeler GUI.





### **Calculate Parameters**

After you specify the 2D transmission line geometry, click this button to calculate the transmission line parameters. The display shows the characteristic impedance, propagation velocity, and the DC resistance per meter of the transmission line in the upper right corner of the form if applicable.

### **Create Macromodel**

After you click the *Calculate Parameters* button, this button is enabled. Click this button to create the transmission *mline* model. A pop-up window tells you the name of the transmission line model file.

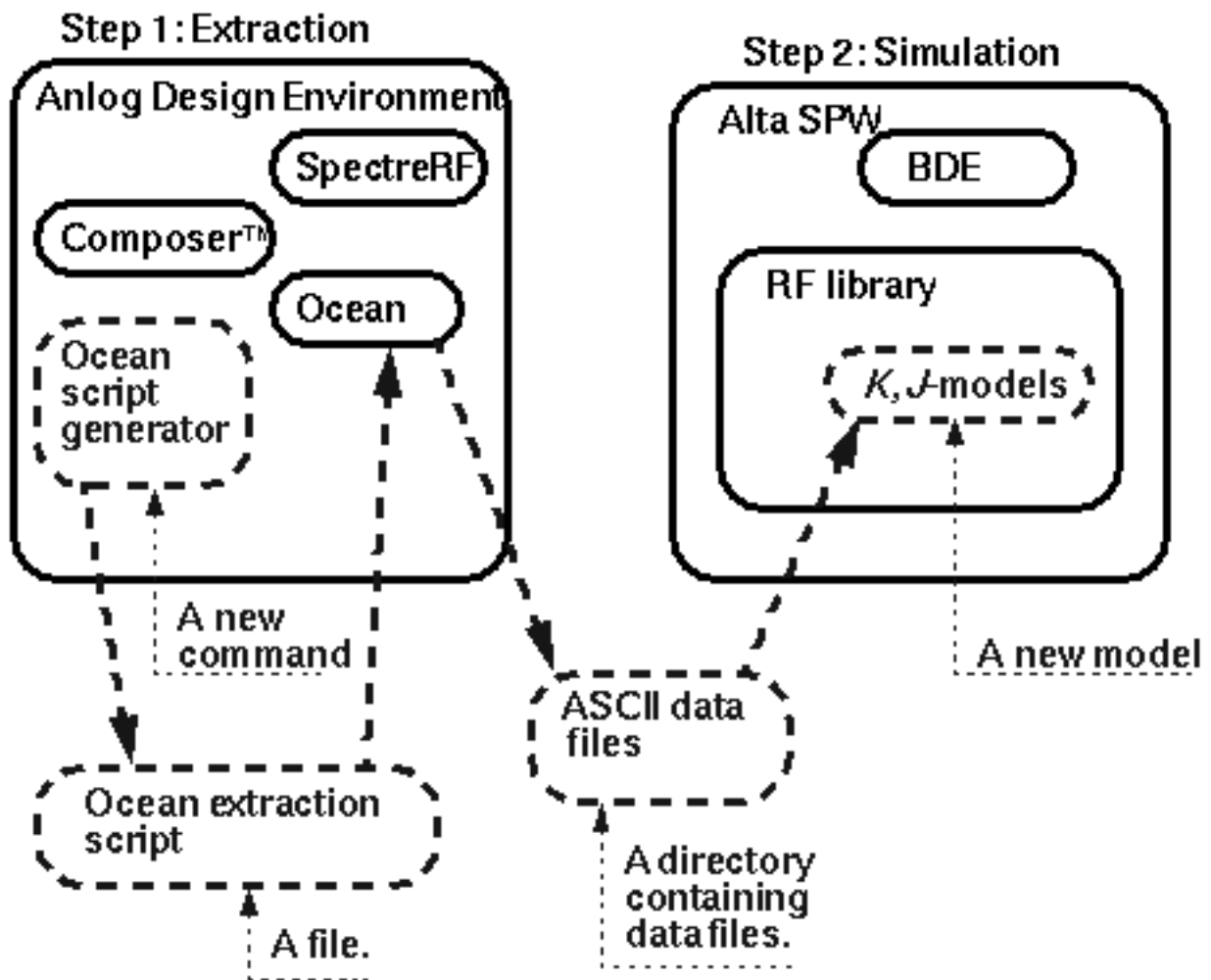
**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Modeling Transmission Lines

---

## Creating and Using Receiver K-Models

The receiver K-model links Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) and Alta™ SPW as shown in Figure .

### Task Flow of K-Model Creation and Use



The Ocean extraction script automatically runs a set of Spectre RF analyses and writes the results to a directory you specify. The results characterize the baseband performance of the RF circuit.

There are three kinds of K-models.

- **Transmitter:** This is actually a J-model and is discussed in chapter 10.
- **Receiver (linear):** This is a linearized model of the receiver. It includes noise except for phase noise. Extraction is very similar to extraction of the non-linear K-model.
- **Receiver (non-linear):** This is a noiseless non-linear model of the receiver.

The receiver K-models only exist in the Alta SPW RF library but you extract the K-model data using Spectre RF in the analog design environment.

This chapter describes the following:

- The procedure for extracting a non-linear receiver K-model.
- The K-model's limitations
- The procedure for handling strong out-of-band blockers

The K-model data files and their format are described at the end of the chapter.

## Procedures for Simulating `k_mod_extraction_example`

Follow the steps below to simulate the `k_mod_extraction_example`:

1. Copy `k_mod_extraction_example` from the Cadence `rfExamples` directory into a local directory.

Be sure the path to the local directory is defined in your `cds.lib` file.

2. Choose `File – Open` in the Command Interpreter Window (CIW).

The Open File form appears. Filling in the Open File form lets you open the schematic.

3. In the Open File form, choose `<Path_to_local_copy>` for Library Name.
4. In the `Cell Names` list box, highlight `k_mod_extraction_example`.

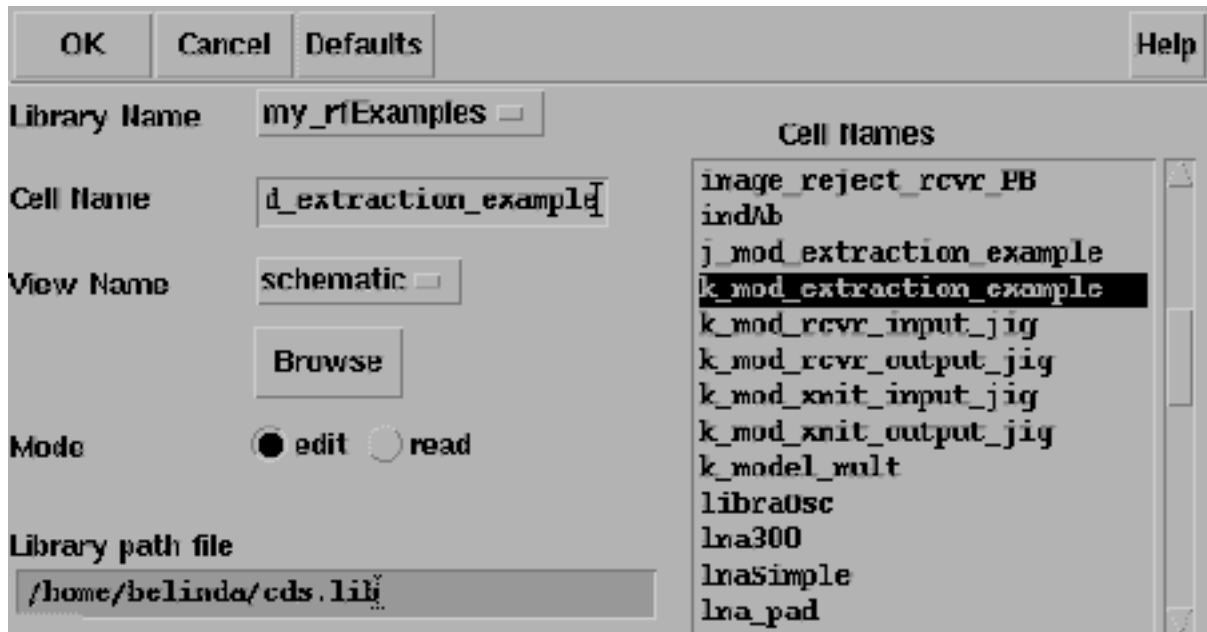
`k_mod_extraction_example` appears in the `Cell Name` field.

5. Choose `schematic` for `View Name`.
6. Highlight `edit` to select the `Mode`.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

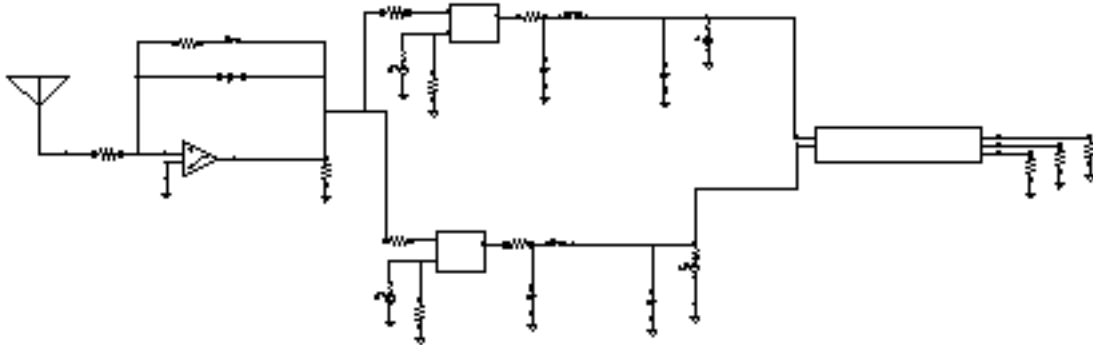
## Creating and Using Receiver K-Models

The Open File form appears as follows.



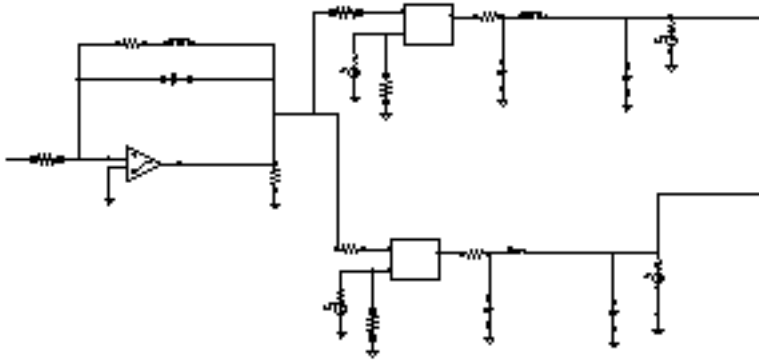
7. Click **OK** in the Open File form.

The Schematic window opens with the *k\_mod\_extraction\_example* schematic.



### Optional: Setting Up the Input and Output Jigs

If you want to see the whole procedure for setting up the K-model, including the editing of the schematic, delete the input and output test jigs at the far left and right of the schematic. The edited schematic now looks like the one below.



From the Schematic window, do the following:

1. Select *Add – Instance*.

The Add Instance form appears.

2. In the Add Instance form, click *Browse*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

The Library Browser – Add Instance form appears. In the Library Browser – Add Instance form, do the following:

3. Select *rfExamples* for the *Library*.

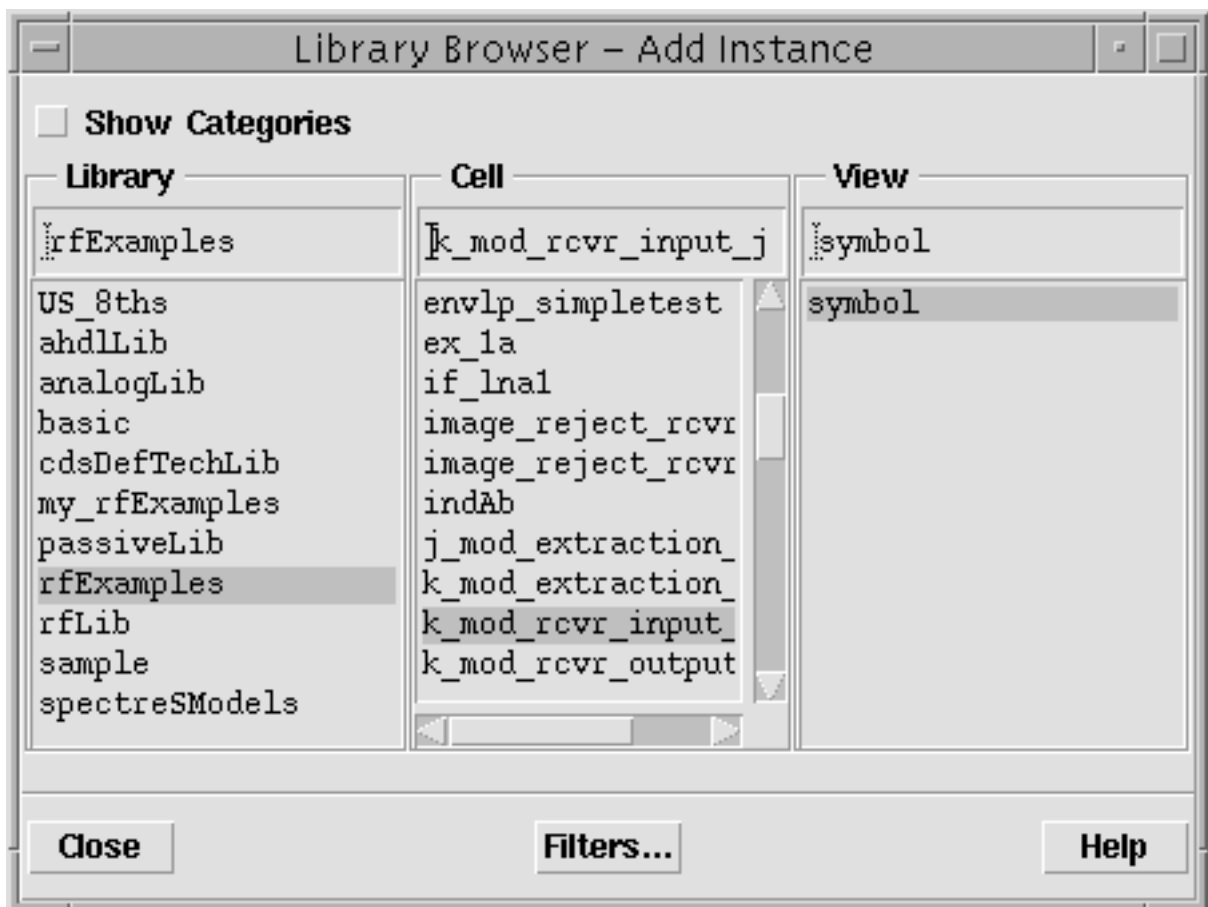
The form changes to let you select a cell.

4. Select *k\_mod\_rcvr\_input\_jig* for the *Cell*.

The form changes to let you select a View.

5. Select *symbol* for the *View*.

The completed form appears like the one below



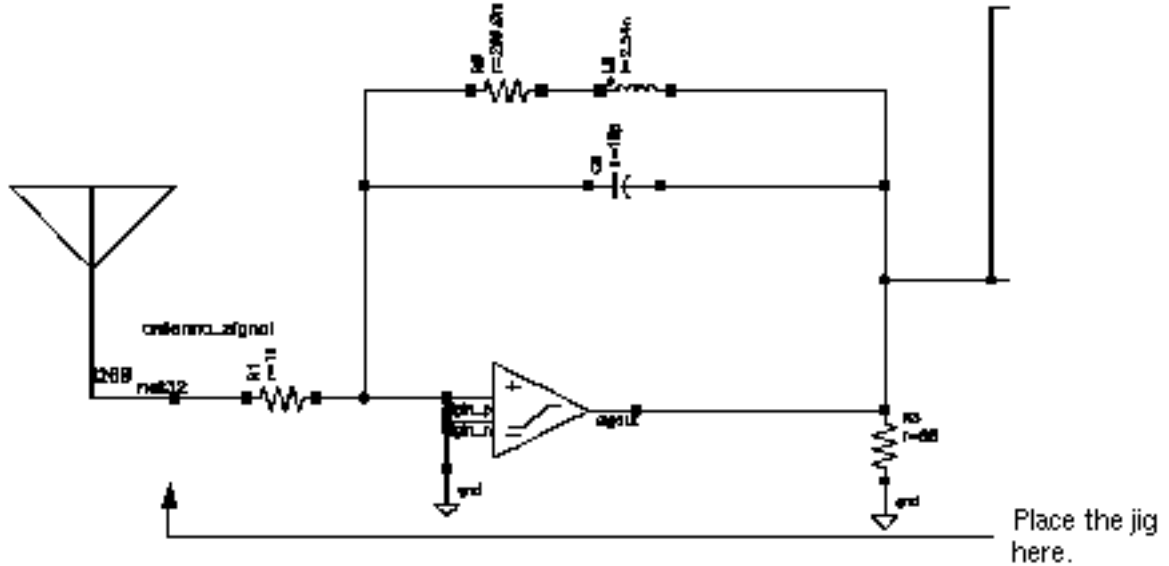
When you place the cursor back in the Schematic window, an instance of *k\_mod\_rcvr\_input\_jig* is attached to it.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

---

6. Place the input jig at the appropriate place in the schematic.



If you need assistance with the procedures for editing a schematic, see the *Virtuoso® Schematic Composer™ User Guide*.

7. In the Schematic window, again select *Add – Instance*.
8. Now repeat the steps you used to add the input jig to the schematic and add the output jig *k\_mod\_rcvr\_output\_jig*. Also add a resistor connected to ground to each of the three output connections of the output jig.

The table below tells you the libraries, cells, and views of the components you add.

Component	Library	Cell	View
output jig	rfExamples	k_mod_rcvr_output_jig	symbol
resistor	analogLib	res	symbol
ground	analogLib	gnd	symbol

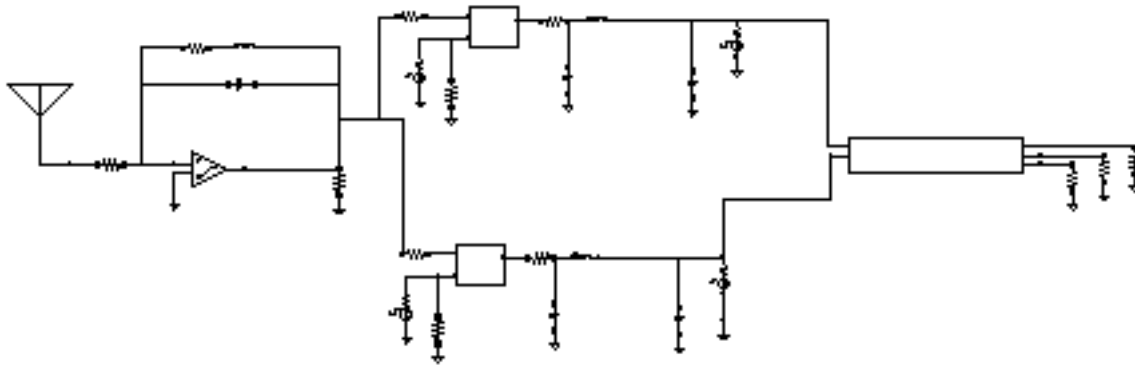


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

---

The edited schematic looks like this.



In the Schematic window, perform the following steps:

1. Click the input jig to select it.
2. Select *Edit – Properties – Objects* in the Schematic window.  
The Edit Object Properties form appears.
3. In the Edit Object Properties form, type 1G for the *carrier frequency* value.
4. In the Edit Object Properties form, click *OK*.

The completed form appears like the one below:

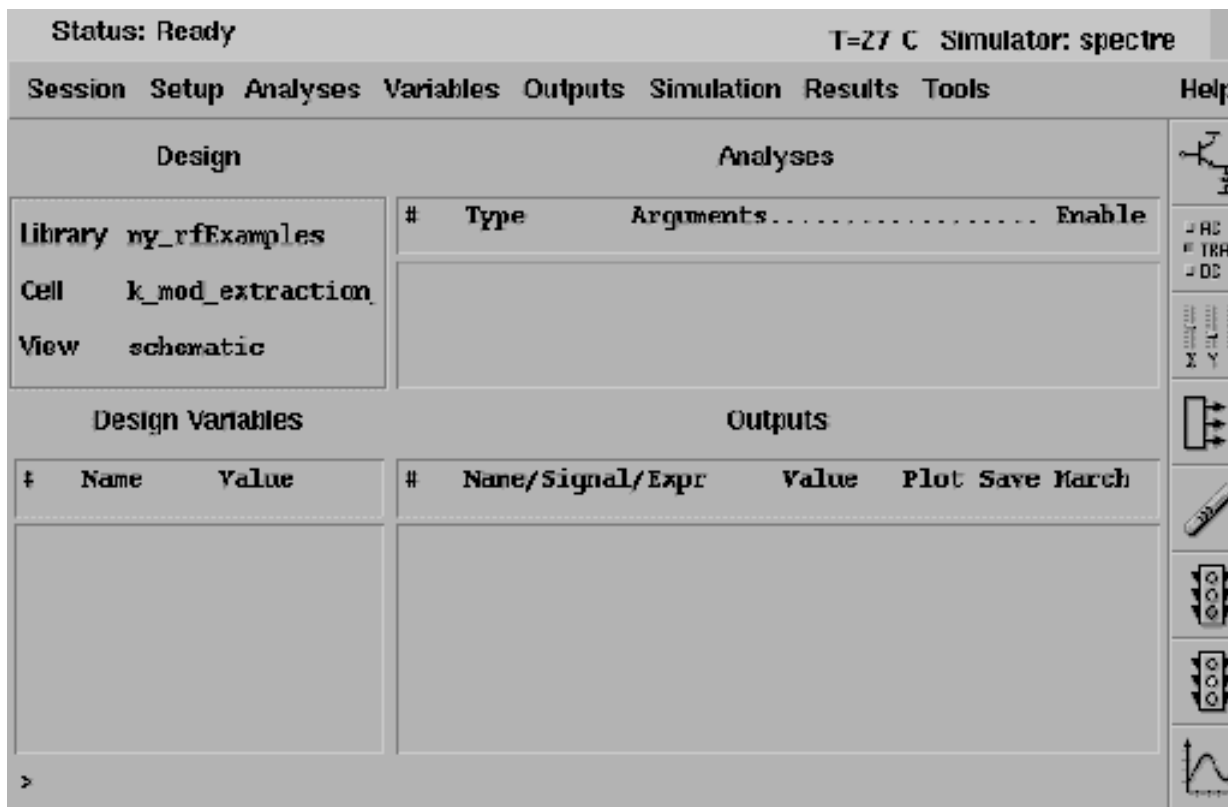
CDF Parameter	Value
<b>carrier frequency</b>	1G Hz
Knottlet: signal bias	sig_bias V
Knottlet: carrier phase	car_ph deg
Knottlet: signal bias end point	bias_end V
Carrier frequency name	car_freq

5. In the Schematic window, select *Design – Check and Save*.

## Analysis Setup

- Choose *Tools – Analog Environment* in the Schematic window.

The Simulation window appears.



## Setup the Simulator, Directory, and Host

1. Choose *Setup – Simulator/Directory/Host* in the Simulation window.

The Choosing Simulator/Directory/Host form appears. In the Choosing Simulator/Directory/Host form, do the following:

2. Highlight *spectre* for *Simulator*.
3. Type in the name of the project directory, if necessary.
4. Highlight *local*. (If you highlight *remote Host Mode*, type in the name of the host machine and the remote directory in the appropriate type-in fields.)

The completed form appears like the one below.

OK Cancel Defaults Help

Simulator spectre

Project Directory ~/.simulation

Host Mode  local  remote  distributed

Host

Remote Directory

5. In the Choosing Simulator/Directory/Host form, click *OK*.

### Copy and Edit Design Variables

1. In the Simulation window, choose *Variables – Copy From Cellview*.

The variables *sig\_bias*, *car\_ph*, and *bias\_end* appear in the *Design Variables* list box of the Simulation window.

Design Variables		
#	Name	Value
1	sig_bias	
2	car_ph	
3	bias_end	

2. In the Simulation window, choose *Variables – Edit*.

The Editing Design Variables form appears.

3. In the Editing Design Variables form, do the following, click *sig\_bias* in the *Table of Design Variables* list box.

4. Type `.01` in the *Value(Expr)* field.
5. Click *Change*.
6. Repeat steps one through three for the *car\_ph* and *bias\_end* variables. Type 0 for *Value(Expr)* for *car\_ph* and 1.8 for the *bias\_end* variable.

The completed form looks like the one below:

Selected Variable		Table of Design Variables		
Name	<input type="text" value="bias_end"/>	#	Name	Value
Value (Expr)	<input type="text" value="1.8"/>	1	sig_bias	10m
<input type="button" value="Add"/> <input type="button" value="Delete"/> <input type="button" value="Change"/> <input type="button" value="Next"/> <input type="button" value="Clear"/> <input type="button" value="Find"/>		2	car_ph	0
		3	bias_end	1.8
Cellview Variables		<input type="button" value="Copy From"/> <input type="button" value="Copy To"/>		

7. Click *OK*.

The new variable values display in the *Design Variables* list box in the Simulation window.

Design Variables		
#	Name	Value
1	sig_bias	10m
2	car_ph	0
3	bias_end	1.8

## Setting Up the PSS Analyses

1. Choose *Analyses* – *Choose* in the Simulation window.

The Choosing Analyses form appears.

2. In the Choosing Analyses form, click *PSS* for the analysis.

The form changes to display options needed for PSS. You perform PSS first because the periodic steady state solution must be determined before you can perform the small-signal PAC analysis.

3. *Beat Frequency* is highlighted by default. Type 1G in the *Beat Frequency* field.
4. Choose *Number of Harmonics* in the *Output Harmonics* cyclical field, and type 0 for the number of harmonics.

The top of the PSS Analysis area looks like the following.

**Periodic Steady State Analysis**

**Fundamental Tones**

#	Name	Expr	Value	Signal	SrcId
1	car_freq	1G	1G	Moderate	I240
3	101	1G	1G	Moderate	PORT1
2	1o2	1G	1G	Moderate	PORT2

Moderate

Clear/Add

Delete

Update From Schematic

**Beat Frequency**
 **Beat Period**

**Auto Calculate**

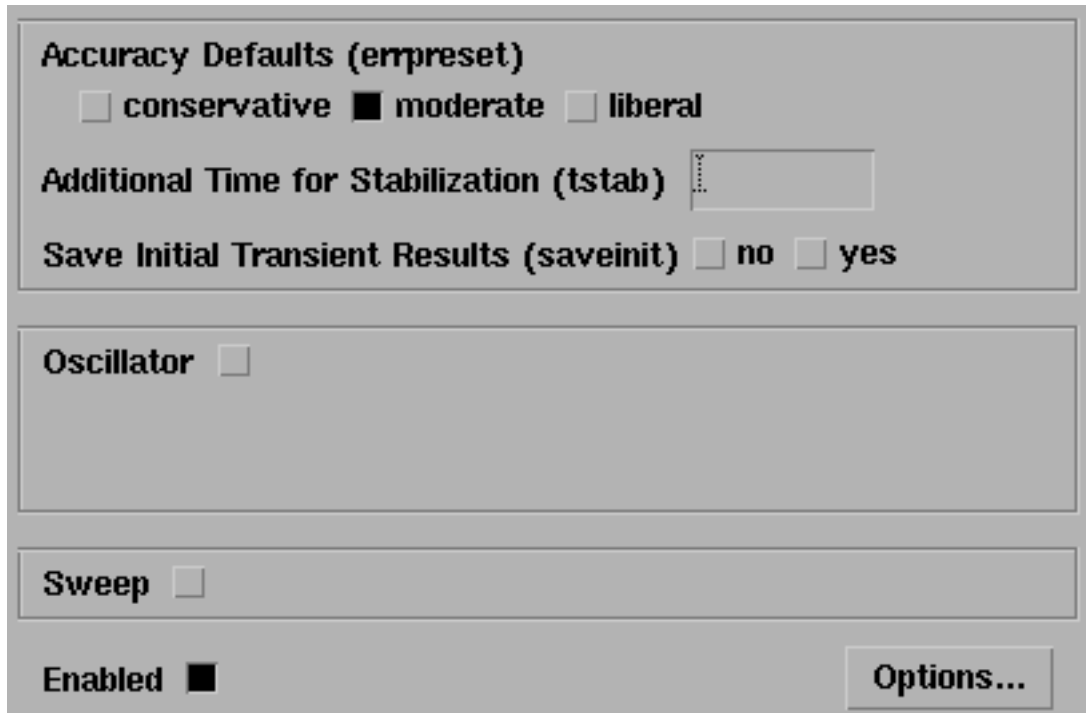
**Output harmonics**

Number of harmonics

5. Highlight *moderate* for the *Accuracy Defaults (errpreset)* value.

6. Highlight *Enabled*, if necessary.

The bottom of the PSS Analysis area looks like the following.



The screenshot shows a dialog box with the following controls:

- Accuracy Defaults (errpreset)**
  - conservative
  - moderate
  - liberal
- Additional Time for Stabilization (tstab)** [text input field]
- Save Initial Transient Results (saveinit)**  no  yes
- Oscillator**
- Sweep**
- Enabled**
- Options...** button

7. Click the PSS *Options* button.

The Periodic Steady State Options form appears.

8. In the PSS Options form, type 1 . 6G for the *maxacfreq* value.

The parameters are as follows.

**ACCURACY PARAMETERS**

**relref**       pointlocal    alllocal    sigglobal    allglobal

**Iteratio**     

**steadyratio**     

**maxacfreq**     

**maxperiods**     

**finitediff**       yes    refine    no

**highorder**       yes    no

**psratio**     

**maxorder**     

**fullpssvec**       yes    no

9. Click *OK* in the PSS Options form.

The PSS Options form closes and you return to the PSS Choosing Analyses form.

### Setting Up the PAC Analysis

1. In the Choosing Analyses form, choose *pac* for the *Analysis*.

The form changes to let you specify data for the *pac* analysis that follows the *pss* analysis.

2. Choose *Start – Stop* in the *Frequency Sweep Range (Hz)* cyclical field and type .01 and 500M for the *Start* and *Stop* values, respectively.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

---

The starting frequency must be very low but not zero. If the DC values of the PAC results are not captured, the K-model can incur large errors.

3. In the *Sweep Type* cyclical field, choose *Linear*.
4. Highlight *Number of Steps*, and type 100 in the *Number of Steps* field.
5. In the *Sidebands* cyclical field, choose *Maximum Sidebands*, and type 0 for the *Maximum Sidebands* value.
6. Highlight *Enabled*, if necessary, for the *pac* analysis.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

---

The completed *pac* section looks like this.

The image shows a dialog box titled "Periodic AC Analysis". At the top, "PSS Beat Frequency (Hz)" is set to "1G". Below this, there are two sections. The first section, "Sweep", includes a "Sweeptype" dropdown menu, a checkbox for "Sweep is Currently Absolute", and a "Frequency Sweep Range (Hz)" section with "Start-Stop" dropdown, "Start" field (.01), and "Stop" field (500M). The "Sweep Type" section has a "Linear" dropdown, radio buttons for "Step Size" and "Number of Steps" (selected), and a field for "Number of Steps" (100). There is also an "Add Specific Points" checkbox. The second section, "Sidebands", has a "Maximum sideband" dropdown and a field set to "d". At the bottom, there is an "Enabled" checkbox (checked) and an "Options..." button.

7. Click *OK* in the Choosing Analyses form.

## Running the Simulation

1. In the Simulation window, select *Simulation – Netlist and Run* to run the simulation.

The output log file appears and displays information about the simulation as it runs. Check the CIW for a message saying the simulation completed successfully.

2. In the Simulation window, choose *Results – Direct Plot – Main Form*.

The Waveform window and the Direct Plot form appear.

3. In the Direct Plot form, Choose *Replace* for *Plot Mode*.

4. Choose *pac* for *Analysis*.

5. Choose *Voltage* for *Function*.

The form changes to display information for the *Voltage Function*.

Choose *Magnitude* for *Modifier*.

6. Note that the *Select* cyclic field displays *Net*.

The completed form looks like this.

OK Cancel Help

Plot Mode  Append  Replace

Analysis

pss  pac

Function

Voltage  Current  
 IPN Curves

Select

Sweep

spectrum  sideband

Signal Level  peak  rms

Modifier

Magnitude  Phase  dB20  
 Real  Imaginary

Add To Outputs

> Select Net on schematic...

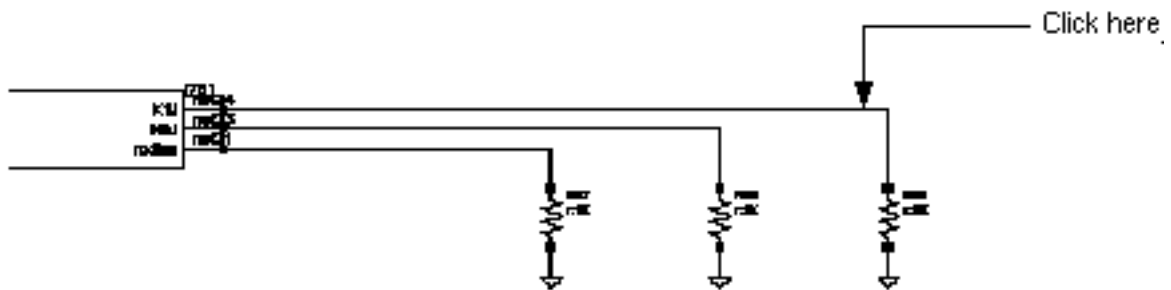
7. Follow the instruction at the bottom of the form,  
*Select Net on schematic...*

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

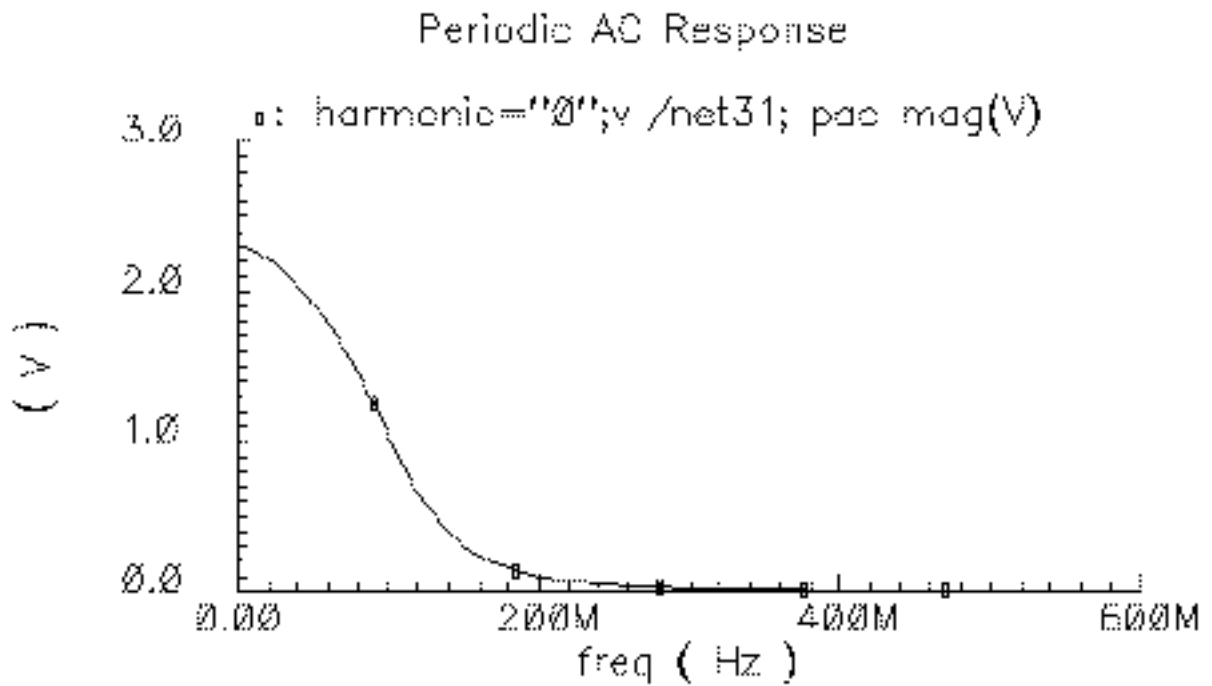
### Creating and Using Receiver K-Models

---

8. In the Schematic window, click the appropriate wire.



The plot in the Waveform window looks like the one below.



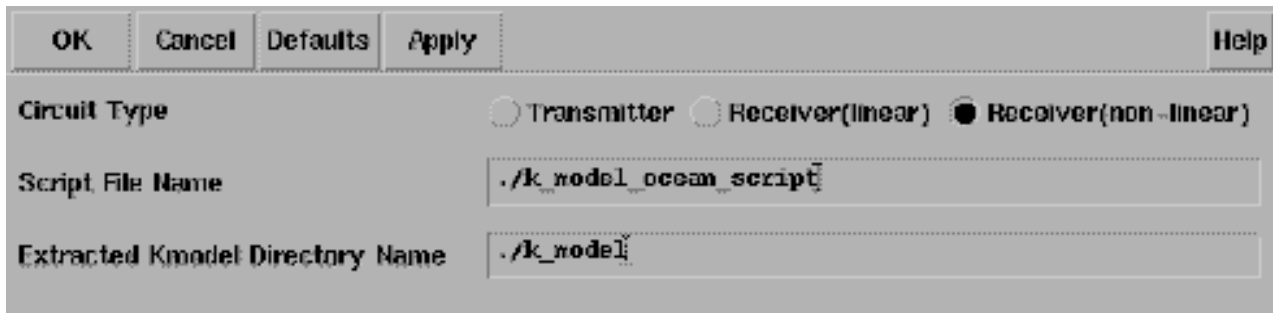
## Creating the K-Model

1. In the Simulation window, select *Tools – RF – Link to SPW*.

The RFIC Modeler for SPW form appears.

2. Choose *Receiver(non-linear)* for the *Circuit Type*.
3. Edit the *Script File Name* and the *Extracted Kmodel Directory Name*, if necessary.

The file name is the name of the Ocean script to run later, and the directory name is the output directory.



OK Cancel Defaults Apply Help

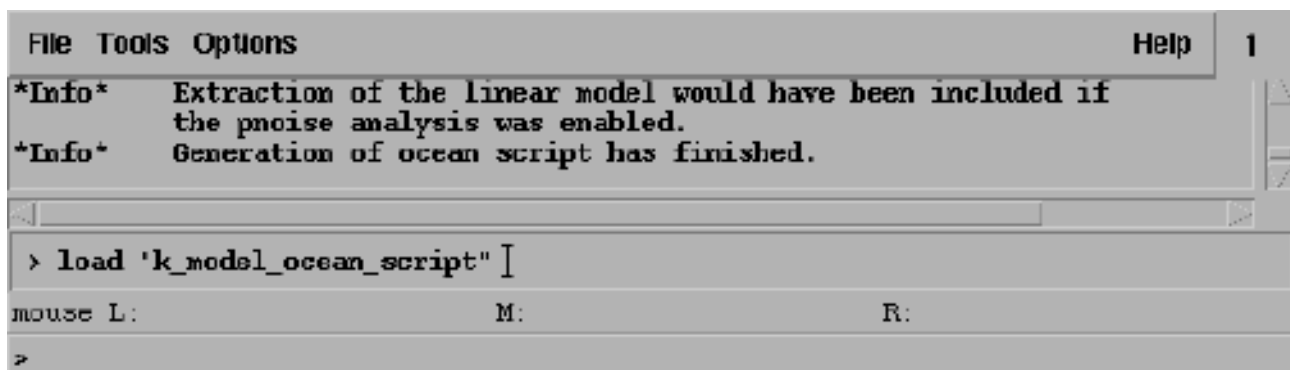
Circuit Type  Transmitter  Receiver(linear)  Receiver(non-linear)

Script File Name ./k\_model\_ocean\_script

Extracted Kmodel Directory Name ./k\_model

4. Click *OK*.
5. In the CIW, watch for a message that the ocean script is generated.
6. In the CIW, load the Ocean script file by typing in the following command. Be sure to include the double quotation marks around the script name.

```
load "k_Model_ocean_script"
```



File Tools Options Help 1

\*Info\* Extraction of the linear model would have been included if the pnoise analysis was enabled.

\*Info\* Generation of ocean script has finished.

> load 'k\_model\_ocean\_script' |

mouse L: M: R:

>

7. Press *Return*.

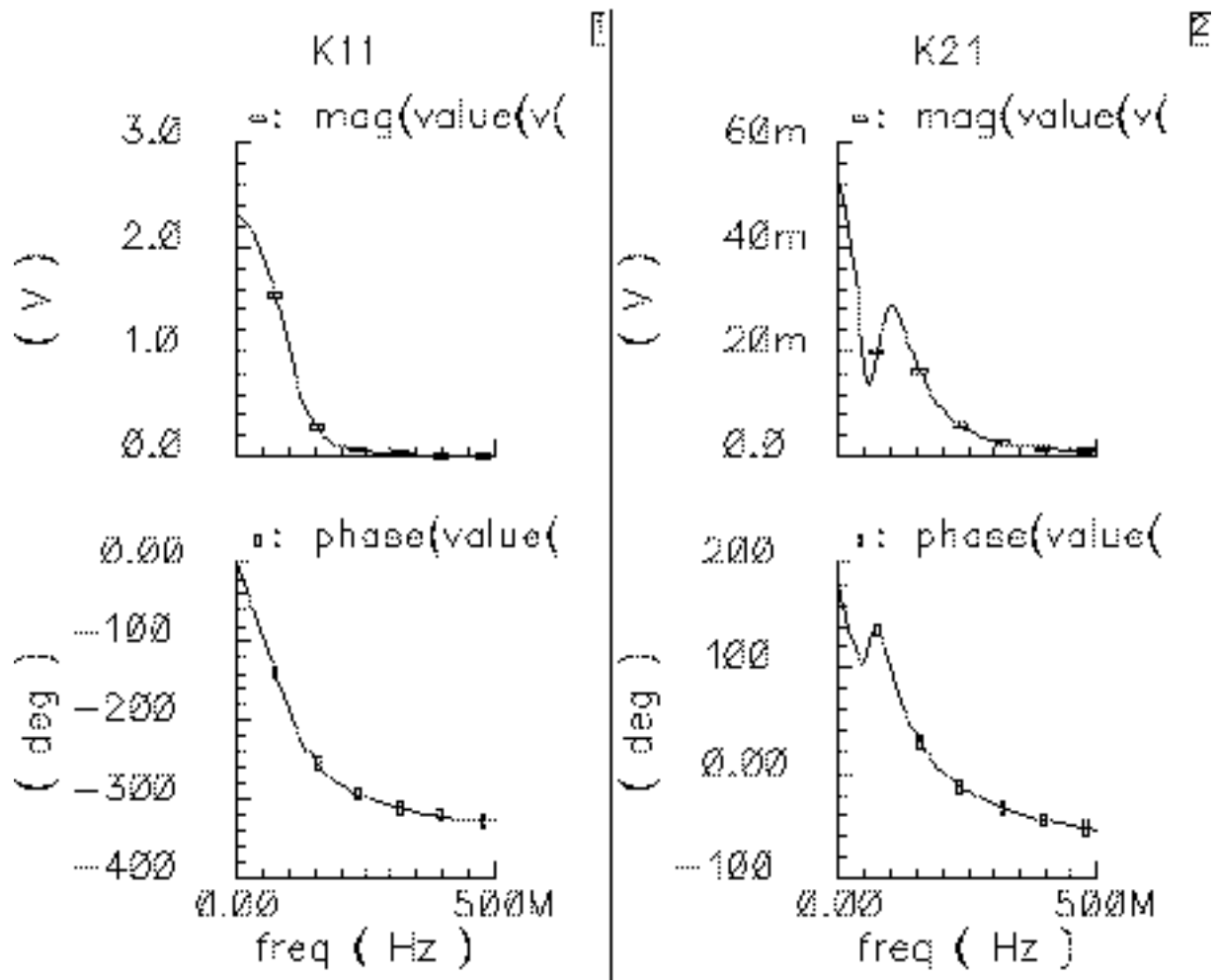
The script runs and generates the k-model.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

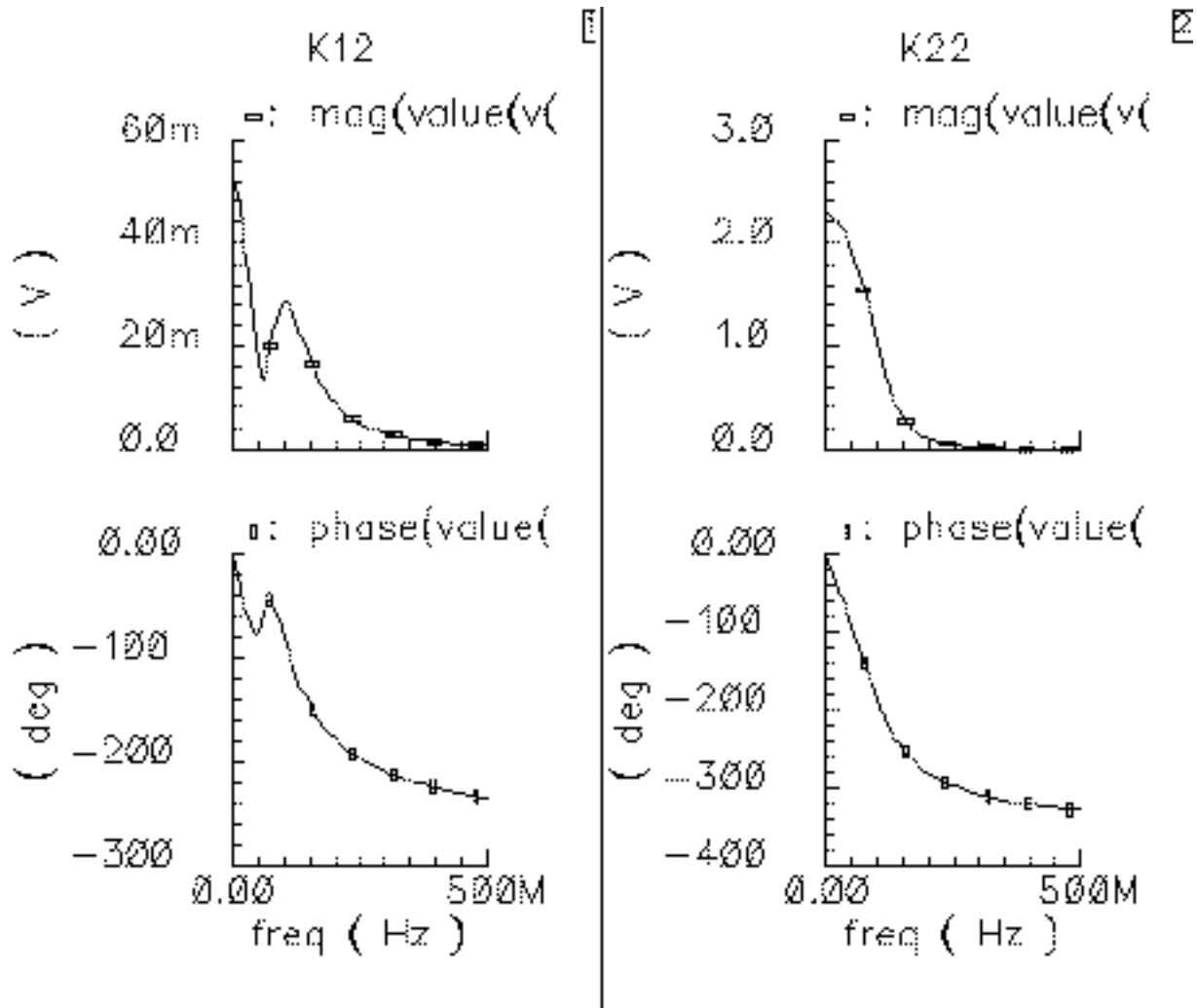
### Creating and Using Receiver K-Models

---

When the script finishes, it generates the following plots. The K11, K12, K21, and K22 plots are the I-to-I, Q-to-I, I-to-Q, and Q-to-Q transfer functions in the linear K-model.



The linear K-model is extracted at the *sig\_bias* signal level.



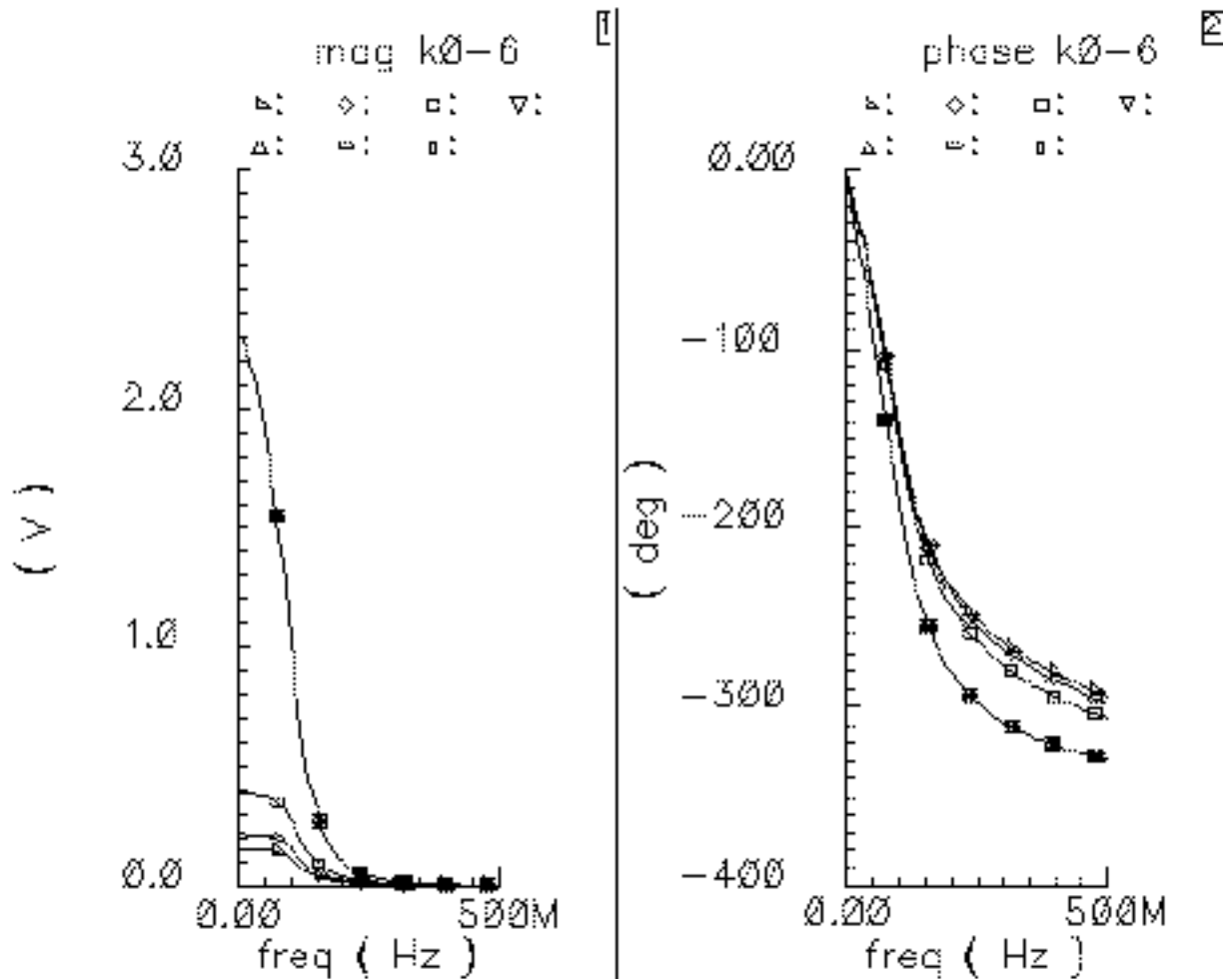
The overlaid curves are the K11 transfer functions. These transfer functions are extracted with I-input signal biases that range linearly from *sig\_bias* to *bias\_end*.

You can reconstruct these plots by loading the Ocean script in the file located at

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Creating and Using Receiver K-Models

<k\_model>/plotDir/oceanKmodQuickPlot



Examine the curves for the following features:

- **Narrow-band dynamics:** The K-model does not handle narrow-band dynamics well because it uses FIR (Finite Impulse Response) filters. The most likely narrow-band dynamics result from AGC loops and large DC blocking capacitors. They can introduce ultra-low frequency dynamics that require many FIR filter taps. Ultra-low frequency dynamics appear in the above curves as sharp dips near DC.
- **Non-negligible response at the maximum frequency:** Be sure the data goes to a high enough frequency. If the Alta SPW sample rate, divided by two, exceeds the maximum frequency in the data, Alta SPW linearly interpolates from the highest frequency data point down to zero at half the sample frequency. Because the receiver response probably decays exponentially rather than linearly, this interpolation can introduce errors.



Alta SPW ignores data beyond half the sample frequency. This does not cause a problem because an Alta SPW input signal probably would have no power beyond half the sampling frequency.

## More About the K-Model

Like any behavioral model, the K-model has limitations and requirements. The assumptions on which the K-model is based cause the basic limitations. Other limitations are caused by practical considerations. There are also requirements that are specific to the Alta SPW environment. Finally, this section discusses some limitations of the K-model's ability to model out-of-band blockers.

### Limitations and requirements resulting from linear K-model assumptions

The linear K-model makes the following two assumptions beyond the small signal assumption:

- Unless the Spectre RF model includes detailed A/D converter models, the extracted linear K-model assumes that noise at the base-band outputs does not change significantly over the A/D aperture time.

The first K-model assumption involves noise and A/D conversion. The linear model includes noise generated within the front end. "Aperture time" is the time the A/D converter requires just to measure a voltage, not to measure and hold it. The assumption is that the noise is constant over the aperture time. This implies the noise is band limited. However, as the following discussion explains, the noise bandwidth must also not be too small.

Note that the the linear model does not account for phase noise. However, if one knows the power spectral density of the phase noise one can easily follow the K-model with a complex multiplier that shifts the phase of the K-model output. A filter driven by white Gaussian noise would generate the random phase shift.

- The two equivalent output noise sources, one for the I output and one for the Q output, are uncorrelated.

Frequently, much of the noisy circuitry is common to both outputs. Despite the common circuitry, the K-model assumes the two output noise signals are independent. This is a good assumption if the noise is white over the frequencies of interest. In summary, the noise must be white with respect to filters preceding the A/D conversion but band-limited with respect to  $1/(\text{aperture time})$ .

Why isn't there just one K-model, a noisy non-linear K-model instead of a noisy linear K-model and a noise-less non-linear K-model? Putting small signal noise in the non-linear

K-model would suggest that the model's noise varied with input power. It does not. Furthermore, noise analysis is usually of interest in bit error rate simulations. Such simulations are usually long and the desired signal is weak. The linear K-model suffices for weak desired signals and runs faster than the non-linear K-model.

### **Limitations and requirements resulting from non-linear K-model assumptions**

The K-model requires two basic assumptions:

- Small-signal components of the base-band input can be wide-band, but large components must fall within the receiver's bandwidth.

The signal of interest is usually within the receiver's bandwidth, and interferers usually lie outside this bandwidth. However, if the receiver has a band selection filter immediately after the antenna, and that filter can be modeled apart from the K-model, this first assumption is not a severe restriction. This is true because the filter makes the interferer small by the time it reaches the K-model.

- Non-linear behavior depends only on the input radius. Non-linear PM/PM and PM/AM effects are negligible.

If the input signal is

$$rf(t) = a(t)\cos(\omega t) - b(t)\sin(\omega t)$$

the base-band signal is

$$a(t) + jb(t)$$

a trajectory in the complex plane. This is the rectangular representation.

The polar representation is in terms of radius and angle. The radius is

$$\text{Sqrt}[a(t)a(t) + b(t)b(t)]$$

The angle is

$$\text{ArcTan}[b(t)/a(t)]$$

### **Requirements resulting from the Alta SPW environment**

Both K-models are part of Alta SPW, the Cierito Signal Processing Worksystem. Alta SPW is a DSP design tool. This environment creates the following two consequences you must remember:

- Model extraction requires base-band input and output ports.

Because it is a base-band tool, Alta SPW operates on the information impressed on the RF carrier rather than the instantaneous value of the RF signal. Each of two carrier phases, sine and cosine, carry information. Because Alta SPW represents base-band

signals as complex numbers, the modeled RF circuit must have base-band input and output ports.

The input test jig creates the input port. The input test jig mixes the base-band input signal up to the appropriate carrier frequency, thereby synthesizing an antenna signal from a base-band input. Changing the phase of the carrier by 90 degrees switches the base-band input between the “in-phase” and “quadrature” input ports. The Ocean extraction script changes carrier phase automatically when it extracts a K-model.

If the circuitry does not contain two base-band output ports, you must synthesize them. You can synthesize them easily using ideal behavioral elements. For example, to build a K-model of only the low noise amplifier, you add ideal downconverters to the Spectre RF model and extract a K-model of the combination.

You can partition a receiver into smaller K-models only if there is no non-linear interstage loading at the break points and if the carrier fundamental is the only important quantity.

The input and output test jigs also tell the extraction script where to excite the Spectre RF model and what quantities to measure.

- The Spectre RF noise analysis must extend far enough to capture all important features, including those beyond half the Alta SPW sampling frequency.

This is true because the power spectral density suffers aliasing in the DSP environment. The only data available for DSP is for discrete, evenly-spaced time-points. This restriction imposes a constraint on possible input signals that is usually met within the design. Specified input signals must not have significant power beyond half the sampling frequency.

The channel and RF front end are cascaded physical entities with no sampler between them. But because they are modeled in Alta SPW as separate blocks, Alta SPW implicitly assumes an ideal sampler lies between the channel and front end. That assumption is only valid for base-band signals that have no significant spectral components near or beyond half the sampling frequency. If that assumption were not valid the overall design would have far more serious problems than K-model accuracy.

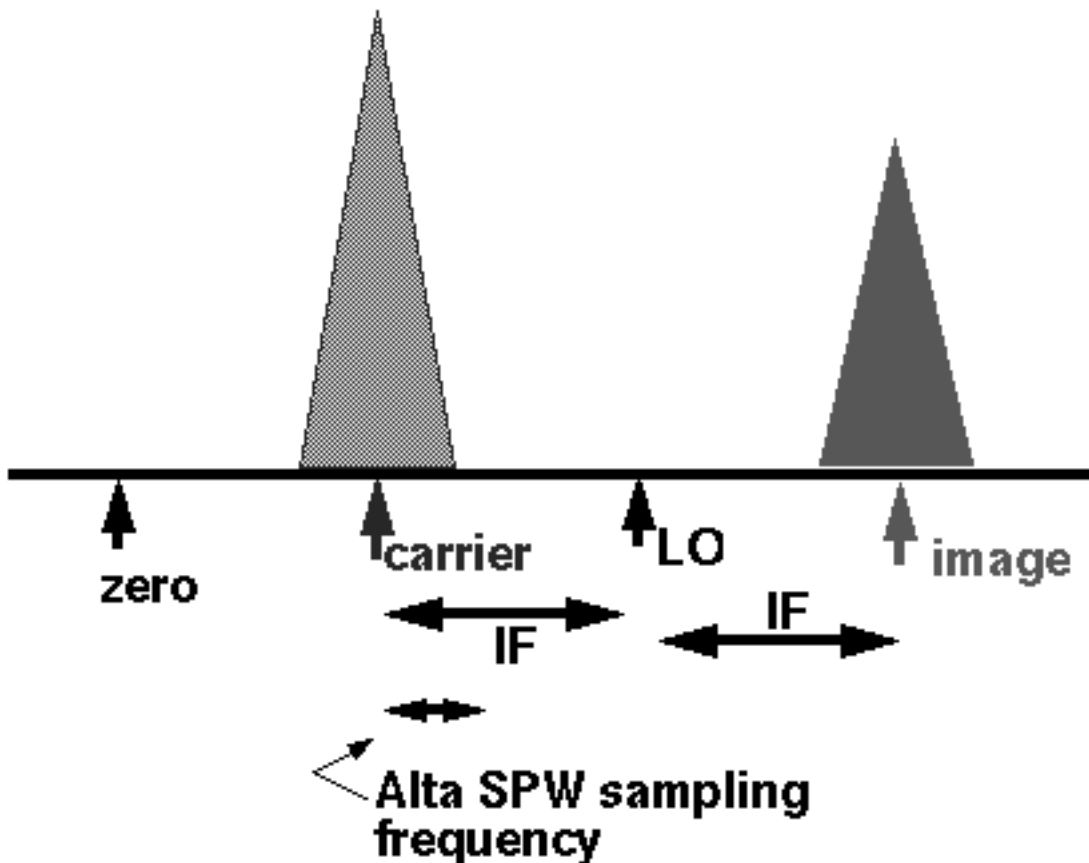
Because the noise is not a signal you specify, it need not meet that constraint. Noise power that is well beyond half the sampling frequency can alias down to the band of interest. If the noise is characterized to a high-enough frequency, the K-model automatically aliases it.

## **Practical limitations of the K-model**

- **Image rejection**

Although the K-model theoretically models image rejection, such modeling is not practical for the Alta SPW implementation. [Figure 5-1](#) on page 324 shows how the carrier, local oscillator, and image frequencies can be related. The Alta SPW sampling frequency is probably approximately the bandwidth of the RF receiver, shown in blue. To simulate dynamics at the image frequency, the Alta SPW sampling frequency must span twice the difference between the carrier frequency and the image frequency. Such a high sampling frequency requires an impractical number of FIR filter taps to model the receiver dynamics. Simulation time is almost as lengthy as with an unsuppressed carrier simulation.

**Figure 5-1 Carrier, Local Oscillator, and Image Frequency Relationships for the Alta SPW Implementation**



#### ■ Ultra-low frequency dynamics

Ultra-low frequency effects such as those caused by PLLs, AGC loops, and large DC blocking capacitors again require many FIR filter taps. However, behavioral phaselock loop models or

AGC loop models can precede the K-model in Alta SPW to approximate these low frequency effects.

### Limitations in modeling large out-of-band blockers

The Figure 5-2 illustrates a basic K-model limitation. The basic K-model does not accurately simulate the effects of large out-of-band blockers. However, if the blocker is narrow-band enough to be considered deterministic (ideally, a single tone), you can extract a model that accurately simulates blocker-induced gain compression by extracting a K-model with the blocker present.

Figure 5-2 K Model with Out of Band Blockers

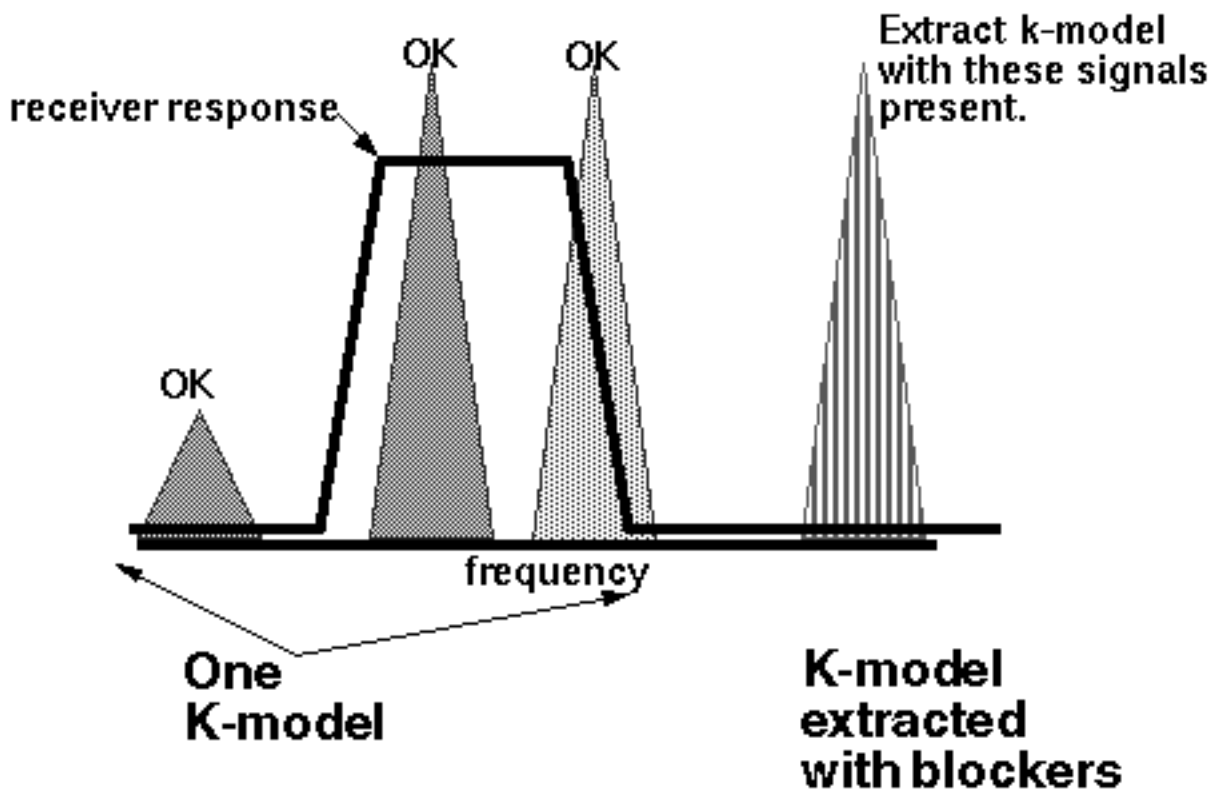
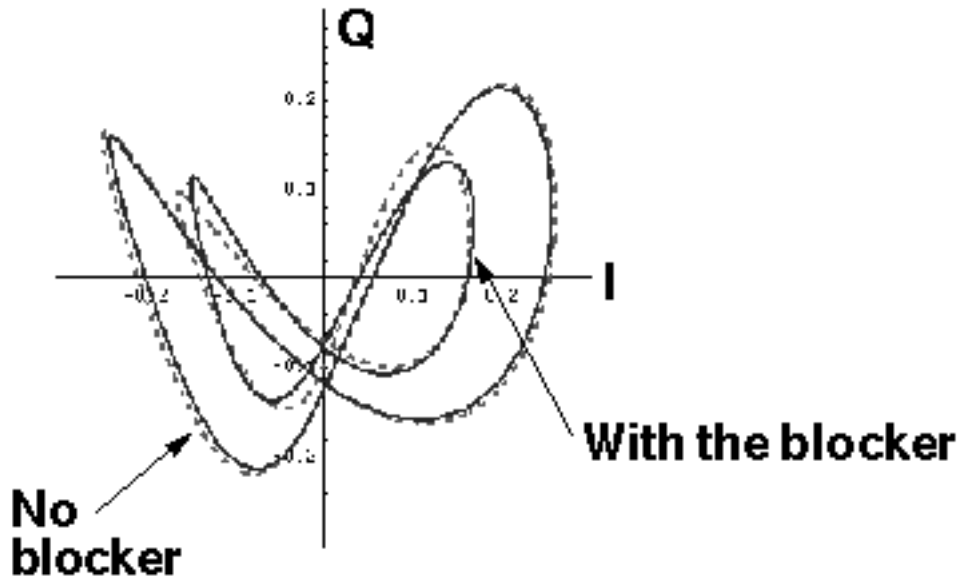


Figure 5-3 compares receiver outputs computed using a transistor-level model in Spectre RF with outputs that result from using K-models in Alta SPW. One K-model includes the blocker. The other does not.

**Figure 5-3 Results Calibrated with a Stronger Signal**



The I-input signal is in-band and the Q-input signal is on the band edge. The blocker is more than two decades out of band. The blocker's amplitude is more than 10 times the in-band signal level that saturates the receiver.

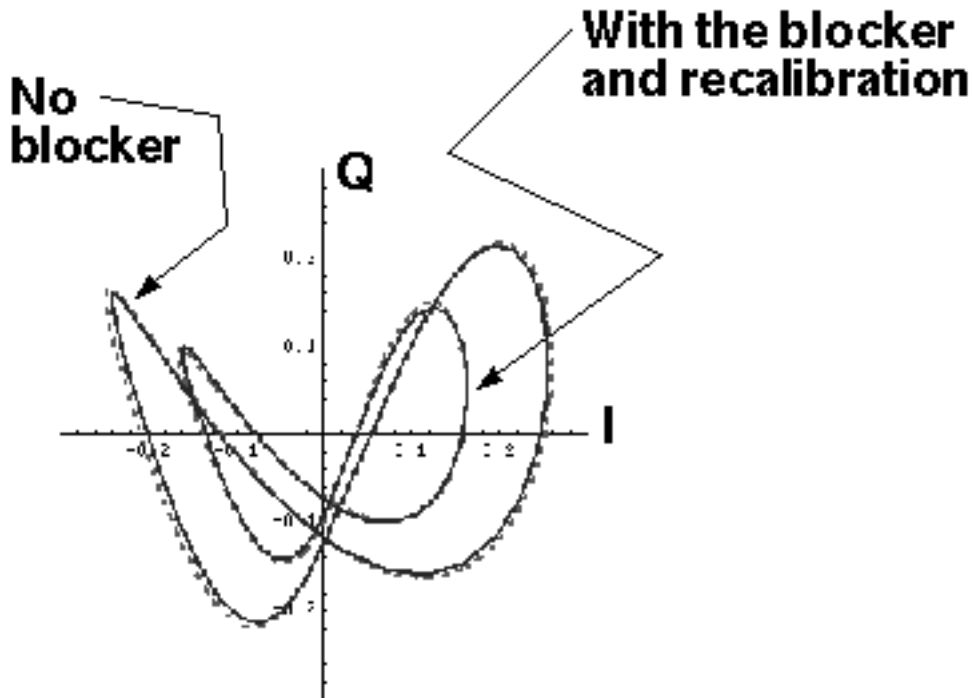
The red dotted trajectory shows direct Spectre RF simulation results. The solid blue line shows results from the K-model simulation. The larger trajectories are without the blocker.

The K-model extracted without the blocker matches device-level simulation quite well. The K-model extracted with the blocker present also matches the corresponding device-level simulation well except for a fixed rotational error. Rotating the blue trajectory by 7 degrees puts it nearly on top of the trajectory computed in Spectre RF. Such a fixed rotational error is usually irrelevant to the DSP section because the equalizer hides the error, and a fixed rotation does not affect the spectrum.

However, you can reduce the error. The K-model internally calibrates for rotational error. A blocker introduces an offset that creates calibration error. Alta SPW can modify the K-model to calibrate at a stronger input signal so that the offset has less effect.

Figure [5-4](#) shows the results with the K-model calibrated at the stronger signal.

Figure 5-4 K Model Calibrated at the Stronger Signal



## K-model data files

This section describes the K-model data files.

### I\_data.ascsig and Q\_data.ascsig

These files characterize the static behavior of the *I* and *Q* outputs, respectively, as the input bias increases along the input *I*-axis. The format is given below:

```
data
$
signal type = double
vector type = interlaced
vector length = 14
number of vectors = 2
number of signal points = 14
sampling frequency = 1
starting time = 0
$
0.001    0.002300
0.3008333  0.692013
0.6006667  1.381728
0.9005    2.071483
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

---

```
1.200333    2.568851
1.500167    2.655311
1.8         2.710871
```

#### **dck**

This file contains the lowest frequency gains of the *K11* and *K21* transfer functions. The format is shown below:

```
2.299950
-0.051965
```

#### **sig\_biases**

This file lists the input signal biases. The format is shown below:

```
0.001
0.3008333
0.6006667
0.9005
1.200333
1.500167
1.8
```

#### **psd\_I.ascsig and psd\_Q.ascsig**

These files contain the power spectral densities of the noise observed at the baseband *I* and *Q* outputs. The first column is frequency, and the second column is power spectral density in volts/Sqrt(Hz). The format is given below:

```
# wavew4sli1() from 10e-3 to 25e6
#          wavew4sli1()

10e-3      1.334e-6
250e3      943.1e-9
500e3      943.1e-9
750e3      943e-9
1e6        942.7e-9
1.25e6     942.4e-9
1.5e6      941.8e-9
1.75e6     940.9e-9
2e6        939.6e-9
2.25e6     938e-9
2.5e6      935.9e-9
```

#### **K11.ascsig, K21.ascsig, K12.ascsig, K22.ascsig, k11\_1 through k11\_6, and k21\_1 through k21\_6**

The first four files contain the *I-I*, *I-Q*, *Q-I*, and *Q-Q* transfer functions at the smallest input bias. The format is identical to the *k21\_j* and *k11\_j* transfer functions.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Receiver K-Models

---

The *k11\_1* through *k11\_6* files contain the transfer functions from the *I*-input to the *I*-output as the input bias sweeps from the *sig\_bias* level to the *bias\_end* level. The *k21\_1* through *k21\_6* files contain the transfer functions from the *I*-input to the *Q*-output for the same input bias levels. The format of all of these files is the same. The first column is the frequency, the second is the magnitude in volts/volt, and the third is the phase in degrees. You must keep exactly three lines of header. The format is shown below:

freq (Hz)	mag(harmonic(v	phase(harmonic
10e-3	2.300	-18.860e-9
1e6	2.300	-1.886
2e6	2.299	-3.771
3e6	2.298	-5.656
4e6	2.297	-7.540
5e6	2.295	-9.424
6e6	2.293	-11.310
7e6	2.290	-13.190

The rest of the files are obsolete binary files. They formerly supported the Alta SPW beta version of the K-model. The released version uses only the ascii files.

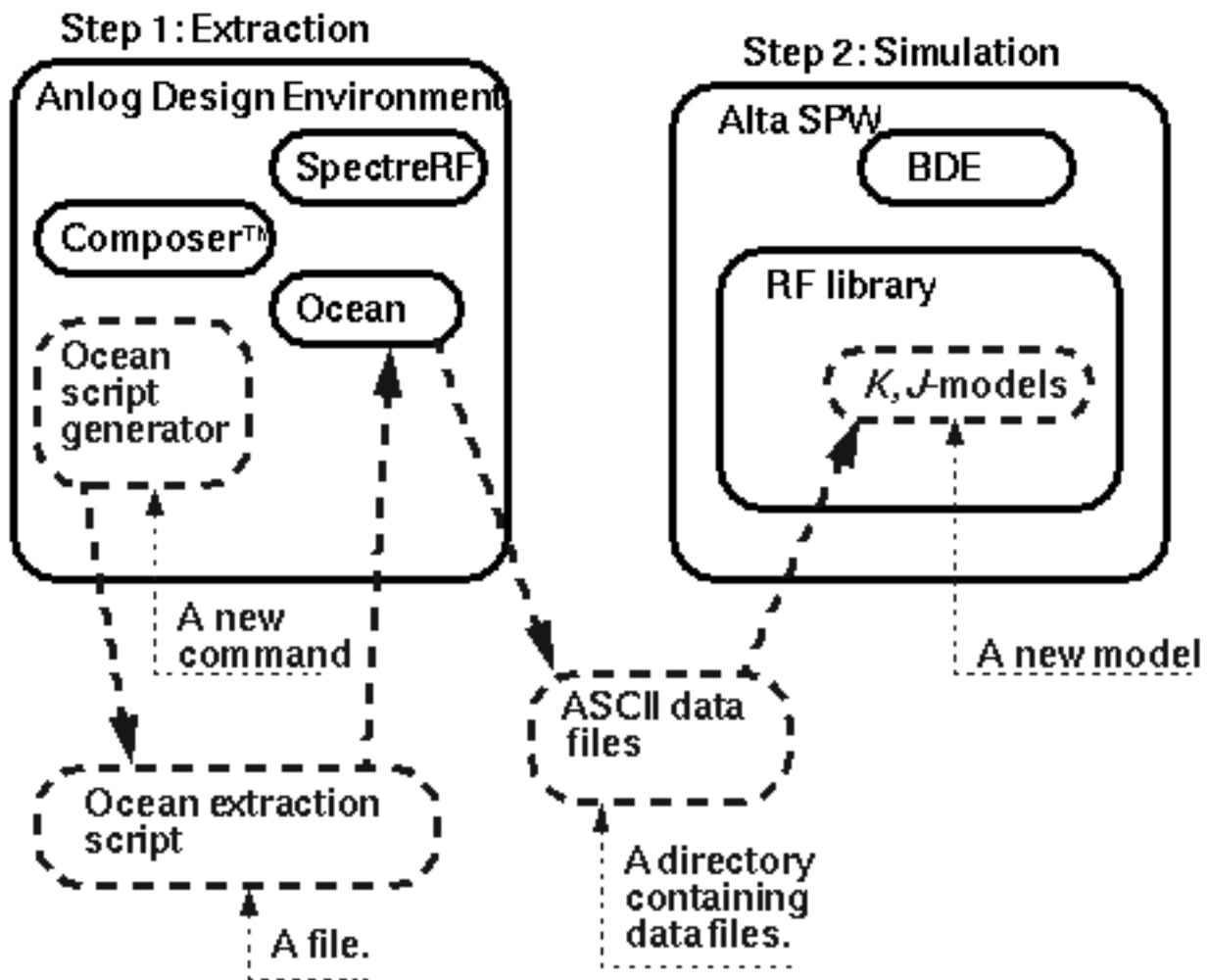
**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Creating and Using Receiver K-Models

---

## Creating and Using Transmitter J-Models

The J-model, also known as a transmitter K-model, links Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) and Alta™ SPW as shown in Figure .

### Task Flow of the J-Model Creation and Use



The J-model, also known as a transmitter K-model, has two primary uses.

- The J-model facilitates Adjacent Channel Power Ratio (ACPR) estimates
- The J-model imports transmitter impairments into the Alta™ SPW environment.

You can extract a J-model and then feed it directly into SPW by instantiating a J-model from the Alta SPW library and specifying the directory containing the extracted J-model data. An Alta SPW user can use the J-model to examine how a direct conversion transmitter affects the end-to-end bit error rate.

Only the model extraction part of the SPW flow is described in this chapter. The analog design environment `rfLib` includes Verilog-A versions of the J-model that read the same data files. The `j_mod_extraction_example` simulation procedures in this chapter describe how to use the Verilog-A version of the J-model to compute ACPR in the analog design environment. The last part of this chapter briefly describes the structure of the J-model and files created by the extraction script.

The example in this chapter is highly ideal but is still a good example because it simulates quickly and contains the main distortion mechanisms captured by the J-model.

 *Important*

You can also use the ACPR wizard described in [“Measuring ACPR and PSD”](#) on page 408 to measure ACPR.

## Procedures for Simulating `j_mod_extraction_example`

Follow the steps below to simulate the `j_mod_extraction_example`:

1. Copy `j_mod_extraction_example` from the Cadence `rfExamples` directory into a local directory.

Be sure the path to the local directory is defined in your `cds.lib` file.

2. Choose *File – Open* in the CIW.

The Open File form appears. Filling in the Open File form lets you open the schematic.

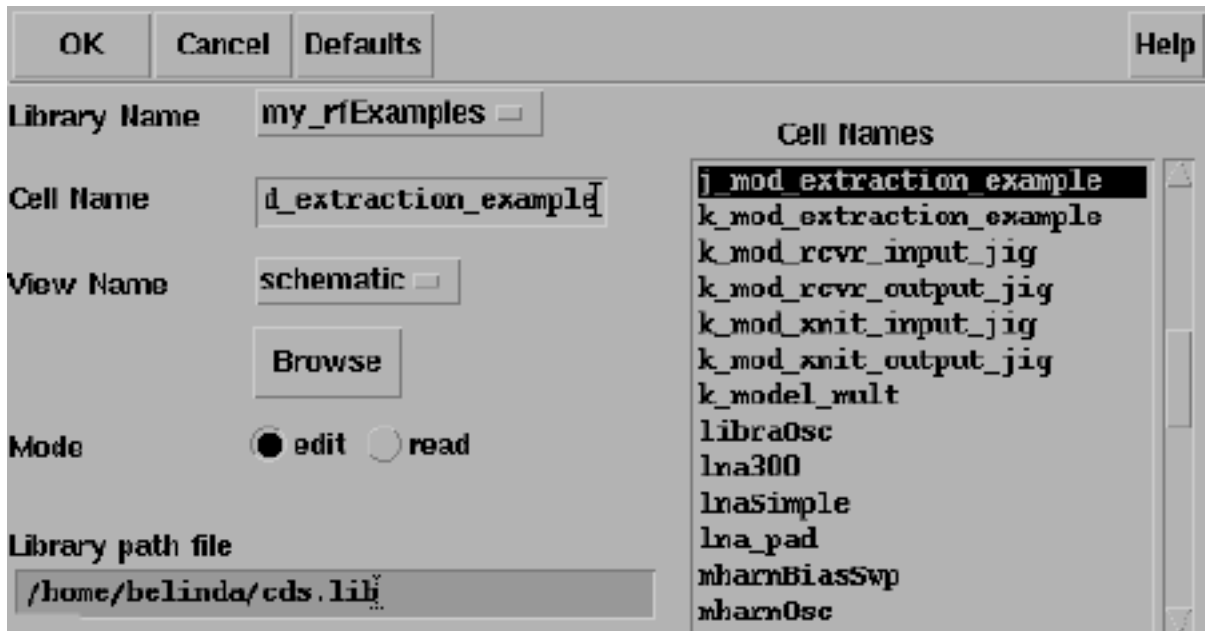
3. In the Open File form, choose `<Path_to_local_copy>` for *Library Name*.
4. Choose *schematic* for *View Name*.
5. In the *Cell Names* list box, highlight `j_mod_extraction_example`.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

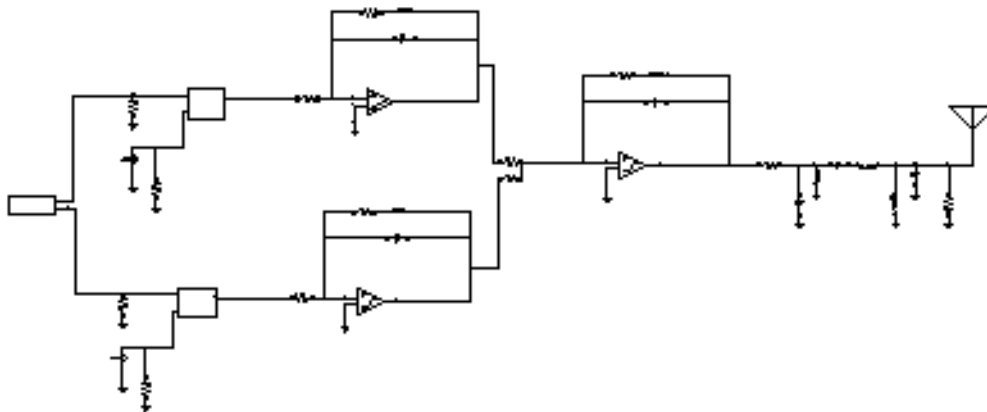
### Creating and Using Transmitter J-Models

`j_mod_extraction_example` appears in the *Cell Name* field. (The j-model is another name for a transmitter k-model.)

6. Highlight *edit* to choose the *Mode*.



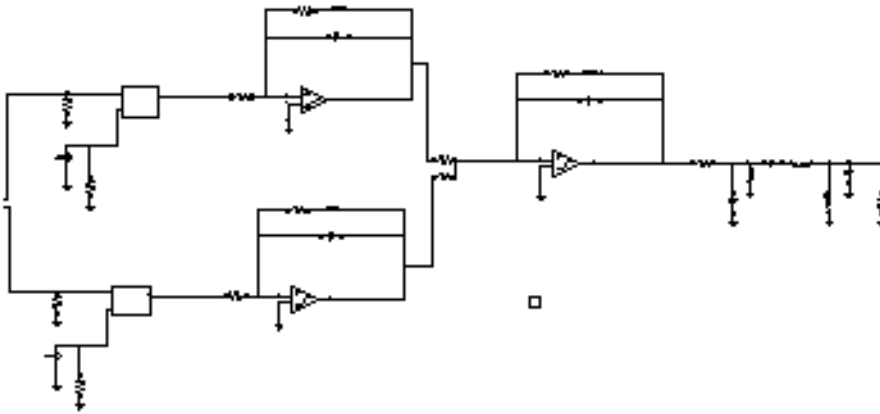
7. Click *OK* in the Open File form.
8. The Schematic window opens and displays the schematic.



## Optional: Setting Up the Input and Output Jigs

If you want to see the whole procedure for setting up the J-model, including the editing of the schematic, delete the input and output test jigs at the far left and right of the schematic. The edited schematic now looks like the one below.

If you do not want to edit the schematic but only want to set up and run the simulation, skip to the Analysis Setup section.



From the Schematic window, do the following:

1. Choose *Add – Instance*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

---

The Add Instance form appears.

The Add Instance form is a dialog box with a title bar containing buttons for 'Hide', 'Cancel', 'Defaults', and 'Help'. The main area contains several input fields and buttons:

- Library:** A text field containing 'my\_rfExamples' and a 'Browse' button to its right.
- Cell:** An empty text field.
- View:** An empty text field.
- Names:** An empty text field.
- Array:** A section with 'Rows' and 'Columns' labels, each followed by a text field containing the number '1'.
- Buttons:** Three buttons at the bottom: 'Rotate', 'Sideways', and 'Upside Down'.

2. In the Add Instance form, click *Browse*.

The Library Browser – Add Instance form appears.

The Library Browser – Add Instance form is a dialog box with a title bar containing a checkbox labeled 'Show Categories'. The main area is divided into three columns:

- Library:** A list box containing the following items: 'my\_rfExamples', 'basic', 'cdsDefTechLib', 'my\_rfExamples', 'passiveLib', and 'rfExamples'. The 'my\_rfExamples' item is highlighted.
- Cell:** A list box containing the following items: 'k\_mod\_xmit\_input\_jig', 'k\_mod\_xmit\_input\_jig', 'k\_mod\_xmit\_output\_jig', 'k\_model\_mult', and 'libraosc'. The 'k\_mod\_xmit\_input\_jig' item is highlighted.
- View:** A list box containing the following items: 'symbol' and 'symbol'.

At the bottom of the form are two buttons: 'Close' and 'Filters...'.

In the Library Browser – Add Instance form, do the following:

1. Choose `my_rfExamples` for the *Library*. (Choose the name of the editable copy of the `rfExamples` library that you created.)

The form changes to let you choose a cell.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

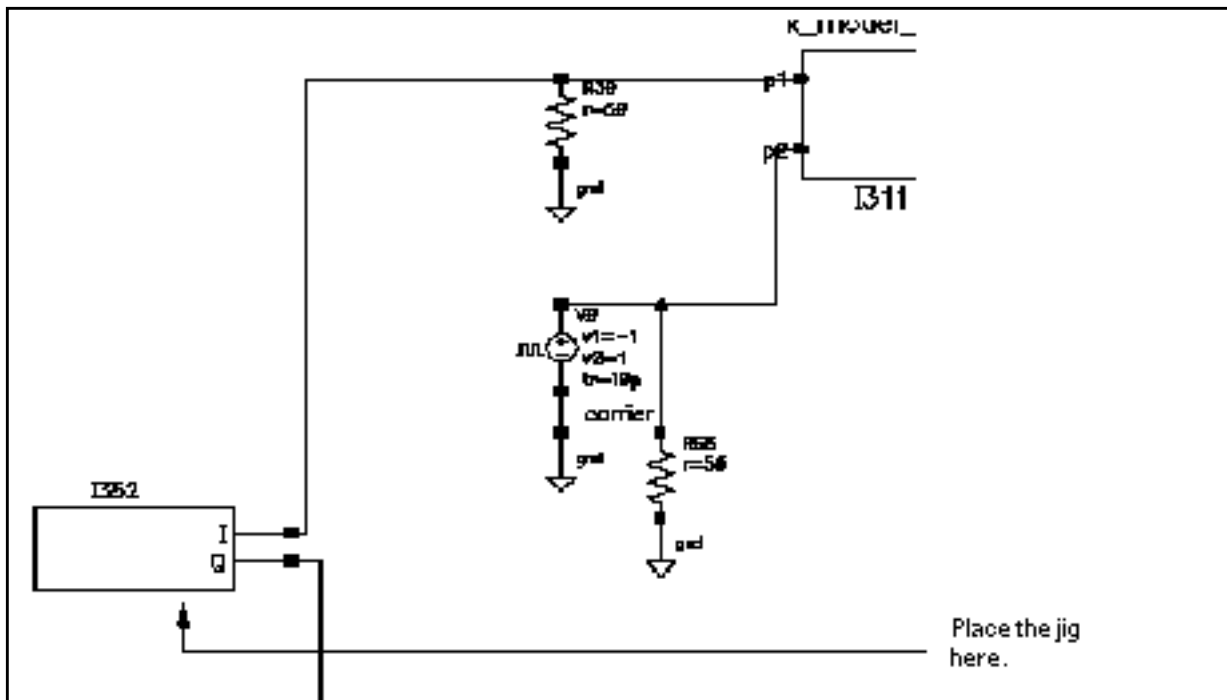
2. Choose `k_mod_xmit_input_jig` for the *Cell*.

The form changes to let you choose a *View*.

3. Choose `symbol` for the *View*.

When you place the cursor back in the Schematic window, an instance of `k_mod_xmit_input_jig` is attached to it.

4. Place the input jig at the appropriate place in the schematic.

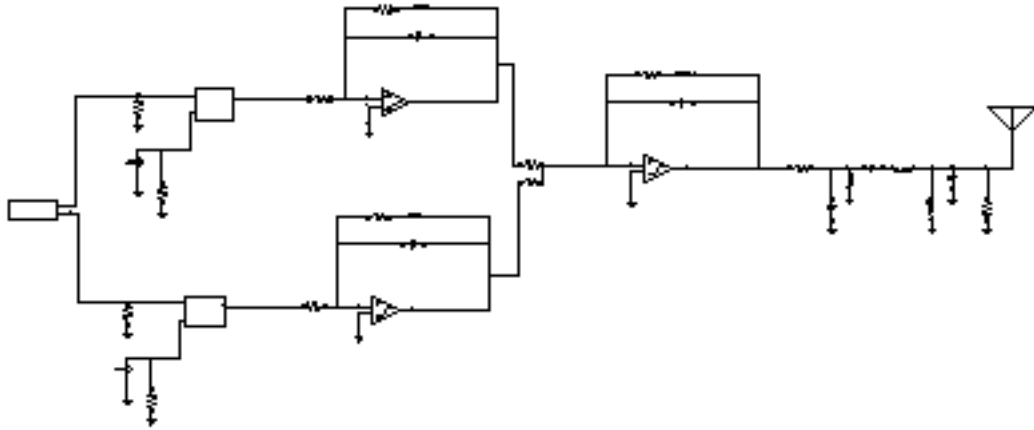


If you need assistance with the procedures for editing a schematic, see the *Virtuoso® Schematic Composer™ User Guide*.

5. In the Schematic window, again choose *Add – Instance*.
6. Now repeat the steps you used to add the input jig to the schematic and add the output jig `k_mod_xmit_output_jig`.



The edited schematic looks like this.



7. In the Schematic window, click the input jig to select it.
8. Choose *Edit – Properties – Objects* in the Schematic window menu choices.  
The Edit Object Properties form appears.
9. In the Edit Object Properties form, type 1M in the *Base band frequency* field and click *OK*.
10. In the Schematic window, choose *Design – Check and Save*.

## Analysis Setup

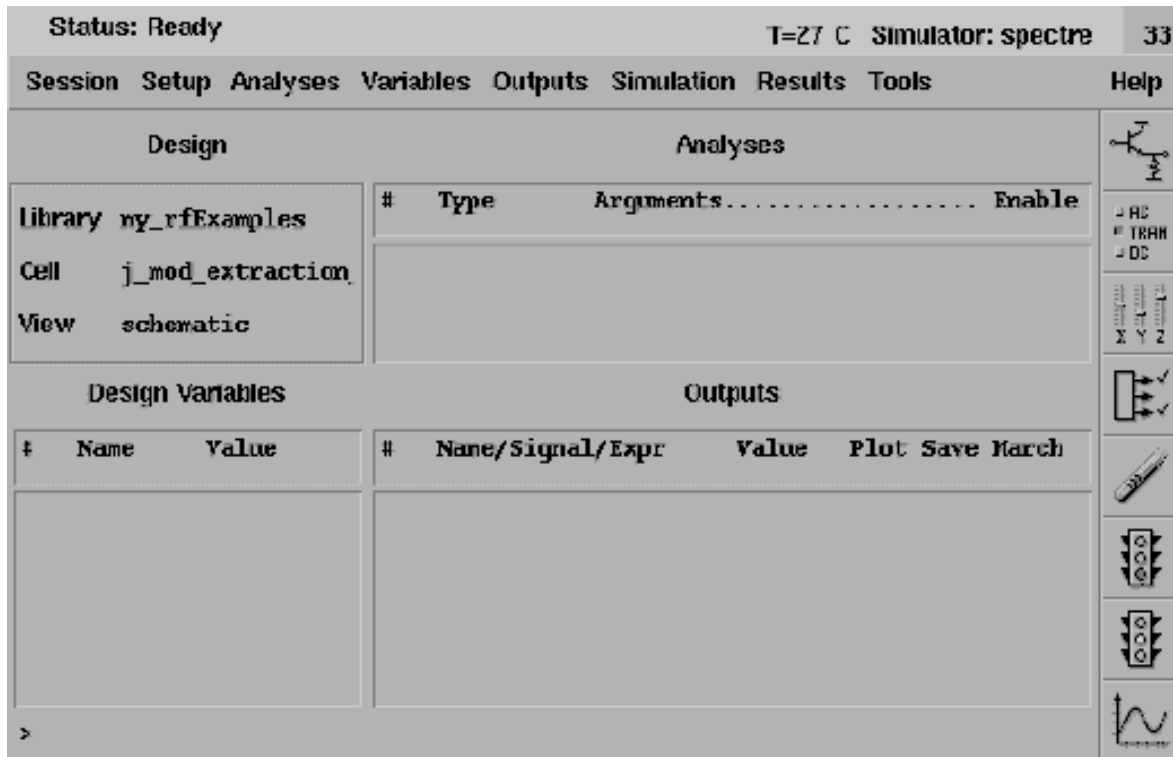
1. Choose *Tools – Analog Environment* in the Schematic window.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

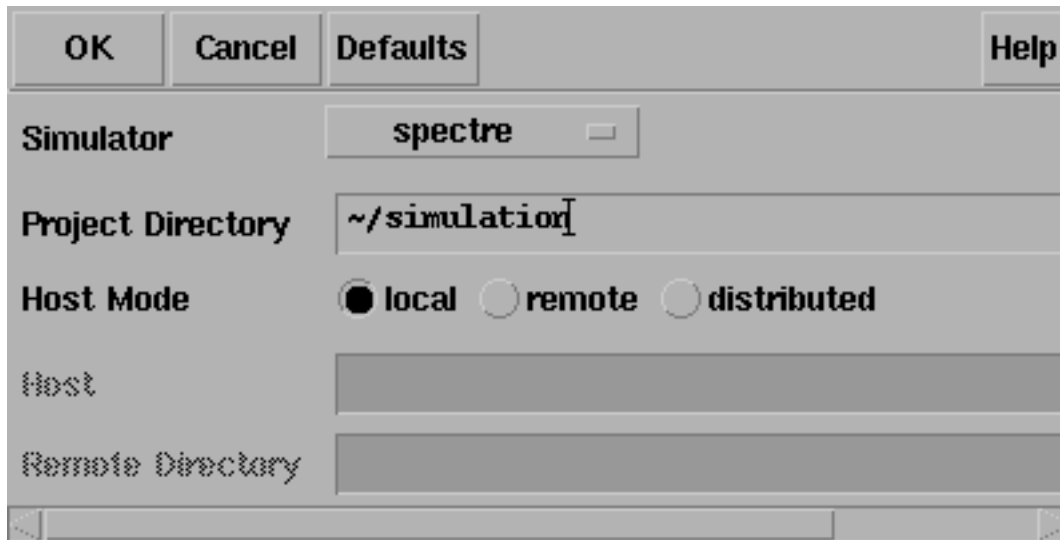
---

- The Simulation window appears.



- Choose *Setup – Simulator/Directory/Host* in the Simulation window.  
The Choosing Simulator/Directory/Host form appears.
- In the Choosing Simulator/Directory/Host form, choose *spectre* for *Simulator*.
- Type in the name of the project directory, if necessary.
- Highlight *local*. (If you highlight *remote*, specify the *Host Mode*. For remote simulation, type in the name of the host machine and the remote directory in the appropriate fields.)

The completed form appears like the one below.



The screenshot shows a dialog box with the following fields and options:

- Simulator:** spectre
- Project Directory:** ~/simulation
- Host Mode:** local (selected), remote, distributed
- Host:** (empty text field)
- Remote Directory:** (empty text field)

7. In the Choosing Simulator/Directory/Host form, click *OK*.

### Copy and Edit Design Variables

1. In the Simulation window, choose *Variables – Copy From Cellview*.

The variable `radius` appears in the *Design Variables* list box of the Simulation window.

Design Variables		
#	Name	Value
1	radius	

2. Choose *Variables – Edit*.

The Editing Design Variables form appears.

3. In the Editing Design Variables form, in the Table of Design Variables list box, click *radius*.
4. Type `1.5` in the *Value(Expr)* field.
5. Click *Apply*.

The completed form looks like the one below:

Selected Variable		Table of Design Variables	
Name:	radius	#	Name Value
Value (Expr):	1.5	1	radius 1.5
<input type="button" value="Add"/> <input type="button" value="Delete"/> <input type="button" value="Change"/> <input type="button" value="Next"/> <input type="button" value="Clear"/> <input type="button" value="Find"/>			
Cellview Variables		<input type="button" value="Copy From"/> <input type="button" value="Copy To"/>	

6. Click *OK*.

The new variable value displays in the Simulation window.

Design Variables		
#	Name	Value
1	radius	1.5

### Setting Up the Analyses

1. Choose *Analyses – Choose* in the Simulation window.

The Choosing Analyses form appears.

2. In the Choosing Analyses form, click *qpss* for the analysis.

The form changes to display options needed for QPSS. In the Choosing Analyses form, do the following:

3. In the *Fundamental Tones* list box, choose one of the tones labeled *carrier*.

4. In the fields immediately below the *Fundamental Tones* list box, do the following:

- a. Choose *Large* in the *Signal* cyclic field.
- b. Type 1 for the *Harms* field value.
- c. Choose `bb_freq`.
- d. Type 9 for the *Harms* field value for `bb_freq`.

Because this is a down converting situation, a *Harms* value of 1 for the *Large* tone is sufficient here. For the *Moderate* tone, the higher *Harms* value of 9 guarantees higher order intermodulation terms. However, for *Moderate* tones, increasing the *Harms* value increases the simulation run time.

5. Highlight *moderate* for the *Accuracy Defaults (errpreset)* value.

The top of the QPSS form looks as follows.

**Quasi-Periodic Steady State Analysis**

Fundamental Tones						
#	Name	Expr	Value	Signal	SrcId	Harms
1	<code>bb_freq</code>	<code>1M</code>	<code>1M</code>	Moderate	I344	9
3	<code>carrier</code>	<code>1/(1n-0)</code>	<code>1G</code>	Large	V1	1
2	<code>carrier</code>	<code>1/(1n-0)</code>	<code>1G</code>	Large	V0	1

Moderate ▾

View Harmonics

**Accuracy Defaults (errpreset)**

conservative
  moderate
  liberal

Additional Time for Stabilization (tstab)

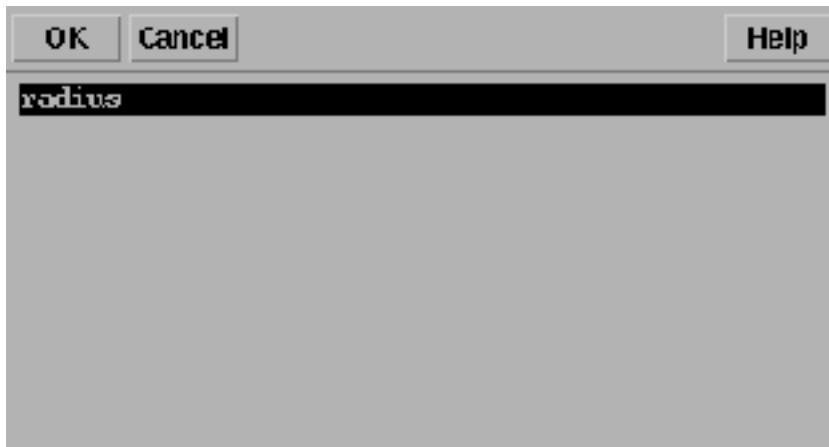
Save Initial Transient Results (saveinit)  no  yes

6. Highlight *sweep*.

The form expands to let you specify sweep data. The defaults are *Variable* in the *Sweep* cyclic field and *no* for *Frequency Variable*.

7. Click *Select Design Variable*.

The Select Design Variable form appears.



8. In the Select Design Variable form, highlight *radius* and click *OK*.
9. In the QPSS form, highlight *Start – Stop* for *Sweep Range*.
10. Type *.001* in the *Start* field and *1.5* in the *Stop* field.
11. Highlight *Linear* for *Sweep Type*, and highlight *Number of Steps*. Type *20* in the *Number of Steps* field.
12. Choose *Options*.  
The QPSS Options form appears.
13. In the *Accuracy Parameters* section of the QPSS Options form, type *1* in the *steadyratio* field.

**Important**

This circuit can cause convergence problems because the behavioral amplifiers have hard saturation curves. The *steadyratio* of 1 should work on all platforms. A *steadyratio* value greater than 1 can cause inaccuracy.

If you experience convergence problems at any point in the QPSS sweep, tighten the maximum timestep, *maxstep*, or increase *maxacfreq* in the *Time Step Parameters* section of the QPSS Options form.

14. In the QPSS Options form, click *OK*.

15. Be sure *Enabled* is highlighted.

The bottom of the QPSS form looks as follows.

The screenshot shows the QPSS form with the following settings:

- Sweep** (checkbox checked)
- Frequency Variable?  no  yes
- Variable Name:
- Select Design Variable (button)
- Sweep Range**
  - Start-Stop
  - Start:
  - Stop:
  - Center-Span
- Sweep Type**
  - Linear
  - Step Size
  - Logarithmic
  - Number of Steps
  - Number of Steps:
- Add Specific Points
- Enabled
- Options... (button)

16. In the QPSS Choosing Analyses form, click *OK*.

## Running the Simulation

1. In the Simulation window, choose *Simulation – Netlist and Run* to run the simulation.

The output log file appears and displays information about the simulation as it runs.

2. Check the\_CIW for a message saying the simulation completed successfully.

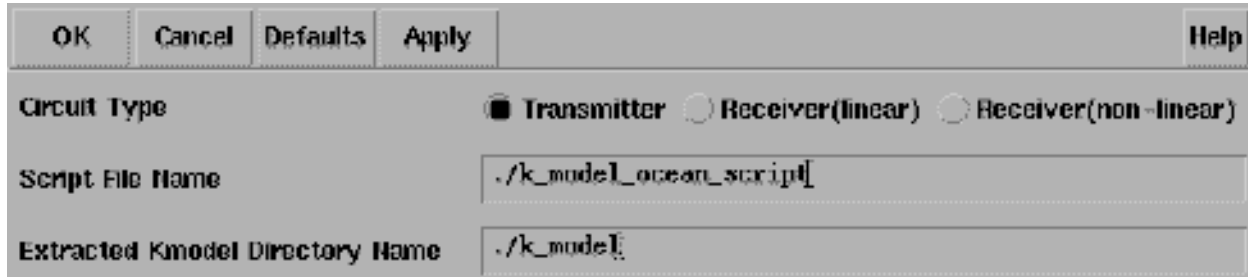
## Creating the J-Model

1. In the Simulation window, choose *Tools – RF – Link to SPW*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

The RFIC Modeler for SPW form appears.



OK Cancel Defaults Apply Help

Circuit Type  Transmitter  Receiver(linear)  Receiver(non-linear)

Script File Name ./k\_model\_ocean\_script

Extracted Kmodel Directory Name ./k\_model

2. Highlight *Transmitter* for the *Circuit Type*.
3. Edit the *Script File Name* and the *Extracted Kmodel Directory Name*, if necessary.

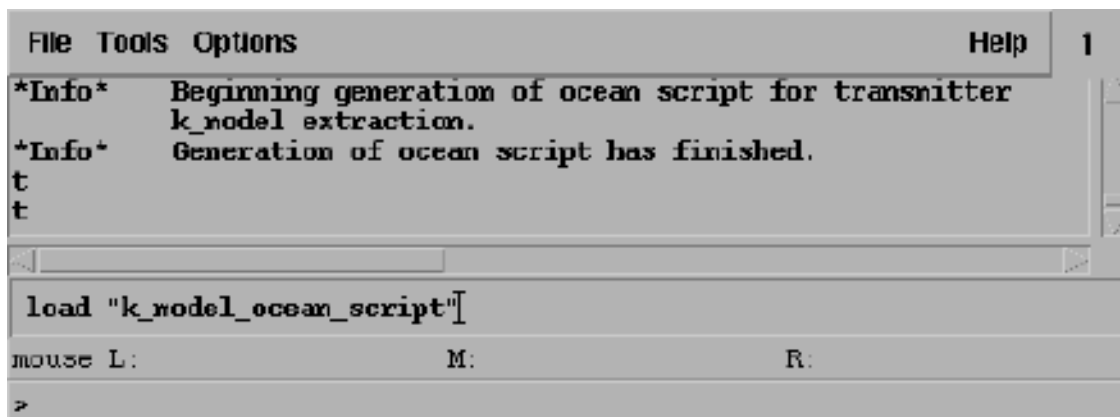
The file name is the name of the Ocean script to run later, and the directory name is the output directory.

4. Click *OK*.

In the CIW, watch for a message that the Ocean script is generated.

5. In the CIW, load the Ocean script as follows. Be sure to put the script name in double quotes.

```
load "k_model_ocean_script"
```



6. Click *Return*.

You can follow the progress of the script in the CIW. When it finishes, the j-model is created in the directory you specified.



## Using the J-model in a Circuit

In this section you create a circuit that contains the j-model you created and run an appropriate simulation.

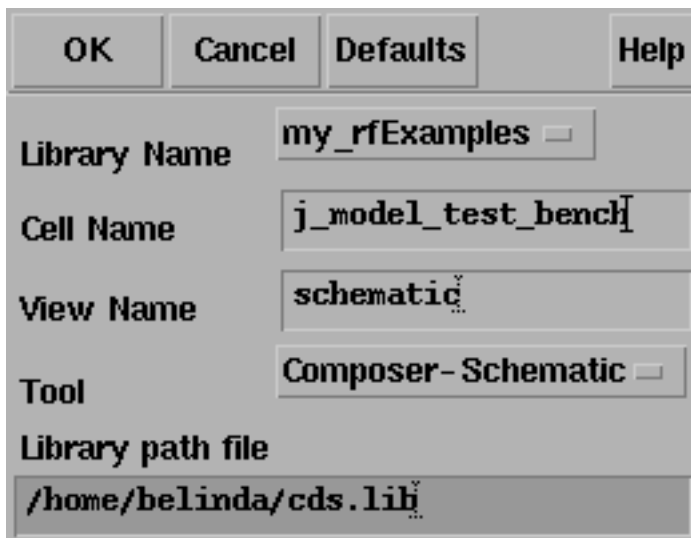
### Setting up the Circuit for Simulation

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

The Create New File form appears.

2. Choose a library name from the *Library Name* cyclic field.
3. Type a name for the *Cell Name*.
4. Type a name for the *View Name*.

The completed form looks like this.



The screenshot shows a dialog box with the following fields and values:

Field	Value
Library Name	my_rfExamples
Cell Name	j_model_test_bench
View Name	schematic
Tool	Composer- Schematic
Library path file	/home/belinda/cds.lib

5. Click *OK*.

An empty Schematic window appears.

6. In the new Schematic window, choose *Add – Instance*.

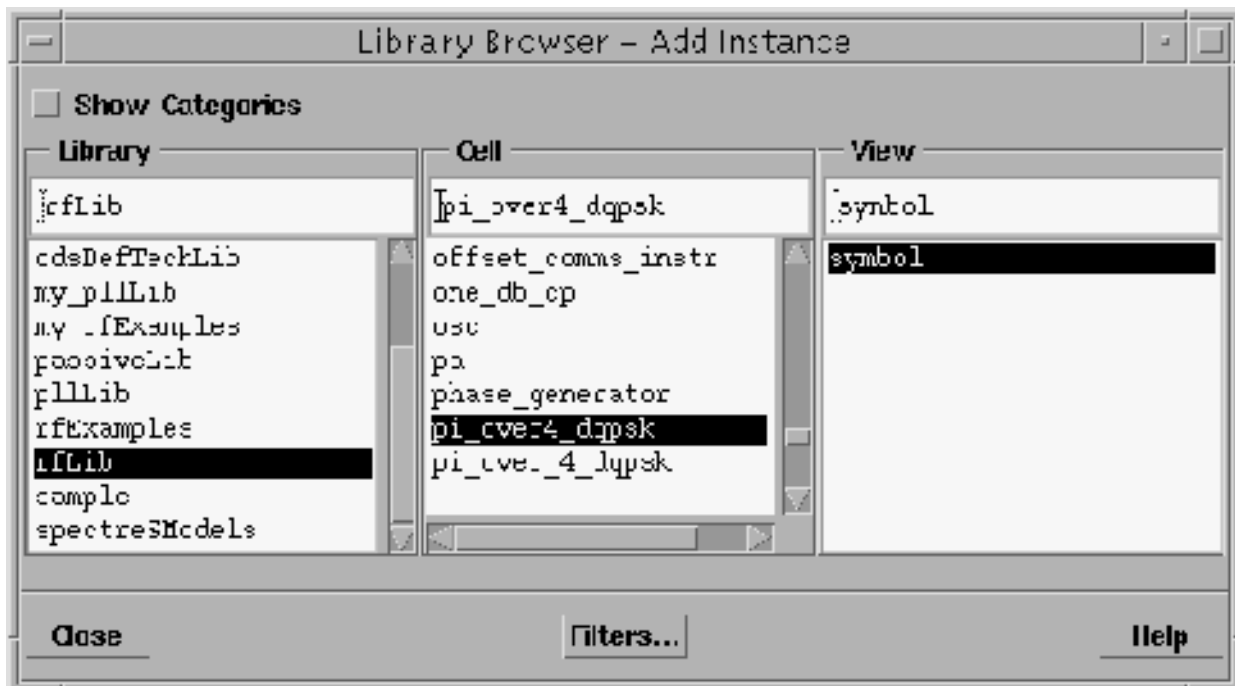
The Add Instance form appears.

7. In the Add Instance form, click *Browse*.

The Library Browser – Add Instance form appears.

8. In the Library Browser – Add Instance form, do the following:
  - a. Choose `rfLib` for *Library*.
  - b. Choose `pi_over4_dqpsk` for *Cell*.
  - c. Choose `symbol` for *View*.

The completed form looks like this



9. Place the cursor at the left end of the Schematic window and left click to place an instance of the `pi_over4_dqpsk` in the schematic. Press **ESC** after placing the instance.

The appropriate instance is attached to the cursor when you place the cursor in the schematic. If you need assistance for working in the Schematic window, see *Virtuoso® Schematic Composer™ User Guide*.

10. Repeat the procedure you used to place the `pi_over4_dqpsk` instance in the schematic to add a `7th_order_j_model` instance from the `rfLib` to the right of the `pi_over4_dqpsk` along with three resistors and three grounds attached to the

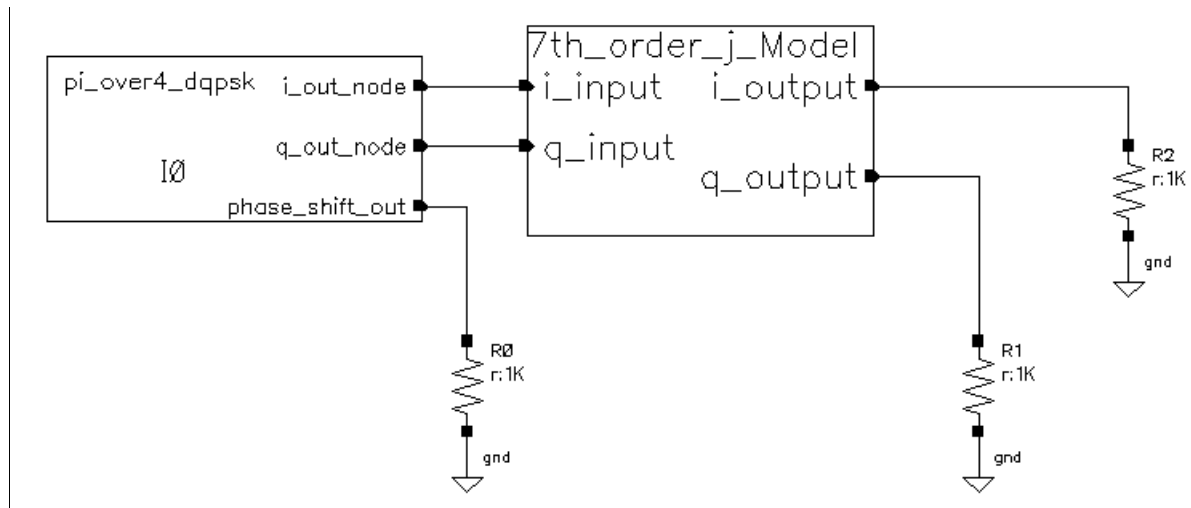
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

resistors. You create multiple copies of an instance with repeated left mouse clicks. The table below tells you the *Library*, *Cell*, and *View* values to choose.

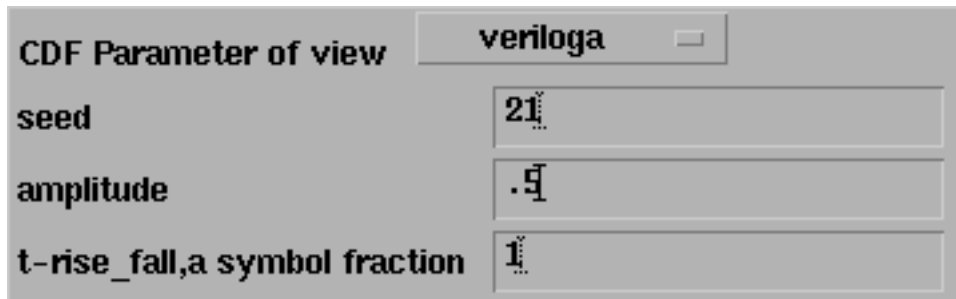
Library	Cell	View
rfLib	7th_order_j_model	symbol
analogLib	res	symbol
analogLib	gnd	symbol

11. In the Schematic window, choose *Add – Wire (narrow)*. Click beginning and ending locations in the schematic to add wires so the final schematic looks like the one below.



12. Press *ESC* after you finish adding the wires.
13. In the Schematic window, highlight the `pi_over4_dqpsk` instance and choose *Edit – Properties – Objects*.
14. Specify the following in the Edit Object Properties form:
- Choose `veriloga` for *CDF Parameter of view*.
  - Type `.5` for *amplitude*.

The completed CDF parameter edits look like this




CDF Parameter of view	veriloga
seed	21
amplitude	.5
t-rise_fall,a symbol fraction	1

c. Click *OK*.

15. In the Schematic window, highlight the `7th_order_j_model` instance, and repeat the previous steps to bring up the Edit Object Properties form. Edit the form as follows:
  - a. Choose *ahdl* for *DCF Parameter of view*.
  - b. Type the full path to the *j\_model* for the *data\_directory* value.

The completed CDF parameter section of the form looks like this.



CDF Parameter of view	ahdl	Display
data_directory	"/home/belinda/j_model"	off

c. Click *OK*.

16. In the Schematic window, choose *Design – Check and Save*.
17. In the Schematic window, choose *Tools – Analog Environment*.

The Simulation window appears.

18. In the Simulation window, choose *Analyses – Choose*.

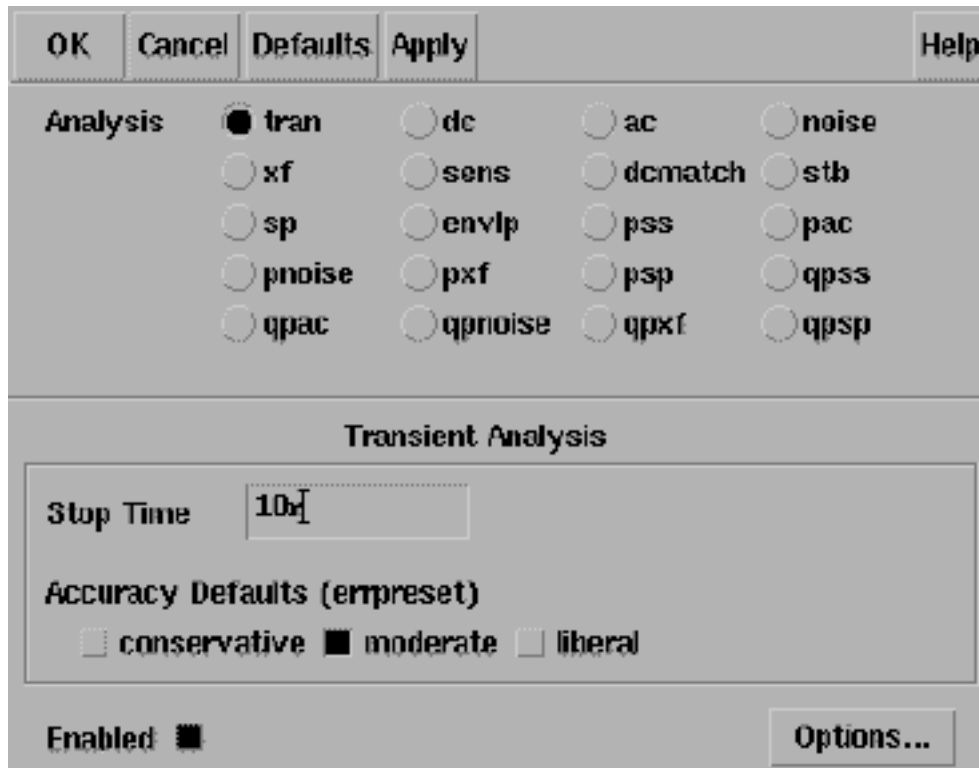
The Choosing Analyses form appears.

19. Specify the following in the Choosing Analyses form:

- a. Type `10m` for the *Stop* time.
- b. Highlight *moderate* for *Accuracy Defaults (errpreset)*.

You can also leave this value unselected because *moderate* is the default.

- c. Be sure the *Enabled* button is highlighted.



- d. Click *OK*.

20. In the Simulation window, choose *Simulation – Netlist and Run* to run the simulation.

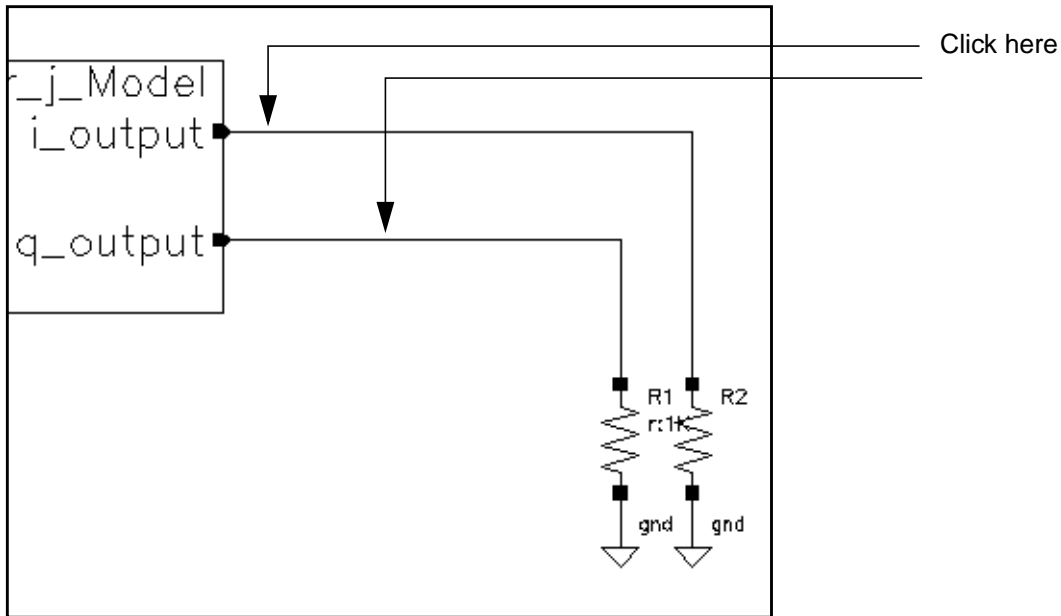
Check the CIW for a message saying the simulation completed successfully.

### Working with the Simulation Results

1. In the Simulation window, choose *Results – Direct Plot – Transient Signal*.

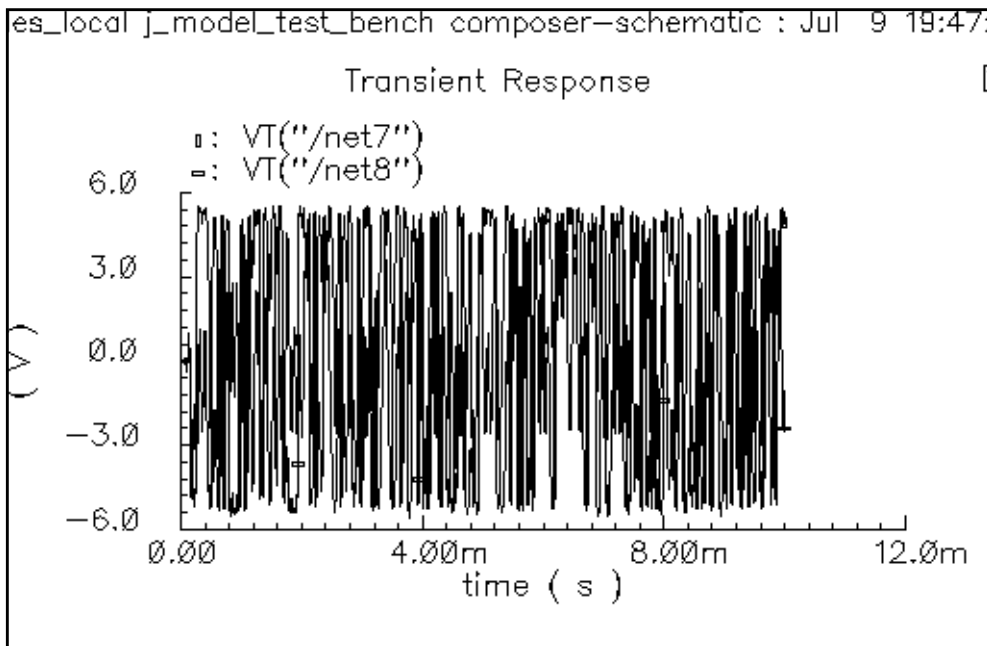
An empty Waveform window appears.

2. In the Schematic window, click the *i\_output* net and then on the *q\_output* net



3. Press the *ESC* key.

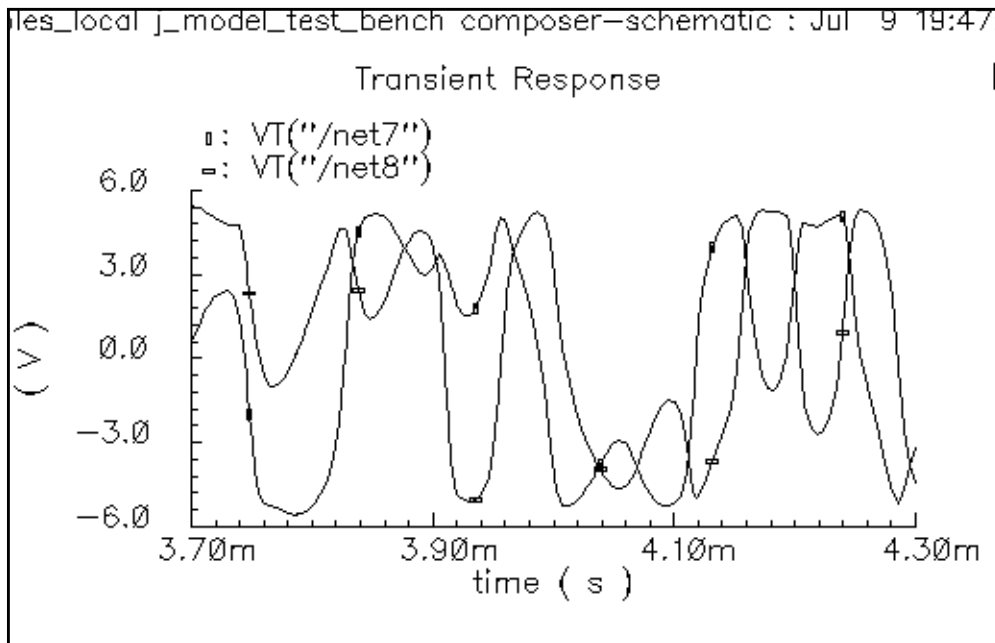
The plots for *i\_output* and *q\_output* appear in the Waveform window.



4. Highlight a small portion of the waveform by performing the following steps:
- a. In the Waveform window, choose *Zoom – Zoom In*.

- b. Left click and drag with the mouse to form a rectangle that includes the waveform over a small time value.
- c. Left click to complete the selection.

The waveform display changes to show the portion you select with the mouse.



- 5. In the Simulation window, choose *Tools – Calculator*.

The Waveform Calculator appears.

- 6. In the Waveform Calculator, click the *wave* key.
- 7. In the Waveform window, click each of the two waves that are displayed.
- 8. In the Waveform Calculator, choose *psdbb...* from the *Special Functions* cyclic field.

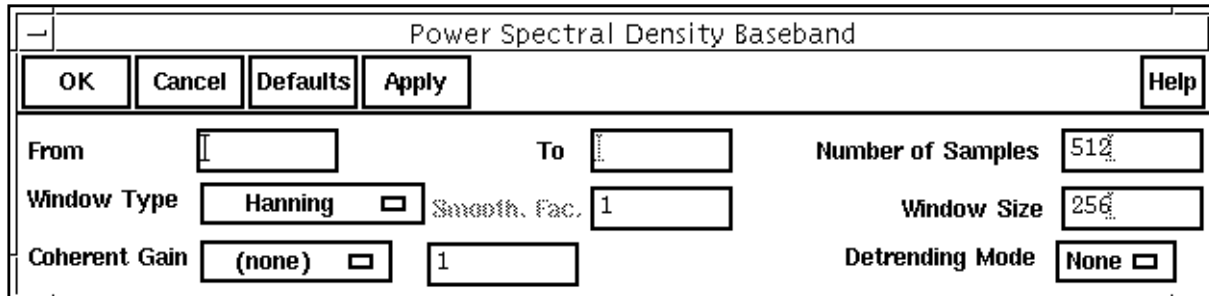
The *psdbb* function is documented in the Waveform Calculator manual.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

---

The Power Spectral Density Baseband form appears.



The screenshot shows a dialog box titled "Power Spectral Density Baseband". It contains several controls: "OK", "Cancel", "Defaults", "Apply", and "Help" buttons at the top. Below the buttons are input fields for "From", "To", "Number of Samples", "Window Type", "Smooth. Fac.", "Window Size", "Coherent Gain", and "Detrending Mode". The "From" field is empty, "To" is empty, "Number of Samples" is 512, "Window Type" is Hanning, "Smooth. Fac." is 1, "Window Size" is 256, "Coherent Gain" is (none), and "Detrending Mode" is None.

Field	Value
From	
To	
Number of Samples	512
Window Type	Hanning
Smooth. Fac.	1
Window Size	256
Coherent Gain	(none)
Detrending Mode	None

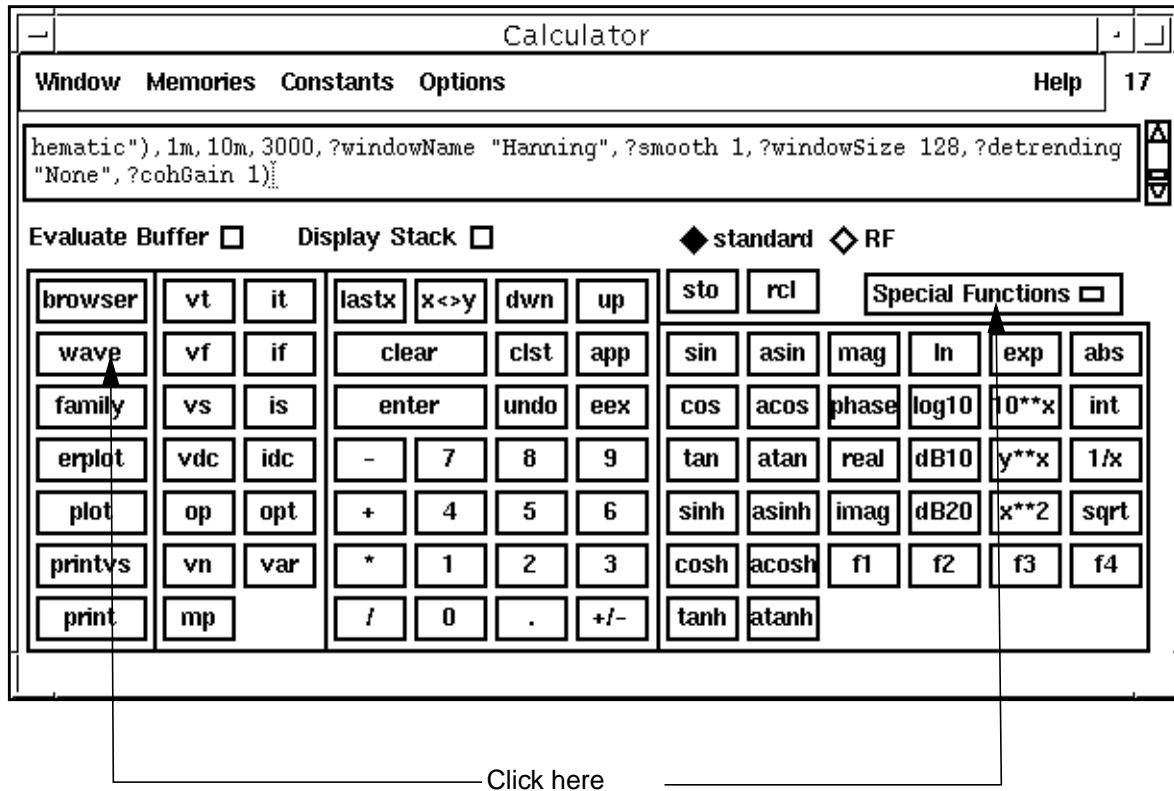
9. In the Power Spectral Density Baseband form, do the following:
  - a. Type 1m in the *From* field.
  - b. Type 10m in the *To* field.
  - c. Type 3000 for the *Number of Samples*.
  - d. Type 128 for the *Window Size*.
  - e. Click *OK*.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

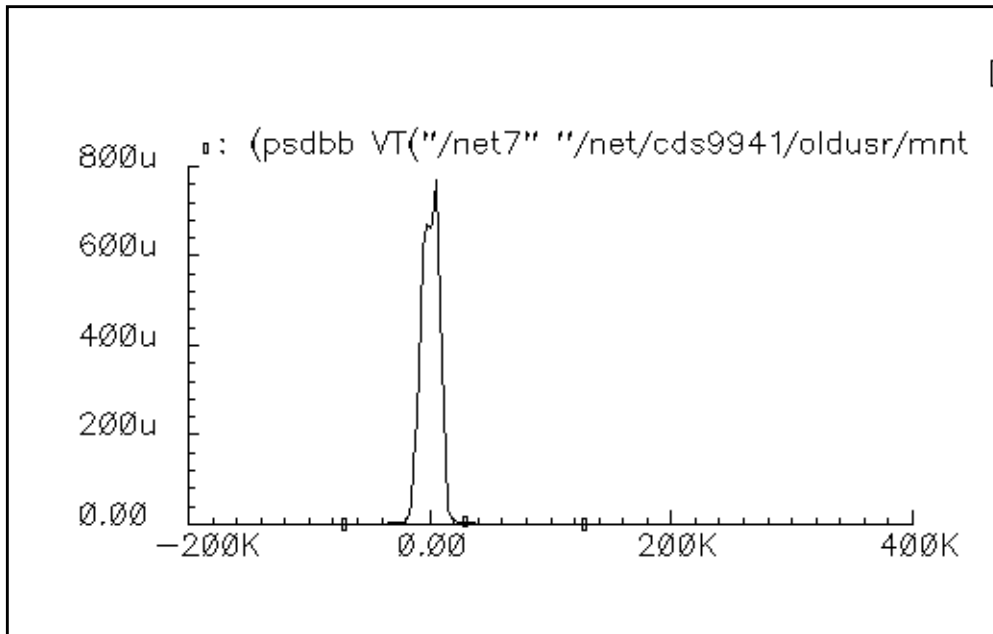
### Creating and Using Transmitter J-Models

The Waveform Calculator now looks like this.



10. In the Waveform window, choose *Window – Reset*.
11. In the Waveform Calculator, click *Plot*.

The plot for *psdbb* appears in the Waveform window.

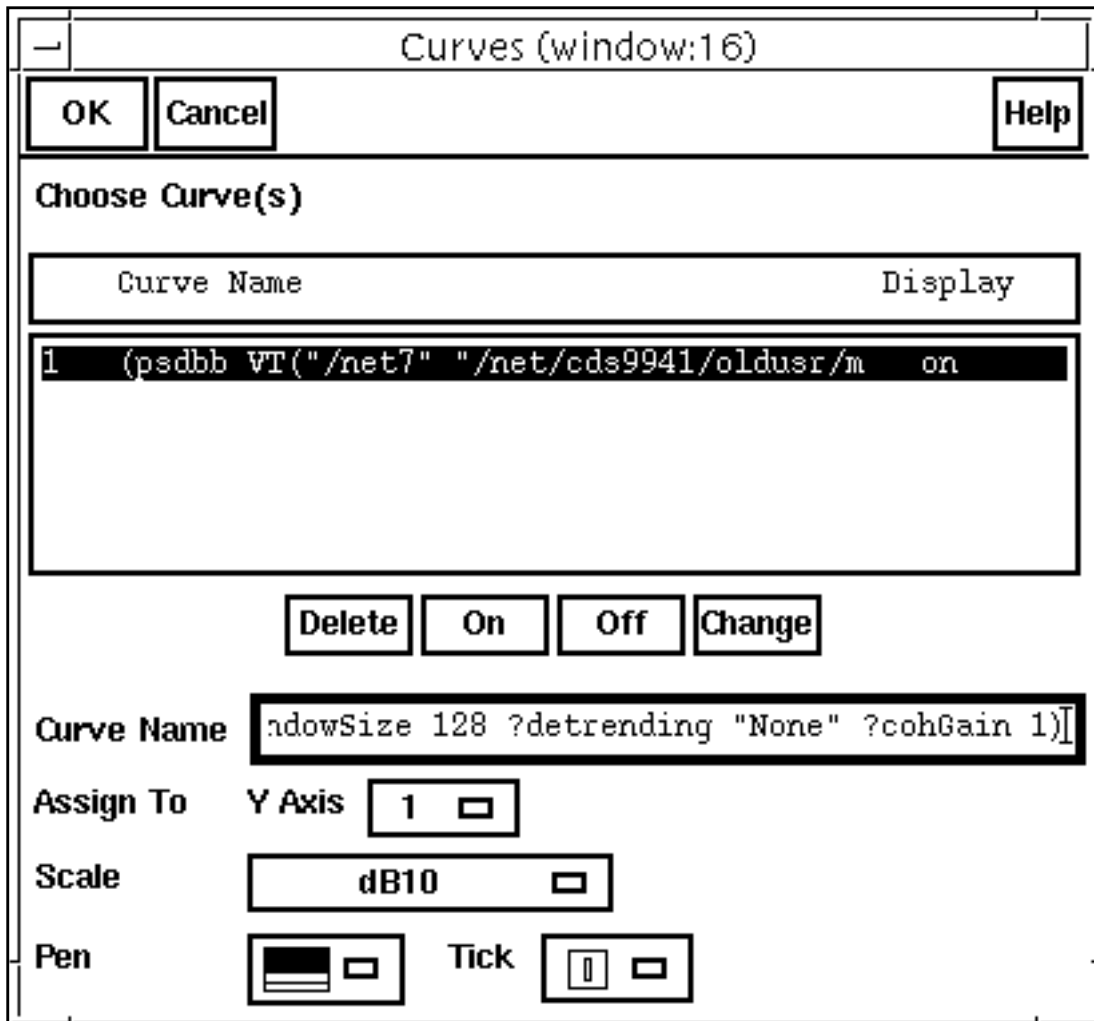


12. In the Waveform window, click *Curves – Edit*.

The Curves window appears.

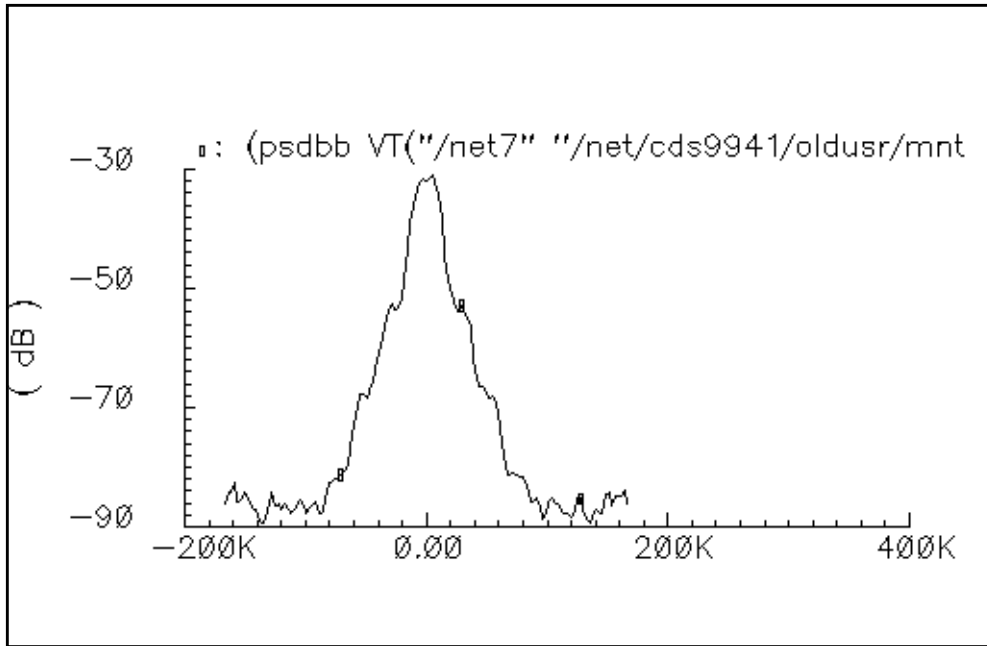
13. In the Curves window, do the following:
  - a. Click the value in the *Choose Curve(s)* list box.
  - b. Choose db10 for *Scale*.

The Choose Curve(s) list box looks like this.



c. Click *OK*.

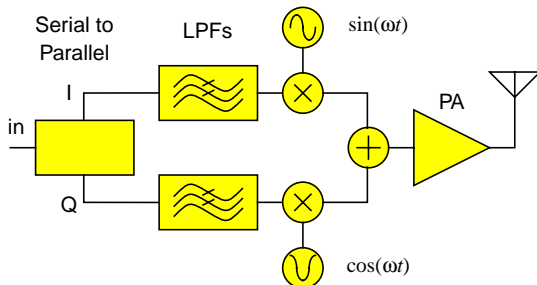
The plot changes to reflect your new specifications. ACPR is computed by taking the difference in db between the power spectral density measured at the center frequency and at the adjacent channel.



## More About the J-model

Figure 6-1 shows a typical direct-conversion digital transmitter architecture.

Figure 6-1 Direct Conversion Transmitter



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

---

The low-pass filters can eliminate out-of-band signal components from the input baseband signal. However, these components might reappear because of intermodulation distortion generated within the transmitter, such as in the downstream power amplifier. This return of unwanted signal components is called “spectral regrowth.”

Adjacent Channel Power Ratio (ACPR) is a popular measure of spectral regrowth in digital transmitters and is often a design specification. To get the frequency resolution required for accurate estimation of ACPR, the simulation often must process a random sequence of between 500 and 10k input symbols.

Envelope analysis simulates the transmitter efficiently, but it must be driven by deterministic or unfiltered random signals, and a simulation of 10k signals is lengthy.

Most filters, like those in [Figure 1](#), are FIR filters. Spectre RF does not work with FIR filters (implemented in AHDL) because they contain hidden state. To drive an Envelope analysis with realistic random baseband signals, you must create the signals in a separate simulation, store them to a file, then read the data into the Envelope analysis with piece-wise linear sources. However, the QPSS analysis can perform a fast, indirect ACPR estimation that can simulate 10k symbols within minutes after you extract the model. Extraction time is independent of the number of symbols you want to simulate.

Initially, QPSS extracts a behavioral baseband-equivalent model of the transmitter. In transmitter circuits, the input baseband signal is usually well within the transmitter’s bandwidth, so a model without memory, such as the J-model, is often sufficient. Because the behavioral model eliminates the carrier and unnecessary circuit details, the subsequent ACPR calculation step is fast, even for large or complex circuits.

The most familiar spectral regrowth mechanism is the intermodulation distortion of amplitude modulation (AM) signal components that enter the power amplifier. Modulation schemes that carry information only on the carrier phase or frequency try to minimize spectral regrowth by eliminating this AM component. However, the digital baseband filters can still convert discontinuous phase changes into amplitude transients in the composite RF signal. Models based on AM/AM and AM/PM conversion [2] capture most such power amplifier related distortion mechanisms. However, imperfections in the I/Q modulators can convert input phase variations into output amplitude and phase variations that also contribute to the distortion. This is particularly true for GSM transmitters. For this reason, we also include PM/AM and PM/PM conversion effects in our model. Formally, the J-model is a narrowband approximation to a vector-valued, periodically time-varying, functional series expansion [3].

To understand the extraction procedure, write the inputs in polar coordinates, such as magnitude  $\rho$  and phase  $\phi$ . Consider the mapping  $f(\rho, \phi)$  from baseband input to the baseband representation of the RF output. That is, if

$$input = \rho \cos(\phi) + j\rho \sin(\phi)$$

$$output = Re[f(\rho, \phi)]\cos(\omega_c t) - Im[f(\rho, \phi)]\sin(\omega_c t) = Re[f(\rho, \phi)\exp(j\omega_c t)] \quad .$$

The baseband representation of the output is  $f(\rho, \phi)$ . The RF carrier frequency is  $\omega_c$ . Because we assume the system is memoryless with respect to baseband signals, and that we have only one frequency translation, we can assume  $f(\rho, \phi)$  is periodic in  $\phi$  and can be expressed as a Fourier series,

$$f(\rho, \phi) = \sum_k A_k(\rho) e^{jk\phi} \quad ,$$

where the magnitude-dependent Fourier coefficients are

$$A_k(\rho) = \frac{1}{2\pi} \int_0^{2\pi} f(\rho, \phi) e^{-jk\phi} d\phi \quad .$$

To extract the Fourier coefficients, the *I* and *Q* inputs of the transmitter are driven with sinusoids in quadrature, at a frequency within the transmitter's bandwidth. For these circular input trajectories,  $\phi = \omega_0 t$ , and the Fourier coefficients are given by

$$A_k(\rho) = \frac{\omega_0}{2\pi} \int_0^{2\pi/\omega_0} f(\rho, \omega_0 t) e^{-jk\omega_0 t} dt \quad .$$

Therefore, for a given input magnitude, the Fourier coefficients are obtained directly from the output spectrum calculated by QPSS. For example, if the input circle is large enough to alternately saturate the modulators, the  $-3$  harmonic of the complex baseband tone dominates the distortion. The output spectrum has lines at  $\omega_c + \omega_0$  and  $\omega_c - 3\omega_0$ . QPSS computes the real and imaginary parts of the Fourier coefficients associated with those lines. The simulation is repeated for a range of input magnitudes to capture the magnitude dependence of the Fourier coefficients. At each input magnitude, the fundamental and its relevant harmonics are recorded for interpolation. The model implementation reads the Fourier coefficients and then processes any amount of input baseband data according to  $f(\rho, \phi)$ .

[Figure 6-2](#) on page 359 shows the model extraction process for the fundamental and  $(-3)$  terms in the expansion  $f(\rho, \phi)$ . The extraction script records all harmonics up to 9th order, including even-numbered harmonics. [Figure 6-3](#) on page 360 shows a block diagram of the associated model for just the fundamental and  $-3$  harmonic.

Figure 6-2 Model Extraction

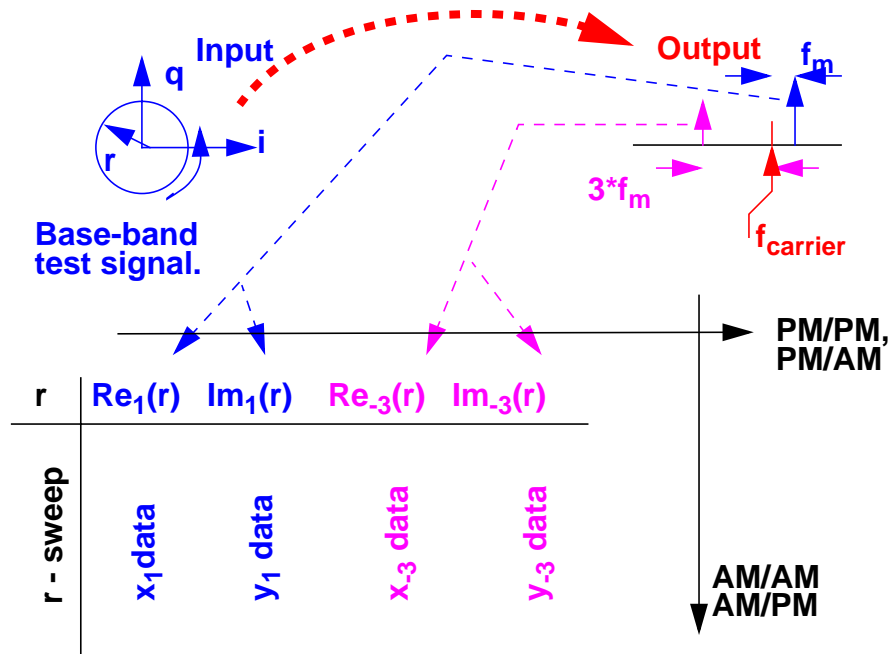
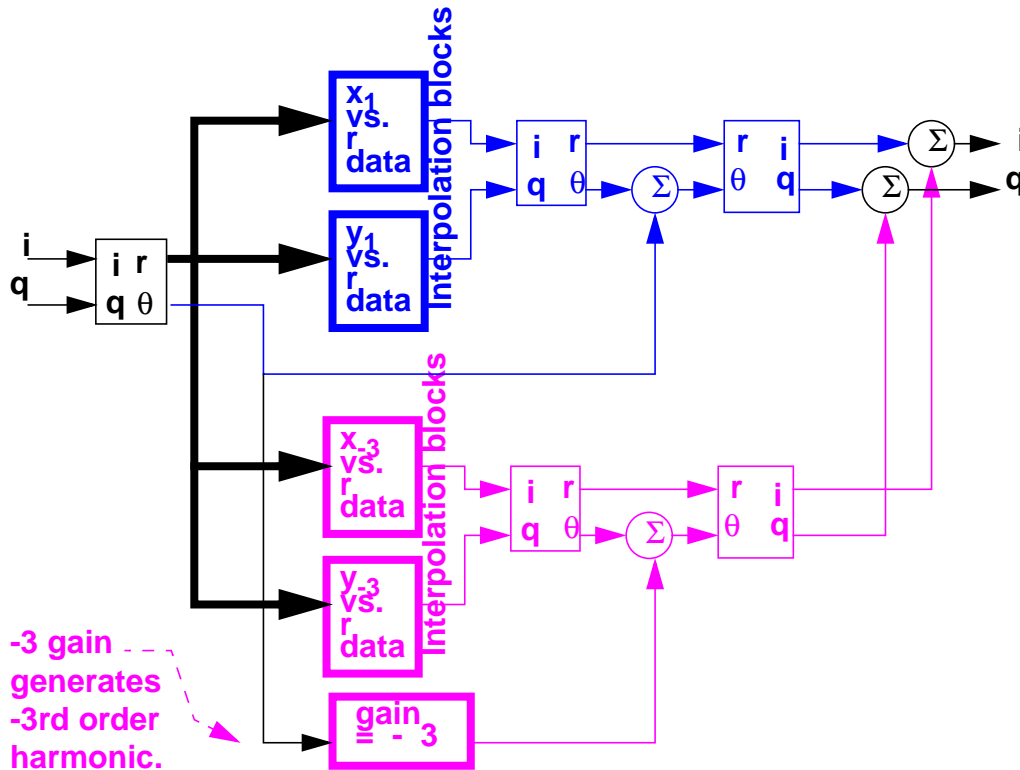


Figure 6-3 Block Diagram

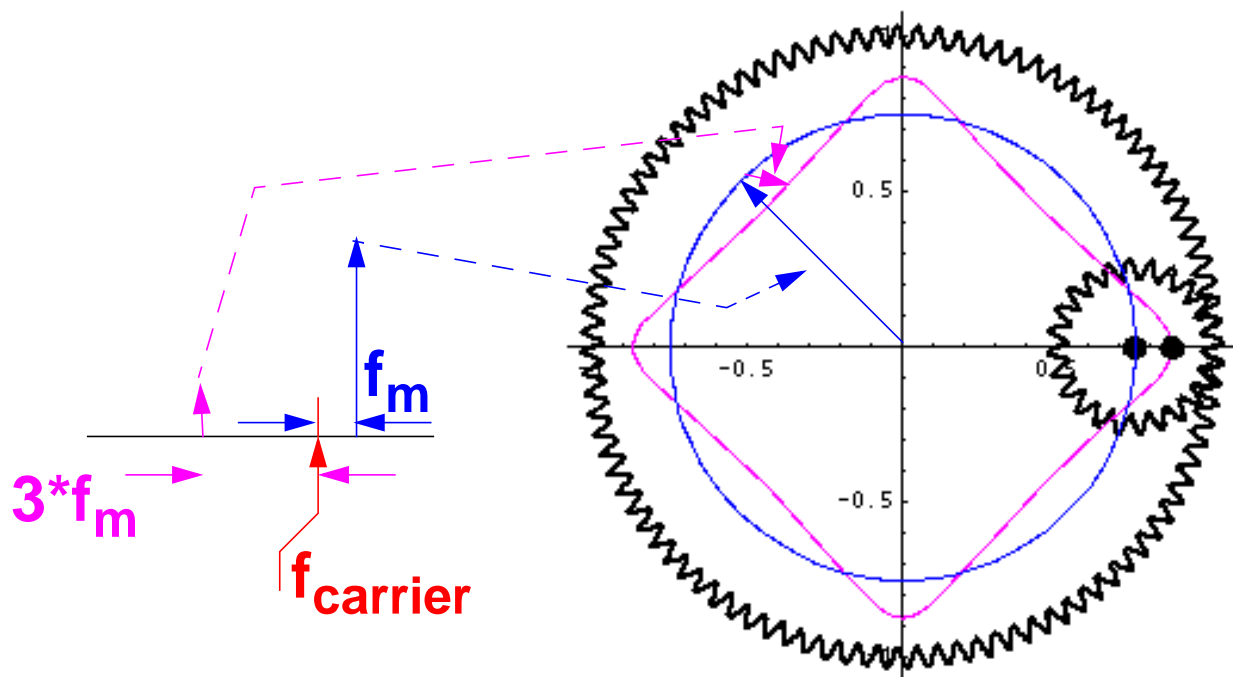


You can visualize PM/AM conversion, PM/PM conversion, and the new macromodel with the help of the Spirograph™ plot shown in Figure 6-4 on page 361. Consider just one distortion harmonic, the minus three harmonic, in the baseband output that is generated when the input trajectory is circular. There are no baseband harmonics generated by AM/AM or AM/PM conversion because the input radius is constant. The fundamental term. The ideal term, in the power amplifier output is represented by a vector from the center of the large inverted gear to the center of the small non-inverted gear. Since the input trajectory is circular, the fundamental vector rotates about the origin at the same frequency the input trajectory rotates about its origin. That is exactly the trajectory traced out by a pen placed in a hole at the center of the small gear as the small gear rolls around the inside of the inverted gear. If we place the pen in a hole **offset** from the center of the small gear, the vector from the center of the small gear to the hole represents the  $-3\omega_0$  distortion term. The small gear rotates -3 times for each complete trip around the inverted gear. This is precisely what the model does for each recorded harmonic. The model determines the radii of both gears, as well as the phase offsets, from the recorded QPSS results. (Although not intuitive, the -3 frequency ratio really does produce a four-cornered trace, demonstrating that the square pattern caused by



independently saturating I/Q modulators results from a minus third harmonic.) The gear analogy clearly illustrates PM/AM and PM/PM conversion. The offset hole distorts the otherwise circular trajectory with phase and amplitude perturbations that vary with the input phase.

Figure 6-4 Spirograph™ P lot



To measure run times we contrived a benchmark transmitter that included the up-conversion mixers and power amplifier. The benchmark circuit contained 46 transistors and generated 328 equations in the circuit simulator. A 7<sup>th</sup> order model took 4 hours to extract and simulated 30ms of CDMA data (40k chips @ 1.25Mchips/sec) in less than 21 minutes using a Sun Ultra 1.

Driving 1st order and 7th order models with CDMA and GSM baseband signals shows when PM/AM effects become important. The first order model only accounts for AM/AM and AM/PM conversion. The 7th order model includes PM/PM and PM/AM conversion as well AM/AM and AM/PM conversion. [Figure 6-5](#) on page 362 shows output CDMA trajectories. Although the trajectory from the 7th order model is more square, the power spectral density (PSD) estimates are nearly identical, as [Figure 6-6](#) on page 363 shows.

Figure 6-7 on page 363 and Figure 6-8 on page 364 show the output trajectories and power spectral densities when the models are driven by GSM signals of the same maximum amplitude. However, unlike the CDMA spectral results, the GSM spectral results differ significantly. Because the GSM input signal has no AM component, only the higher order models capture the true ACPR beyond about 150 KHz.

**Figure 6-5 Output CDMA Trajectories**

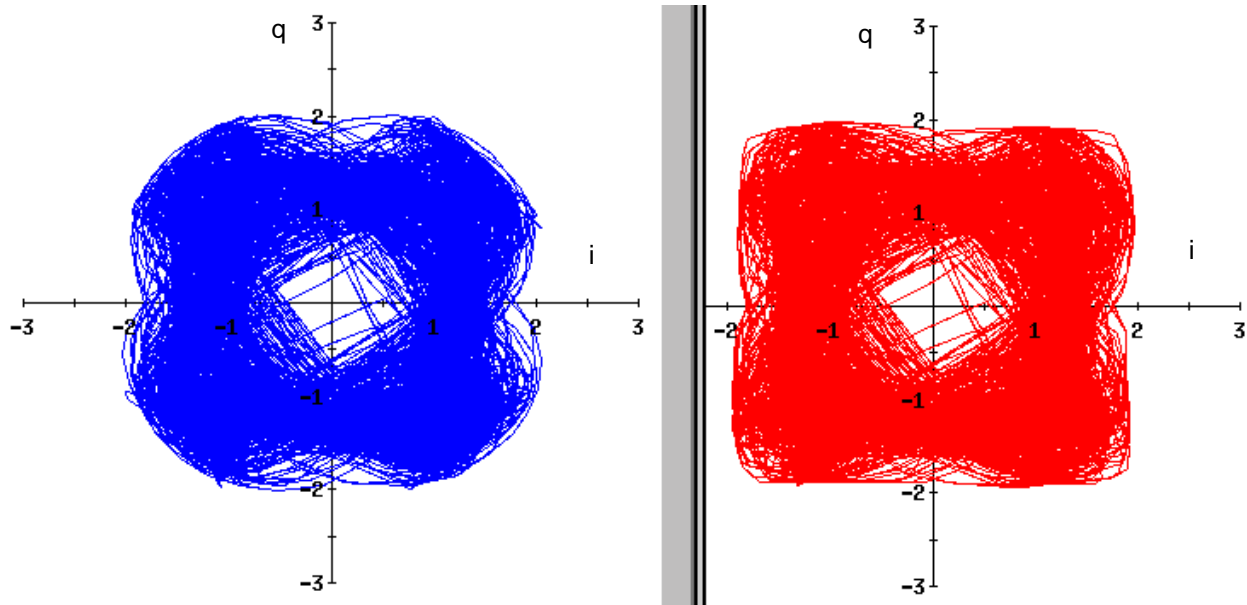


Figure 5a, 1st order output

Figure 5b, second order output

Figure 6-6

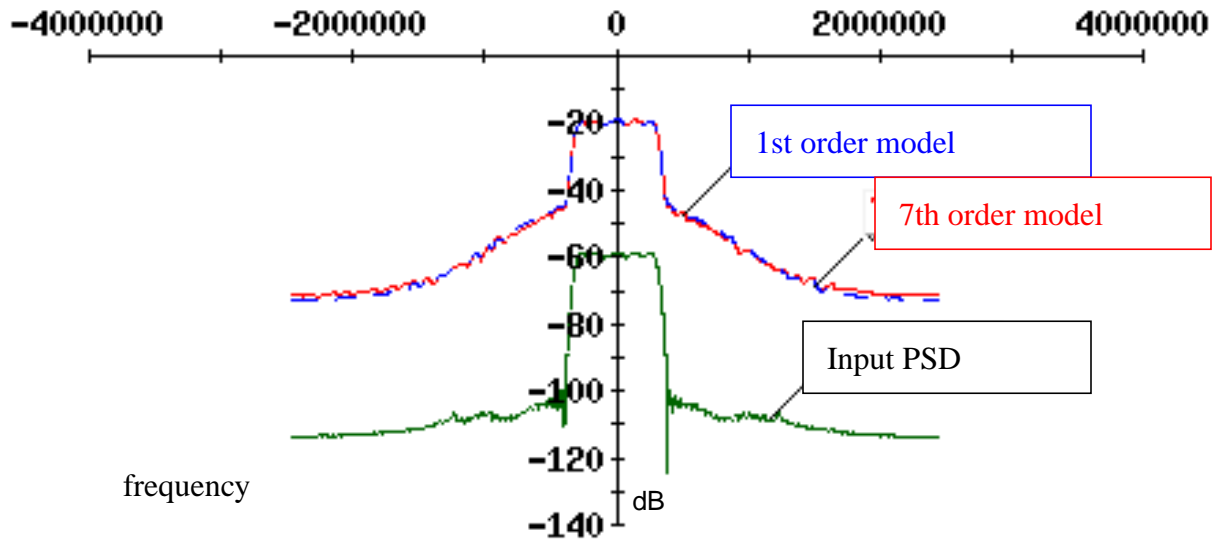


Figure 6-7 Trajectory for GSM

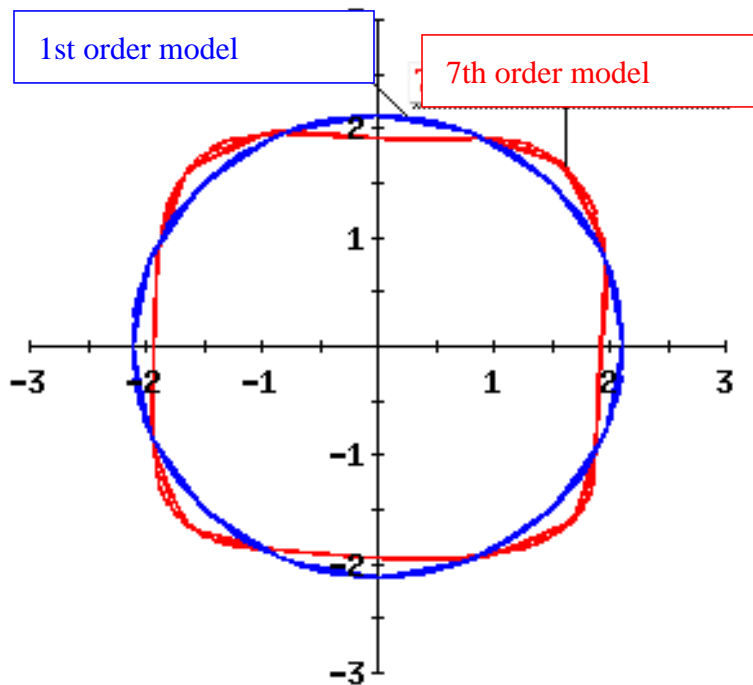
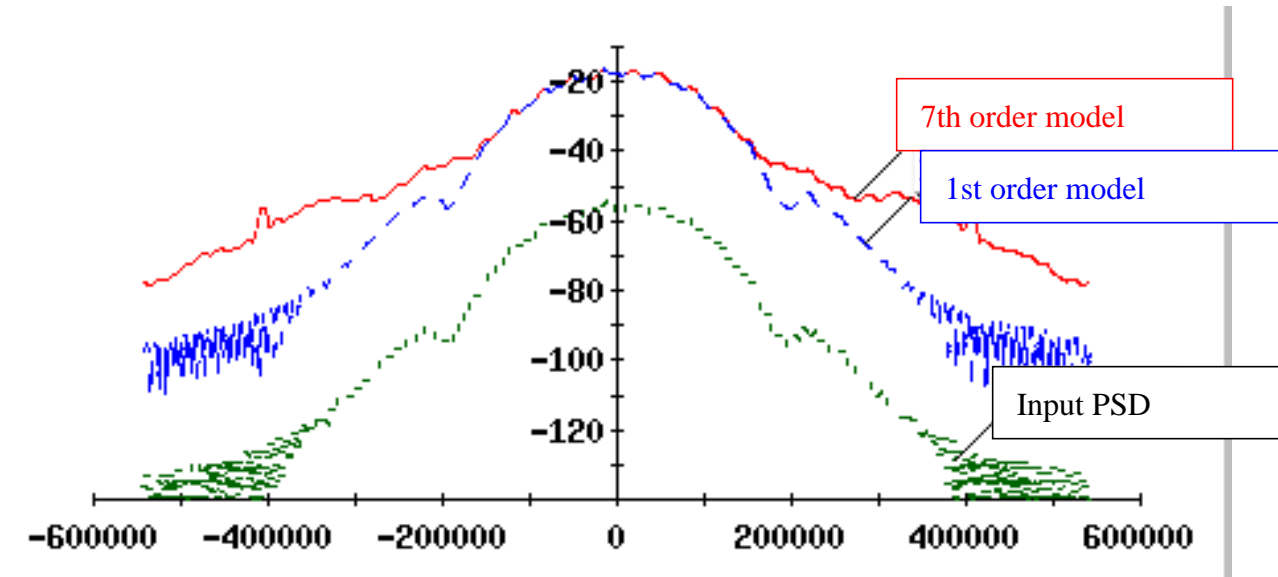


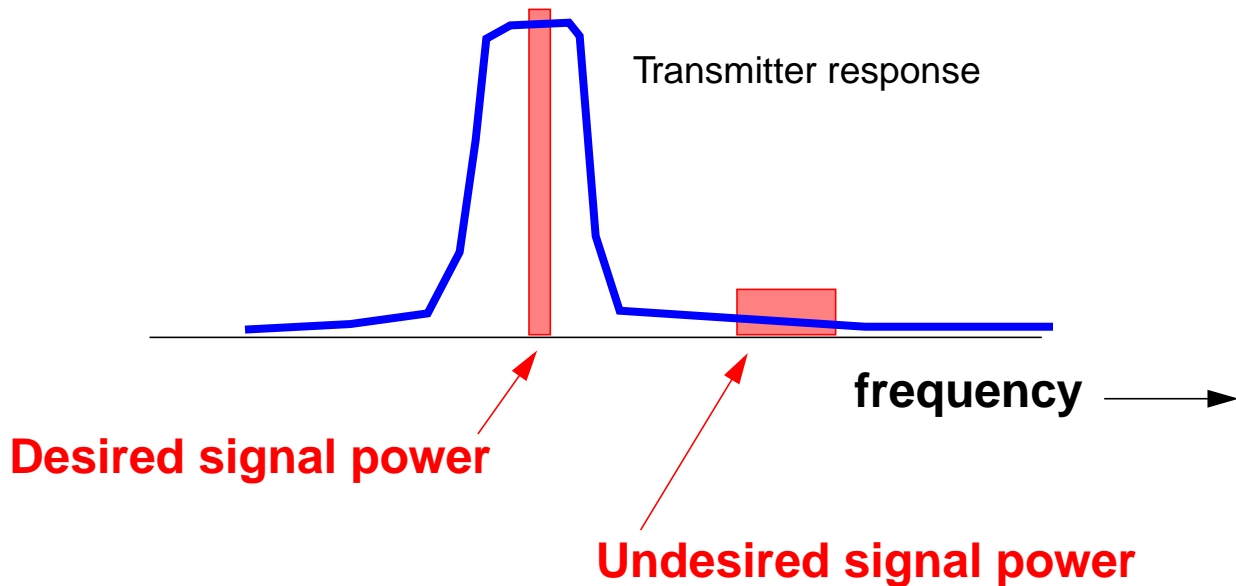
Figure 6-8 PSD for GSM



### J Model Limitations

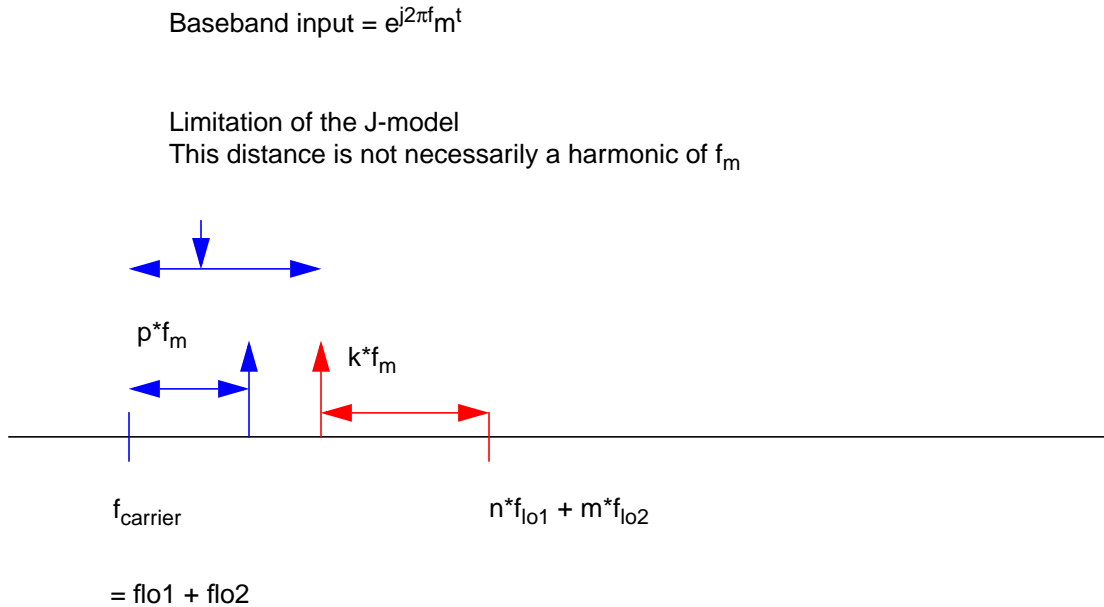
Like any behavioral model, the J-model has limitations. The J-model is a static model that has no memory. You can see this in the mechanical analogy. The pattern traced out by the pen in the small gear does not depend on how fast the pen and gear move. Figure 6-9 illustrates the limitation in the frequency domain. Let the thick curve be the transmitter's frequency response to baseband input. The rectangles show where signal power exists. The smaller rectangle of signal power in the picture below is generated within the transmitter. It is not part of the input signal. The input signal must be contained in the larger rectangle. As long as the transmitter's frequency response is flat over bands containing any significant signal power, the J-model is valid.

Figure 6-9 J Model Limitations in the Frequency Domain



Why is the J-model limited to direct conversion transmitters? Consider a double-conversion transmitter, one with an IF stage. With an IF stage, it might be possible for the output signal to contain power at some linear combination of the two local oscillator frequencies that falls near the output RF signal. Those frequencies are not displaced from the target RF carrier by some multiple of the input baseband frequencies. The figure below shows the problem. If you can ignore terms such as those in [Figure 6-10](#) on page 366, you can use the J-model for a transmitter with more than one frequency translation.

**Figure 6-10 J Model Limitations**



### J-model data files and data format

The J-model data files are listed below:

```

imag_minus1.ascsig
imag_minus2.ascsig
imag_minus3.ascsig
imag_minus4.ascsig
imag_minus5.ascsig
imag_minus6.ascsig
imag_minus7.ascsig
imag_minus8.ascsig
imag_minus9.ascsig
imag_plus1.ascsig
imag_plus2.ascsig
imag_plus3.ascsig
imag_plus4.ascsig
imag_plus5.ascsig
imag_plus6.ascsig
imag_plus7.ascsig
imag_plus8.ascsig
imag_plus9.ascsig
imag_zero.ascsig
real_minus1.ascsig
real_minus2.ascsig
real_minus3.ascsig
    
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

---

```
real_minus4.ascsig  
real_minus5.ascsig  
real_minus6.ascsig  
real_minus7.ascsig  
real_minus8.ascsig  
real_minus9.ascsig  
real_plus1.ascsig  
real_plus2.ascsig  
real_plus3.ascsig  
real_plus4.ascsig  
real_plus5.ascsig  
real_plus6.ascsig  
real_plus7.ascsig  
real_plus8.ascsig  
real_plus9.ascsig  
real_zero.ascsig
```

The first word in the file name, up to the underscore, tells you whether the data file contains the real or imaginary part of the Fourier coefficient. The second part, following the underscore, specifies the signed harmonic number. The suffix identifies an ascii file for Alta SPW.

The format of all file is the same and is shown below. The format is ascii. Each file contains exactly three header lines and no blank lines at the end. The data is arranged in two columns. The first column is the input signal level in volts. The second column is the real or imaginary part of the baseband Fourier coefficient as described above.

<b>sweep</b>	<b>imag(harmonic)</b>
1.000e-03	-8.475e-07
2.450e-03	-1.366e-05
3.900e-03	-4.850e-05
5.350e-03	1.595e-05
6.800e-03	3.690e-05
8.250e-03	-7.153e-04
9.700e-03	-1.055e-04
1.115e-02	-4.324e-05
1.260e-02	-1.431e-04
1.405e-02	-1.143e-03
1.550e-02	-4.352e-03
1.695e-02	-5.443e-04
1.840e-02	1.107e-03
1.985e-02	1.913e-03
2.130e-02	3.774e-02
2.275e-02	8.256e-02
2.420e-02	1.277e-01
2.565e-02	1.680e-01
2.710e-02	2.114e-01

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Creating and Using Transmitter J-Models

---

2.855e-02	2.474e-01
3.000e-02	2.841e-01

#### References

- [1] E. Ngoya and R. Larcheveque. Envelope transient analysis: a new method for the transient and steady state analysis of microwave communication circuits and systems. *IEEE MTT Symposium Digest*, pp. 1365-1368, June 1996.
- [2] M. Jeruchim, P. Balaban and K. Shanmugan, *Simulation of Communication Systems*, Plenum, 1992.
- [3] M. Schetzen, "*The Volterra & Wiener Theories of Non-Linear Systems*", Krieger Publishing Company, 1980.



---

## Modeling Transmitters

---

This chapter tells you how to use several Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) transmitter design features. It emphasizes

- Setting Up an Envelope analysis (Envlp)

The Envelope analysis example explains

- How to interpret Envlp analysis results
- How to visually detect distortion quickly

- Measuring ACPR and PSD. PSD (Transmitted Power Spectral Density) is characterized by the ACPR number (Adjacent Channel Power Ratio).

The ACPR and PSD example explains

- How to use the ACPR wizard.
- How to estimate PSD.

- Measuring Load Pull Contours and Reflection Coefficients

The Load-Pull example explains

- How to select an optimal power amplifier load
- How to determine whether the input matching network needs to be re-designed for the optimal load

- Using S-parameter Input Files

The S-parameter example explains

- How to generate tabulated S-parameters
- How to include tabulated S-parameters as input in a Spectre RF analysis

- Measuring AM and PM Conversion with Modulated PAC, AC and PXF Analyses

The AM/PM conversion example explains how to determine the amplitude and phase modulation effects in RF circuits using the Modulated PAC and PXF analyses.

- How to create the EX\_AMP example.
- How to set up the modulated PAC and PXF analyses

■ **Measuring Jitter with PSS and Pnoise Analyses**

This example explains how to measure different jitter measurements using the Pnoise Jitter measurement.

How to set up or create the EX\_AMP circuit.

- How to set up the Pnoise jitter analyses

## **Envelope Analysis**

This example tells you how to set up and run the Envelope analysis then explains why the time results look somewhat unusual.

Before you start, perform the setup procedures described in [Chapter 3](#).

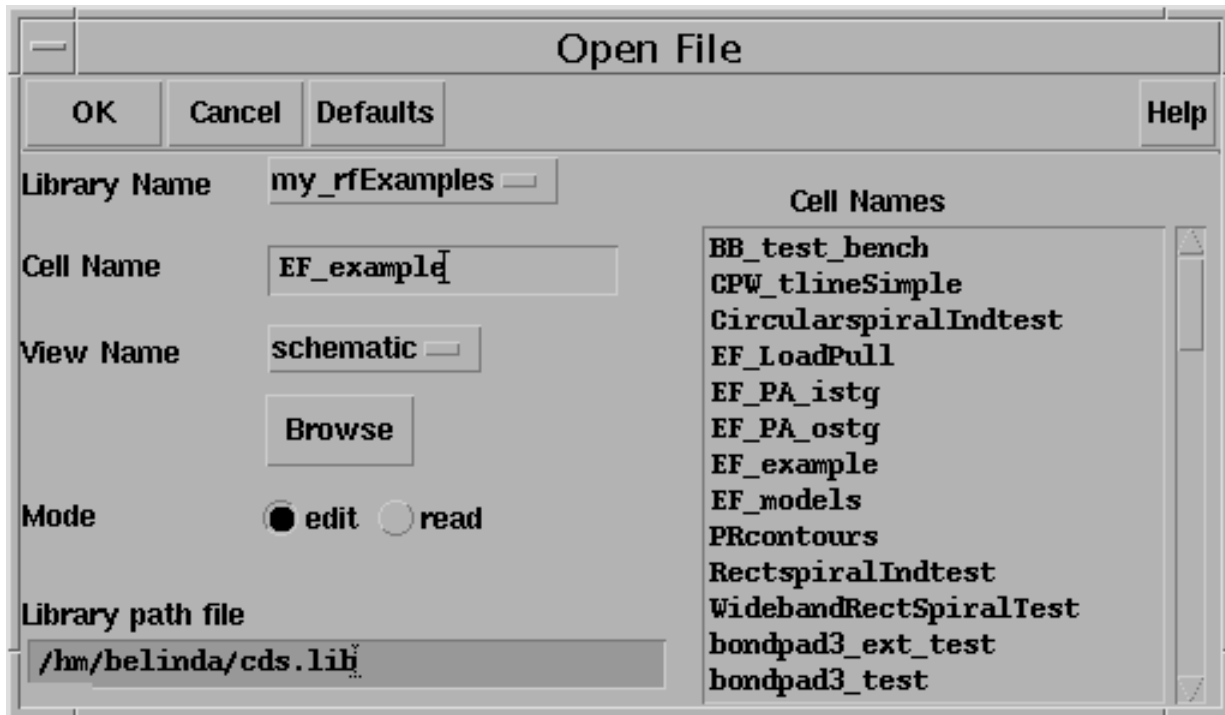
### **Opening the EF\_example Circuit in the Schematic Window**

1. In the CIW, choose *File – Open*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

The Open File form appears.



2. In the Open File form, do the following:

a. Choose *my\_rfExamples* for *Library Name*, the editable copy of *rfExamples*.

Select the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

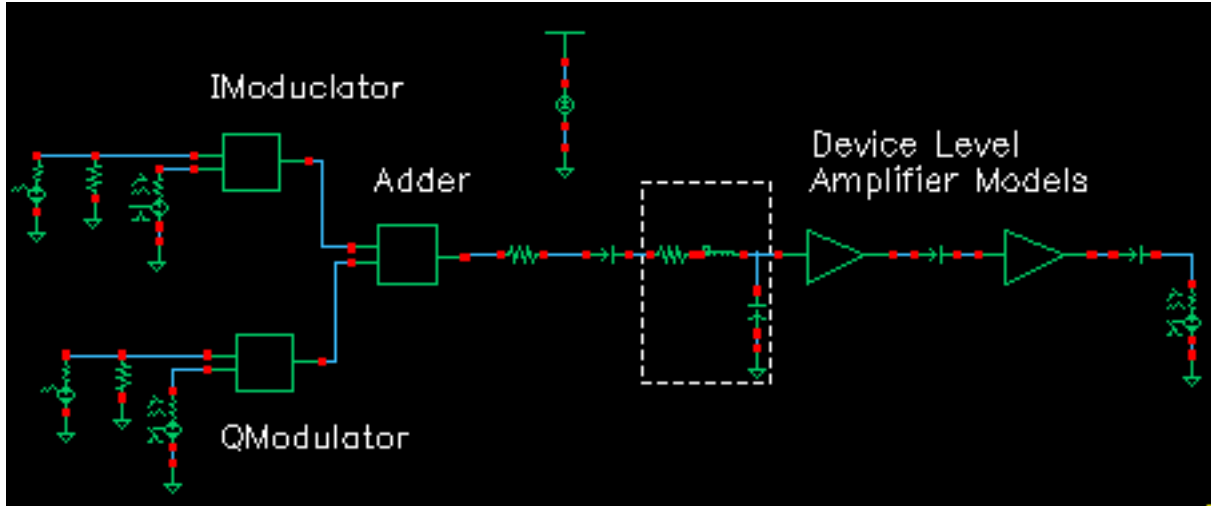
In the *Cell Name* list box, highlight *EF\_example*.

b. Choose *schematic* for *View Name*.

c. Highlight *edit* for *Mode*.

d. Click *OK*.

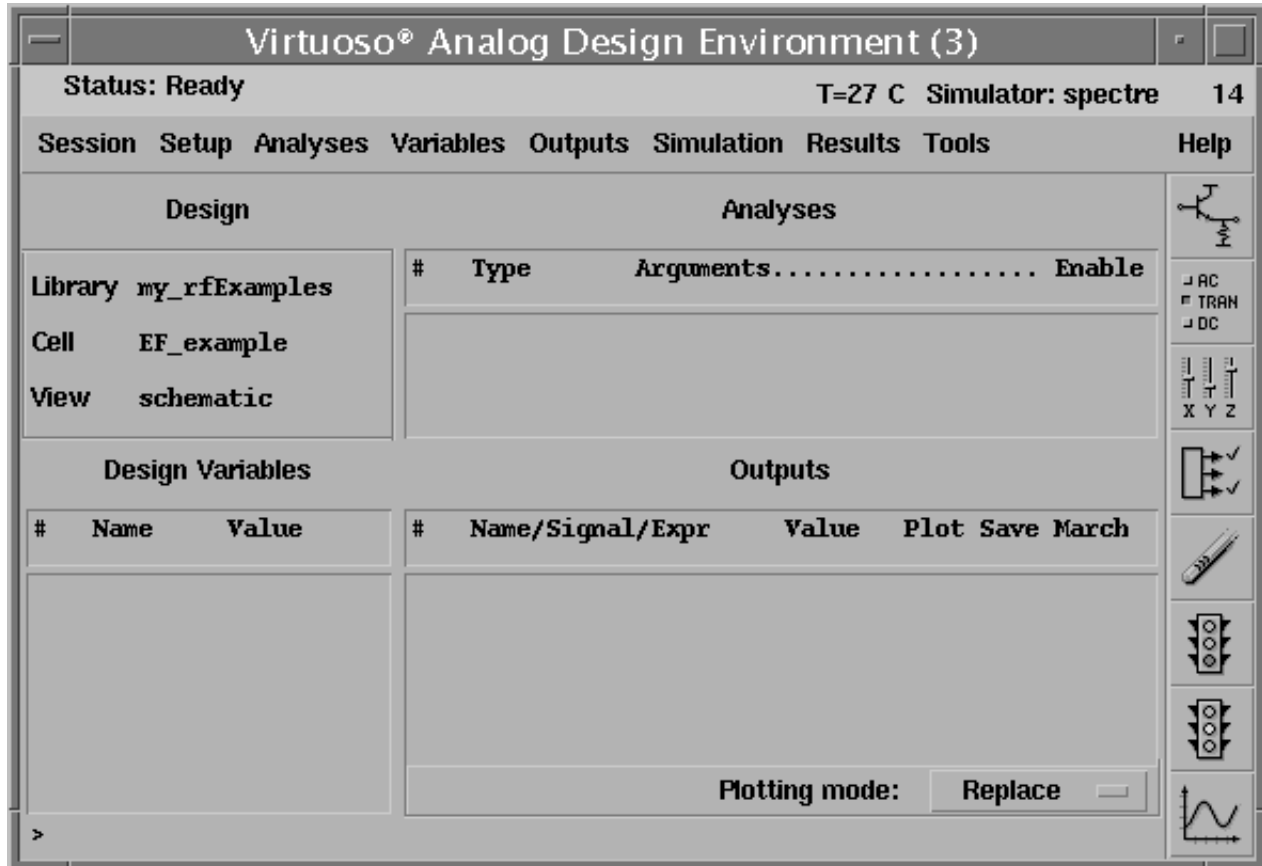
The Schematic window appears with the *EF\_example* schematic. This is a simple direct-conversion transmitter with ideal I/Q modulators.



## Opening the Simulation Window

1. In the Schematic window, choose *Tools – Analog Environment*.

The Simulation window opens.

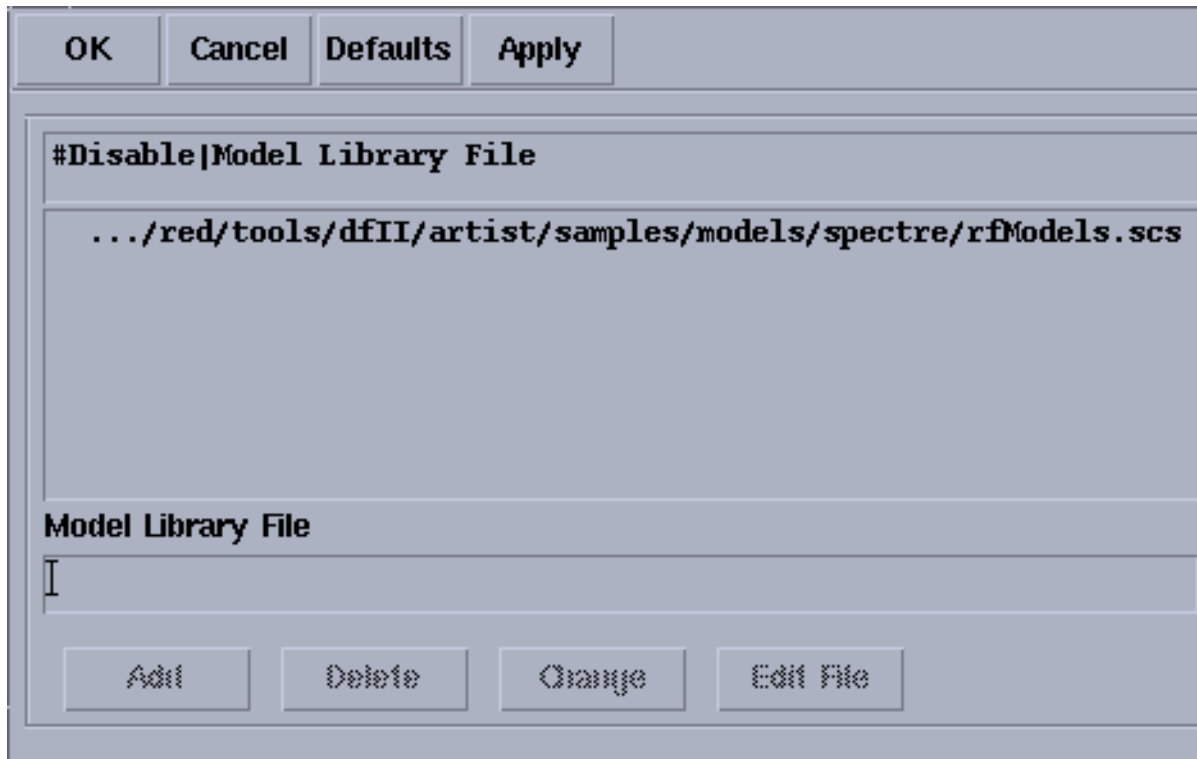


**Note:** You can also use *Tools – Analog Environment – Simulation* in the CIW to open the Simulation window without opening the design. You can open the design later by choosing *Setup – Design* in the Simulation window and choosing *EF\_example* in the Choosing Design form.

## Setting Up the Model Libraries

1. In the Simulation window, choose *Setup - Model Libraries*.
2. The Model Library Setup form appears.
3. In the *Model Library File* field, type the full path to the model file including the file name, where CDSHOME is the installation directory for the Cadence software.  
`<CDSHOME>/tools/dfII/samples/artist/models/spectre/rfModels.scs.`
4. Click *Add*.

The Model Library Setup form looks like the following.



5. First click *Add*. Then click *OK*.
6. Editing PORT0 and PORT1 in the EF\_example Schematic

The *EF\_example* circuit uses the programmable voltage source, *port*. The RF voltage source is based on the *port* sample component. You can easily change the behavior of this programmable component.

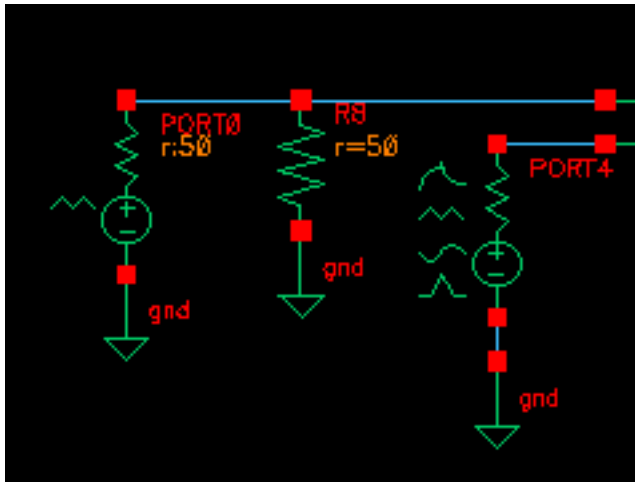
In the example, you edit PORT0 and PORT1 on the left side of the schematic.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

In the Schematic window, select `PORT0`. (The port in the top, left corner of the *EF\_example* schematic.)



1. In the Schematic window, choose *Edit – Properties – Objects*.

The Edit Object Properties form appears and changes to display information for `PORT0`. You use this form to change the list of CDF (component description format) properties for the `PORT0` and modify the schematic for this simulation.

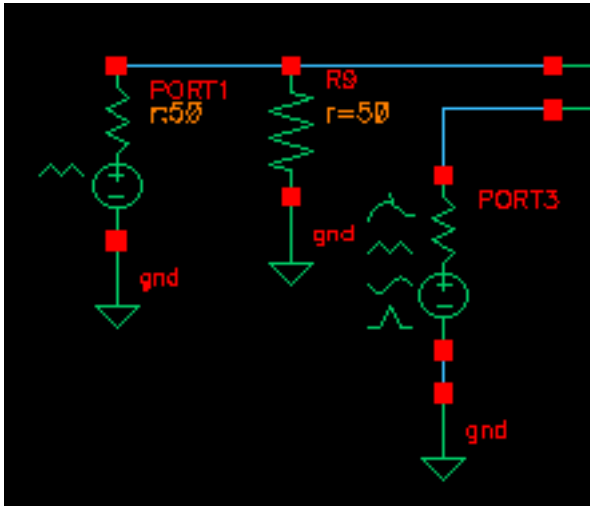
2. In the Edit Object Properties form, edit the *PWL filename* of `PORT0`. Type the following path, where *CDSHOME* is the installation directory for the Cadence software. Then click *Apply*.

```
<CDSHOME>/tools/dfII/samples/artist/rfLib/cdma_2ms_idata.pwl
```

The section of the Edit Object Properties form that includes the *PWL file name* field looks like the following.

CDF Parameter	Value
Frequency name for 1/period	<input type="text"/>
Noise file name	<input type="text"/>
PWL file name	<input type="text" value="flib/cdma_2ms_idata.pwl"/>
Number of noise/freq pairs	<input type="text" value="0"/>
Resistance	<input type="text" value="50 Ohms"/>

3. In the Schematic window, select `PORT1`. (The port in the bottom, left corner of the `EF_example` schematic.)



The Edit Object Properties form changes to display information for `PORT1`.

4. In the Edit Object Properties form, edit the *PWL filename* of `PORT1`. Type the following path, where `CDSHOME` is the installation directory for the Cadence software. Then click *Apply*.

```
<CDSHOME>/tools/dfII/samples/artist/rfLib/cdma_2ms_qdata.pwl
```

5. Click *OK*.
6. Choose *Design - Check and Save* in the Schematic window.

## Setting Up an Envelope Analysis

1. In the Simulation window, choose *Analyses – Choose*.

The Choosing Analyses form appears.

2. In the Choosing Analyses form, highlight *envlp*.

The Choosing Analyses form changes to let you specify values for the Envelope analysis.

3. In the Choosing Analyses form, do the following and then click *OK*.
4. Click *Select Clock Name* to display the Select Clock Name form. In the Select Clock Name form, click `fff` then click *OK*.
5. In the *Clock Name* field, `fff` appears.
6. Type `300u` for *Stop Time*.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- a. In the for *Output Harmonics* cyclic field, choose *Number of harmonics* and type 1 in the adjacent field.
- b. Highlight *moderate* for the *Accuracy Defaults (errpreset)* setting.
- c. Verify that *Enabled* is highlighted.
- d. The completed form looks like this.

**Envelope Following Analysis**

Engine  Shooting  Flexible Balance

Clock Name

Stop Time

Output Harmonics

Number of harmonics

Oscillator

Accuracy Defaults (errpreset)

conservative  moderate  liberal

Enabled

7. In the Simulation window, choose *Simulation – Netlist and Run*.

Look in the CIW for messages saying that the simulation has started and completed successfully. Watch the simulation log file for information as the simulation runs.

Looking at the Envelope results

8. In the Simulation window, choose *Results – Direct Plot – Main Form*.

The Direct Plot form and the Waveform window appear.

- 9.** In the Direct Plot form, do the following:
  - a.** Choose *Replace* for *Plotting Mode*.
  - b.** Highlight *envlp* for *Analysis*.
  - c.** Highlight *Voltage* for *Function*.
  - d.** Highlight *time* for *Sweep*.
  - e.** The *Select* cyclic field displays *Net* and a prompt displays at the bottom of the form.  
> *Select Net on Schematic....*

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The completed form looks like this.

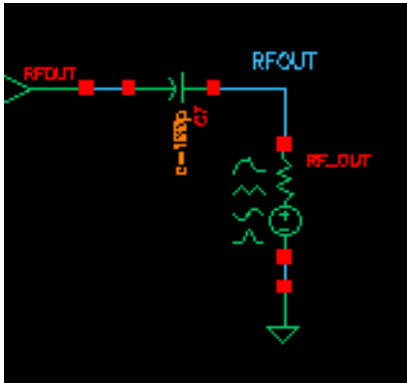
The image shows a dialog box titled "Direct Plot Form" with a standard Windows-style title bar (minimize, maximize, close buttons). The dialog contains several sections:

- Buttons:** "OK", "Cancel", and "Help" are located at the top.
- Plotting Mode:** A dropdown menu is set to "Replace".
- Analysis:** A radio button labeled "envlp" is selected.
- Function:** Three radio buttons are present: "Voltage" (selected), "Current", and "Power".
- Description:** The text "Envelope Voltage vs Time" is displayed.
- Select:** A dropdown menu is set to "Net".
- Sweep:** Three radio buttons are present: "spectrum", "harmonic time", and "time" (selected).
- Add To Outputs:** An unchecked checkbox.
- Footer:** A button labeled "> Select Net on schematic..." is at the bottom.

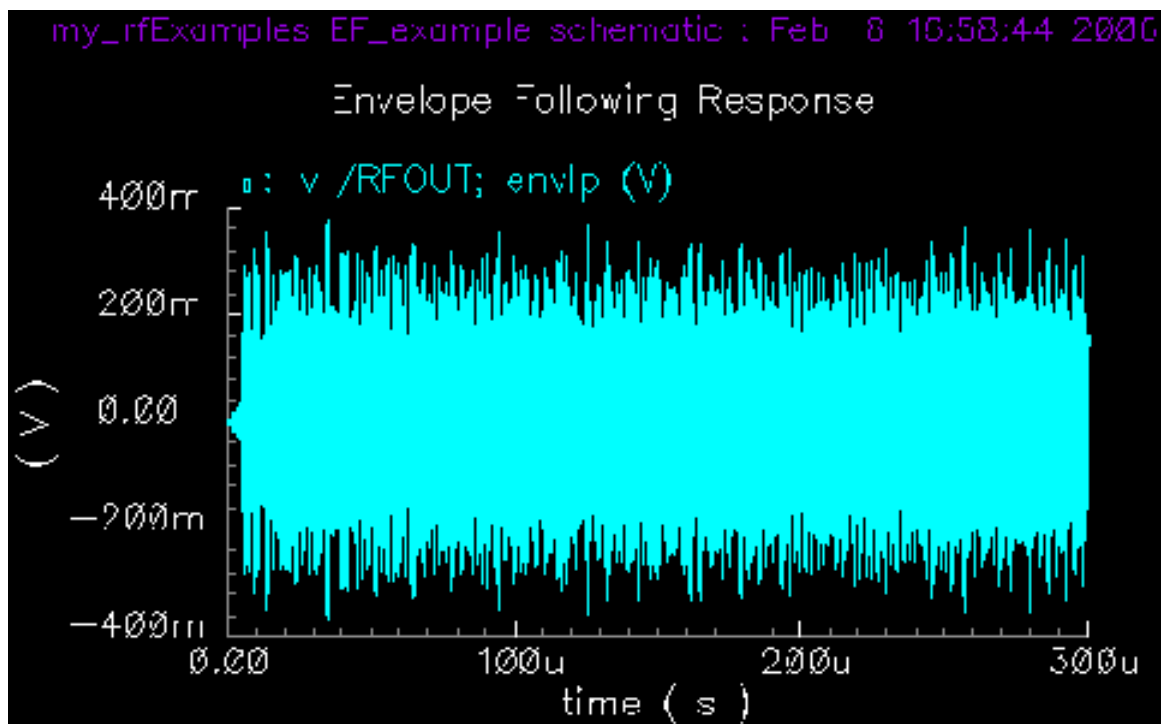
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

10. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the transmitter output net (/RFOUT)

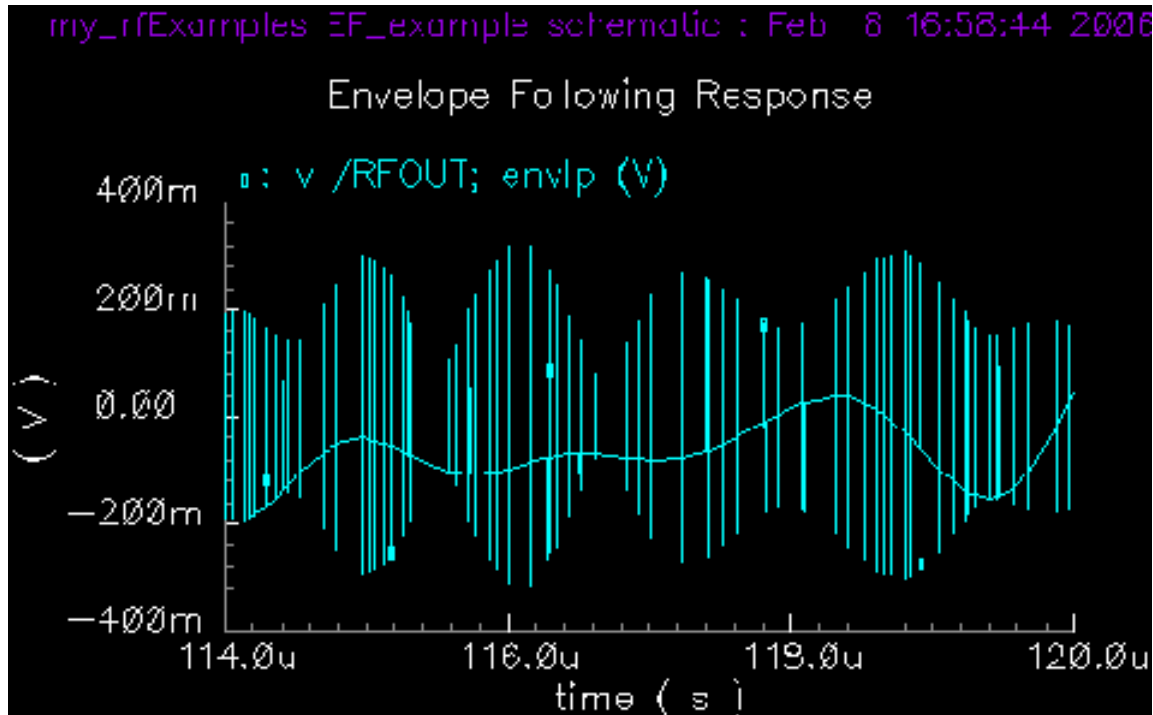


The voltage waveform for *RFOUT* appears in the Waveform window.



11. To get a closer look at a small section of the waveform, in the Waveform window, right click and drag to select a small section of the waveform. (The area you select covers a narrow vertical rectangle.)

You might have to do this several times to get a plot similar to the one shown.



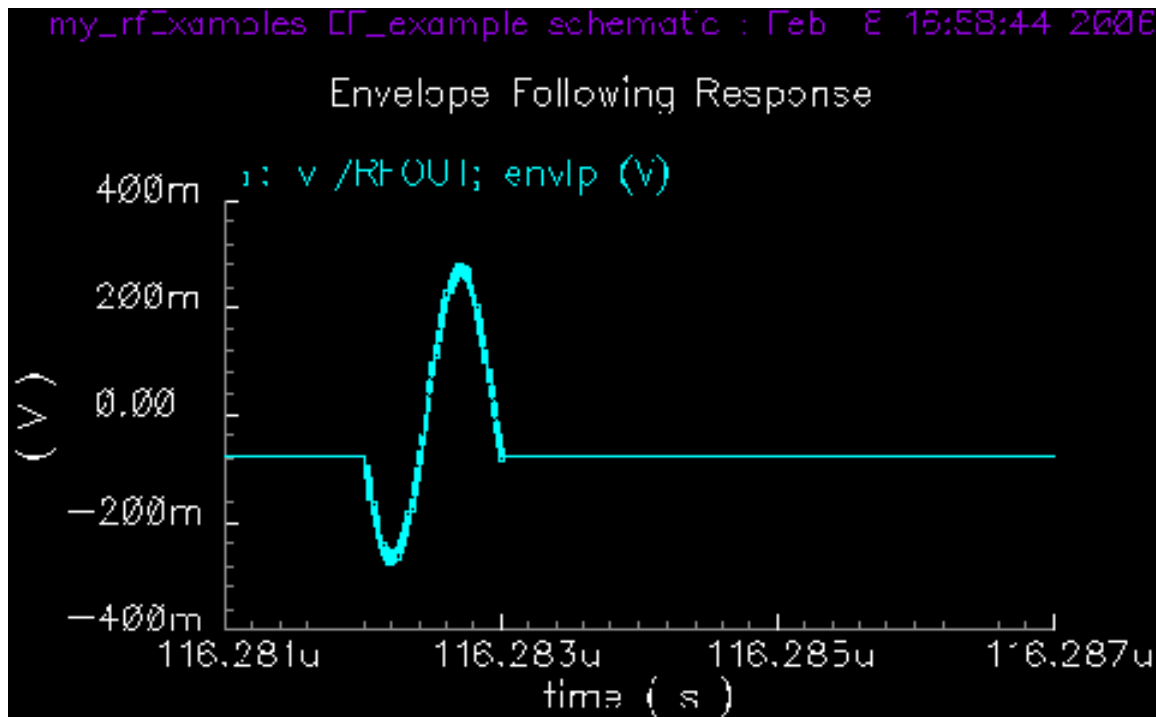
The plot changes to display a number of vertical lines with a wavy line running through them.

Each vertical line is a time point where a detailed calculation was performed. The wavy line connects these points

An Envelope analysis runs much faster than a Spectre transient analysis because Envelope analysis skips carrier cycles when it can do so while still satisfying numerical tolerances.

To get a closer look, right click and drag to select one of the vertical lines. You might have to do this several times.

After several selections, you are able to see the detailed simulation plot for one complete cycle. What you see should be similar to the following waveform depending on the areas you selected.



### Following the Baseband Signal Changes Through an Ideal Circuit

The modulation riding on the RF carrier is the baseband signal, the information to be transmitted. The baseband signal determines the amplitude and phase of the RF carrier. In transmitter design, it is important to determine how the transmitter might alter the baseband signal.

This section illustrates how to extract and plot the baseband signal at several points in the circuit.

1. In the Simulation window, choose *Results – Direct Plot – Main Form*.

The Direct Plot form and the Waveform window appear.

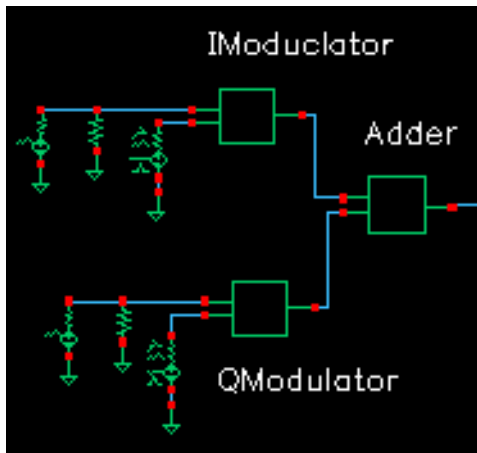
2. In the Direct Plot form, do the following:
  - a. Choose *Replace* for *Plotting Mode*.
  - b. Highlight *envlp* for *Analysis*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

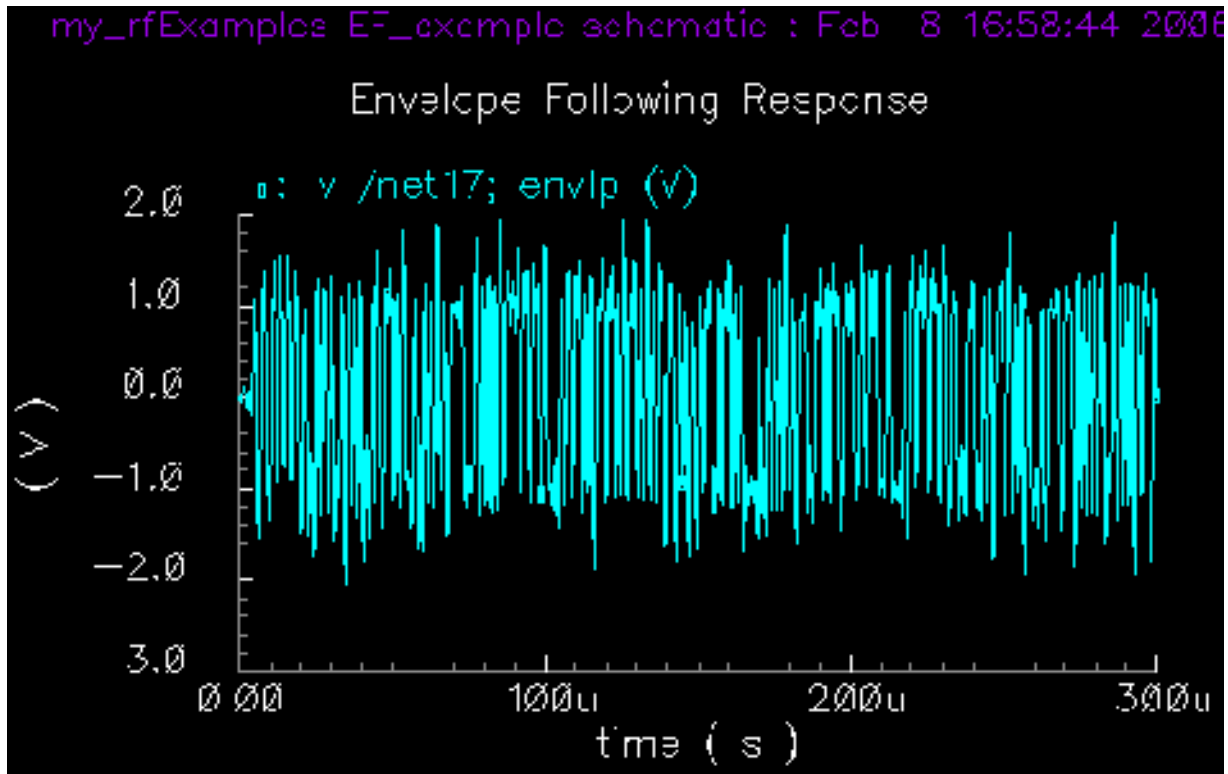
---

- c. Highlight *Voltage* for *Function*.
  - d. The *Select* cyclic field displays *Net* and a prompt at the bottom of the form. displays  
> Select Net on Schematic.
  - e. Highlight *time* for *Sweep*.
3. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the I-modulator source net (the I-modulator input net).



**Note:** You might have to click *Zoom – Fit* in the Waveform window before you are able to see the waveform.

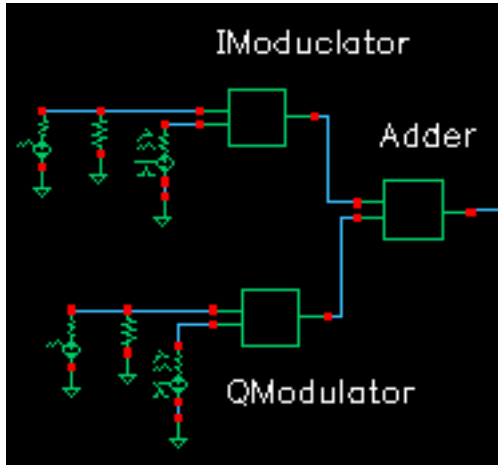
The plot for the modulation source waveform appears.



4. In the Waveform window, choose *Axes – To Strip*.
5. This changes the Waveform window to display multiple waveforms in strips, one above the other.
6. In the Direct Plot form, highlight *Append for Plotting Mode*.
7. This adds new waveforms to any waveforms currently displayed in the Waveform window.



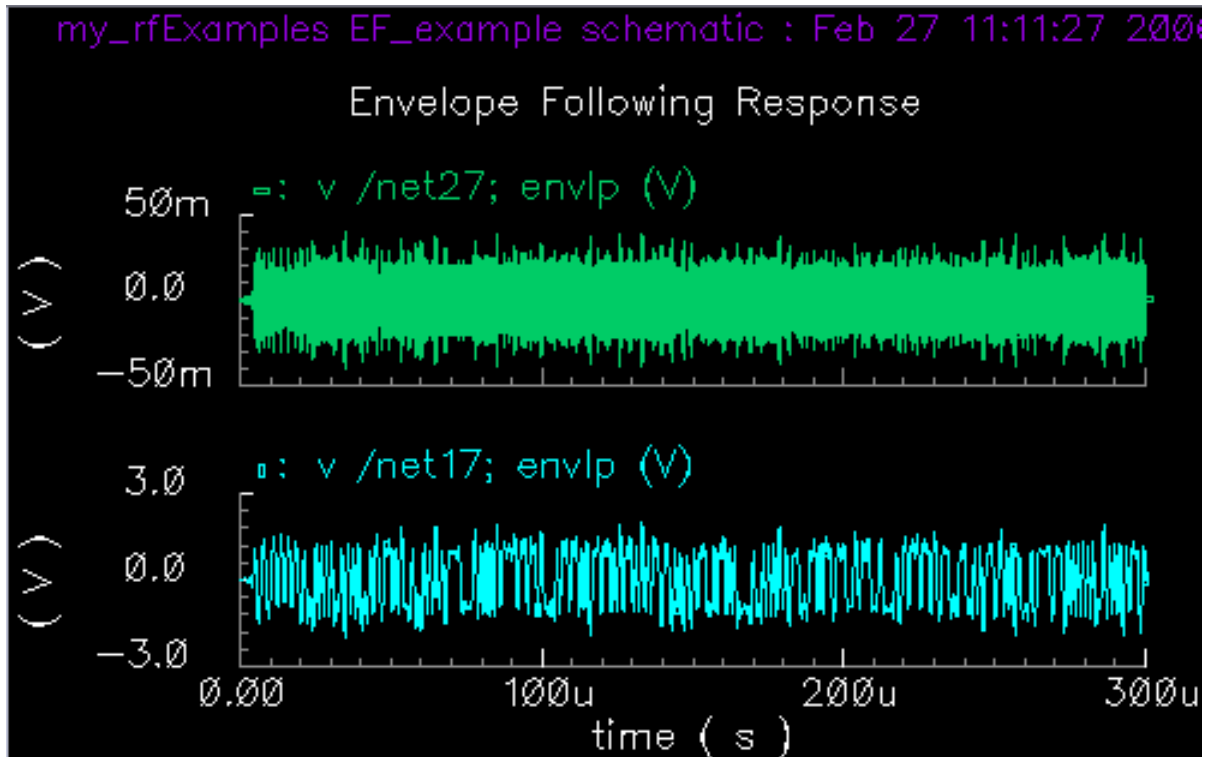
8. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the I-modulator output net as shown in the following schematic.



This is the in-phase carrier modulated by the I-component of the baseband signal. In this example,

- The in-phase carrier is  $\cos(\omega_c t)$ , where  $\omega_c$  is the carrier frequency in radians per second.
- The quadrature carrier is  $\sin(\omega_c t)$ .

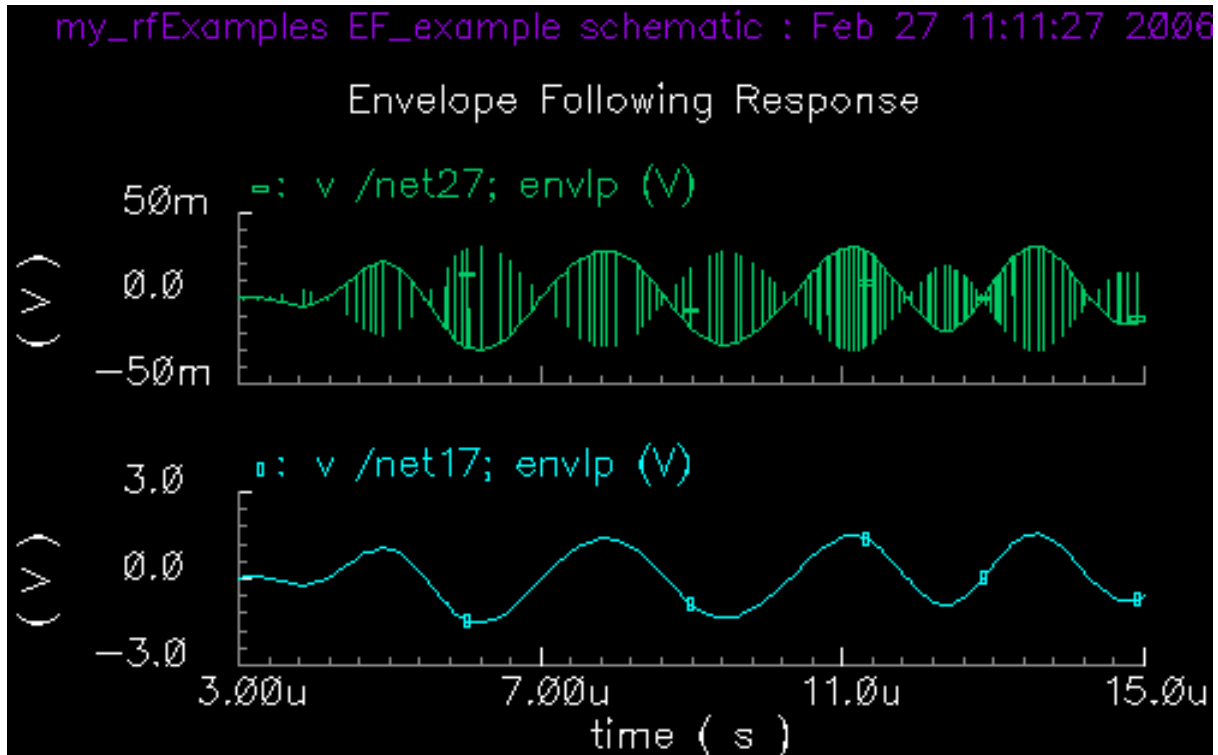
The I-modulator output waveform is added to the Waveform window. Note that the color of the selected net in the schematic is highlighted.



9. In the Waveform window, right click and drag over a narrow section of both waveforms.

You can see the modulation wave within the output wave. It is coincidental that the bottom waveform and smooth curve in the top waveform look alike. If the individual cycles in the

top waveform were sampled at a different phase, the smooth curve in the top waveform might look different.



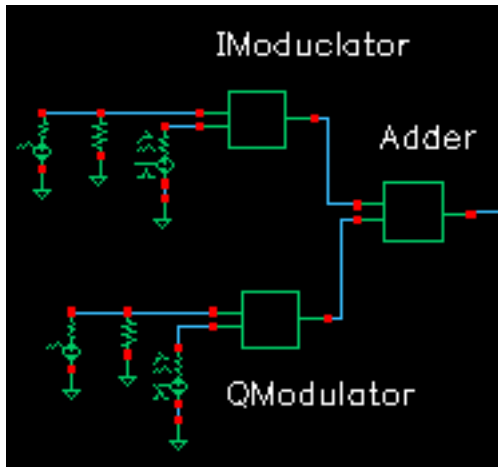
**Note:** The appearance of this plot, as well as the appearance of subsequent plots, depends on the section of the Waveform window in which you perform the click and drag.

10. In the Direct Plot form, do the following:
  - a. Highlight *harmonic time* for *Sweep*.
  - b. Highlight *Real* for *Modifier*.
  - c. Highlight *1* for *Harmonic Number*.
  - d. The *Select* cyclic field displays *Net* and a prompt at the bottom of the form.
  - e. > *Select Net on Schematic....*

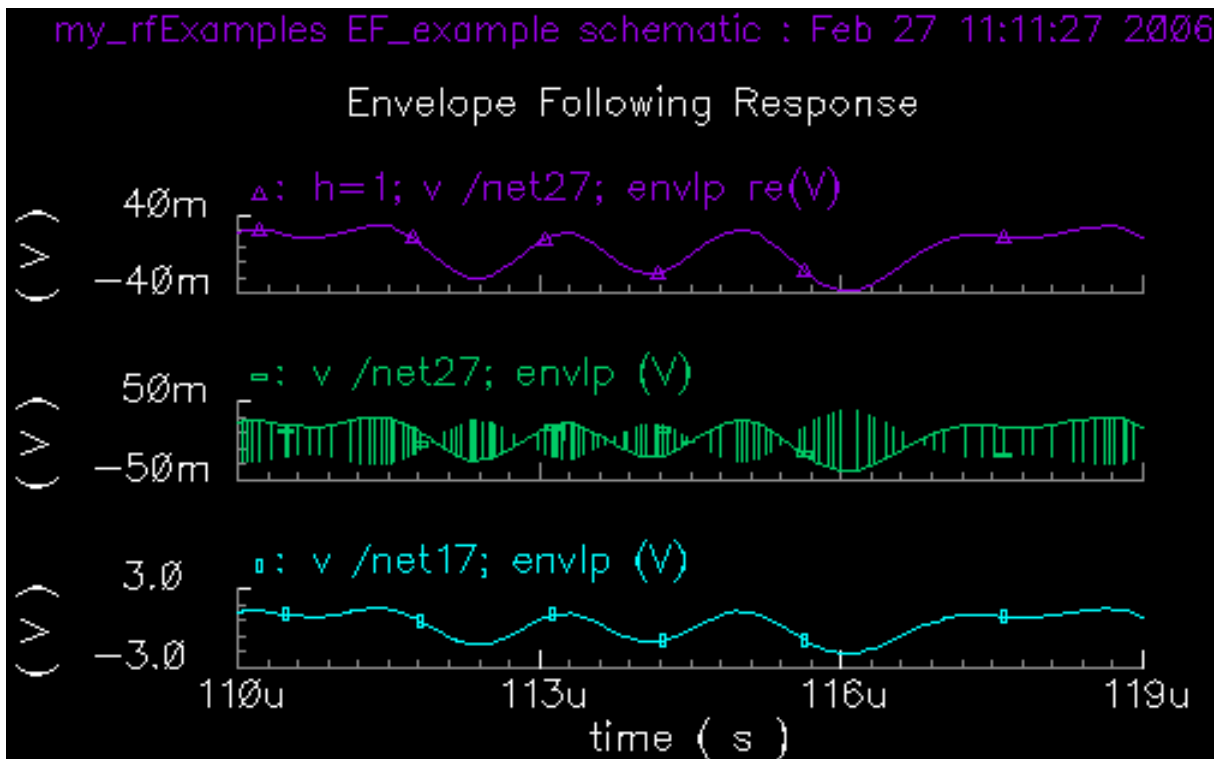
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the I-modulator output net.



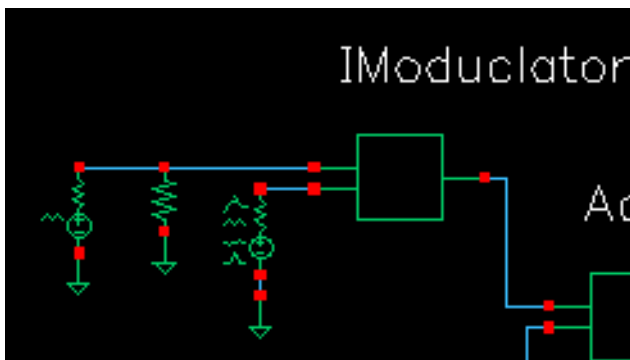
11. The top plot in the Waveform window is the baseband waveform recovered from the modulated RF carrier. Aside from a linear scale factor, It looks exactly like the bottom waveform because the modulator is ideal.



## Following the Baseband Signal Changes Through a Non-Ideal Circuit

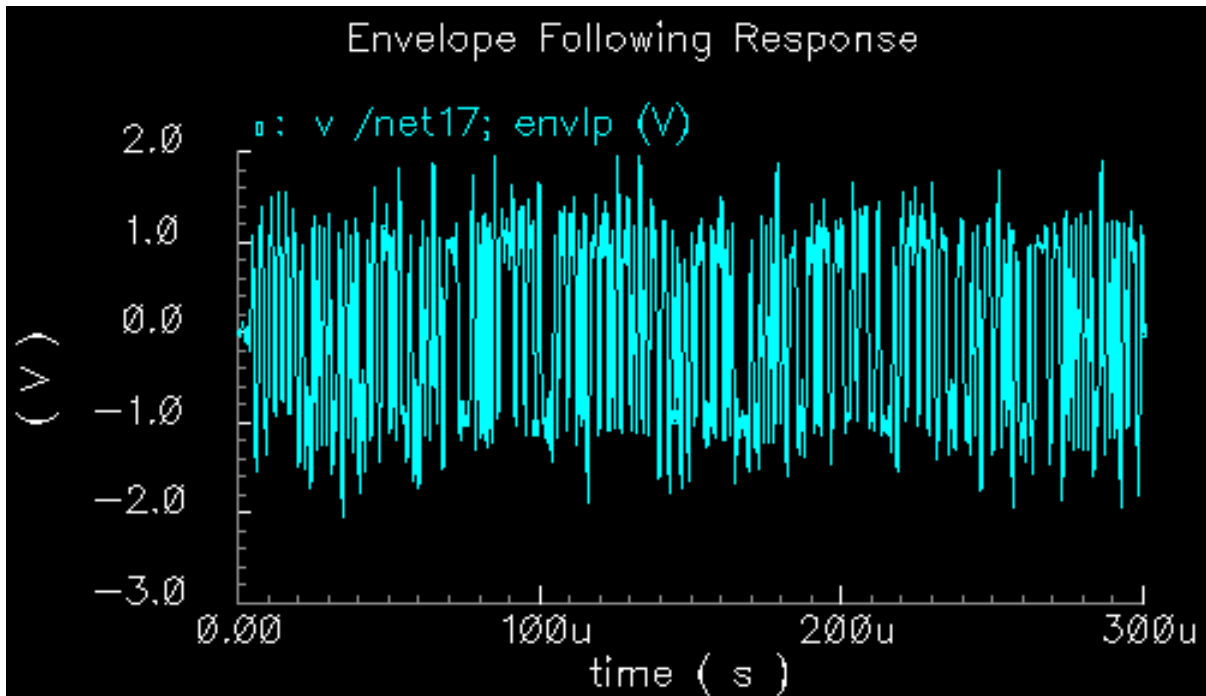
To see the change in the signal as it passes through the entire length of the circuit, which is not ideal, follow the instructions in this section.

1. In the Waveform window, choose *Window - Reset* to clear the Waveform window.
2. In the Direct Plot form, do the following:
  - a. Choose *Replace* for *Plotting Mode*.
  - b. Highlight *envlp* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.
  - d. Highlight *time* for *Sweep*.
  - e. The *Select* cyclic field displays *Net* and a prompt at the bottom of the form  
> Select Net on Schematic....
3. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the I-modulator source net.



The plot for the modulation source waveform appears.

You might have to choose *Zoom – Fit* in the Waveform window before you are able to see the waveform.

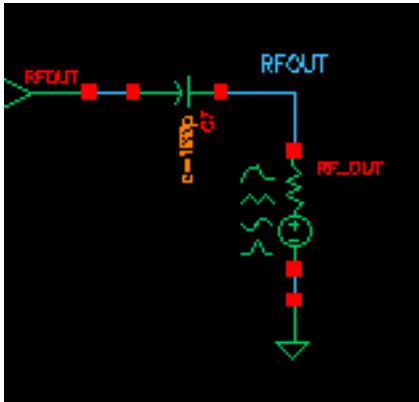


4. In the Waveform window, choose *Axes – To Strip*.
5. This changes the Waveform window to display multiple waveforms in strips, one above the other.
6. In the Direct Plot form, highlight *Append for Plotting Mode*.
7. This adds new waveforms to any waveforms currently displayed in the Waveform window.

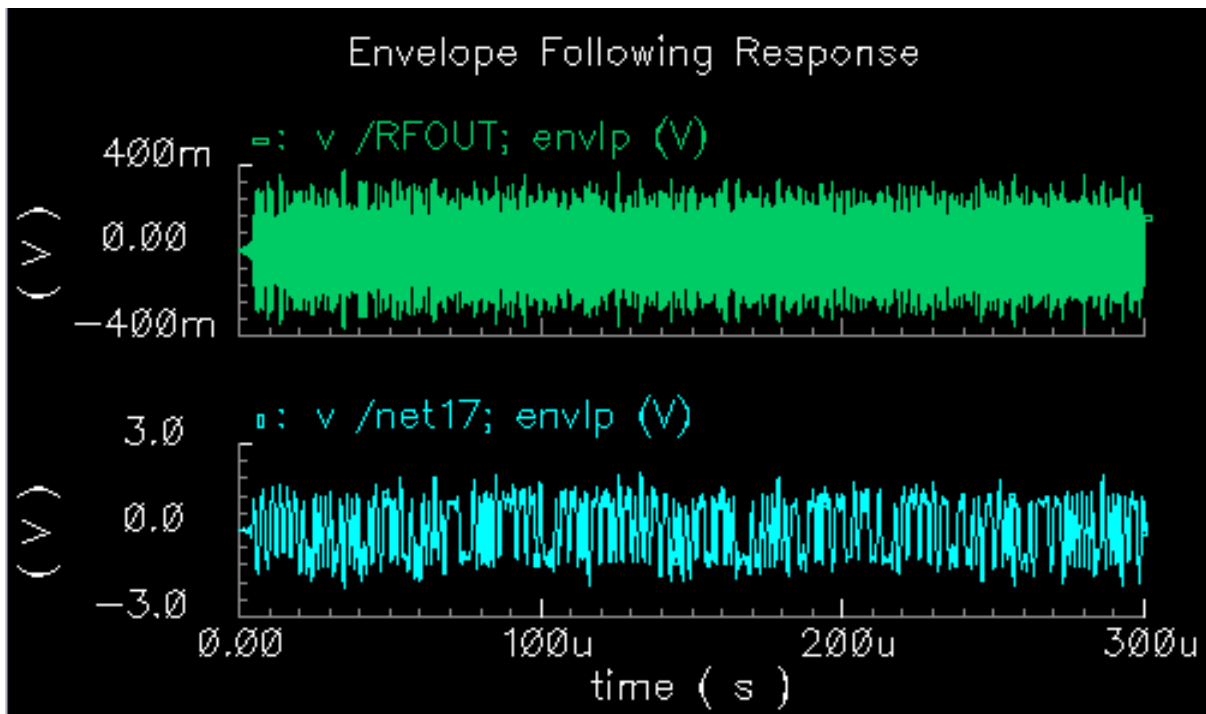
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

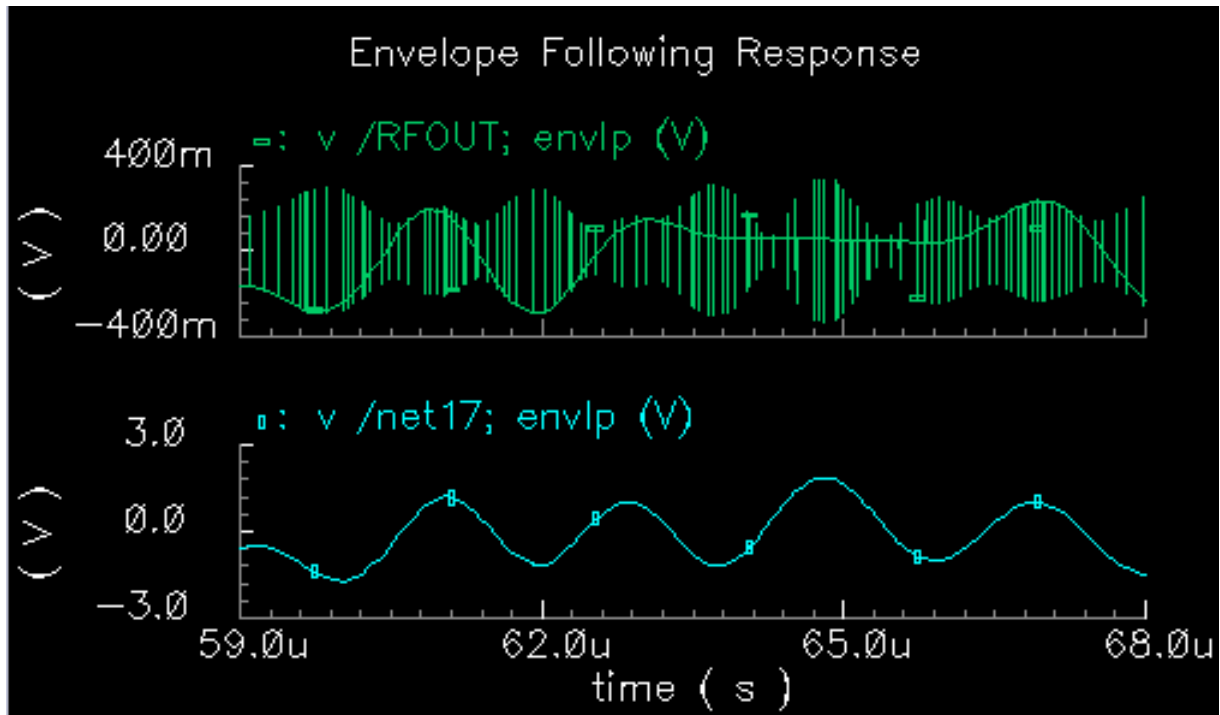
- Following the prompt at the bottom of the Direct Plot form, in the Schematic window, click the transmitter output net (RFOUT) as shown in the following schematic.



The transmitter output waveform is added to the Waveform window.



9. In the Waveform window, right click and drag over a narrow section of both waveforms.



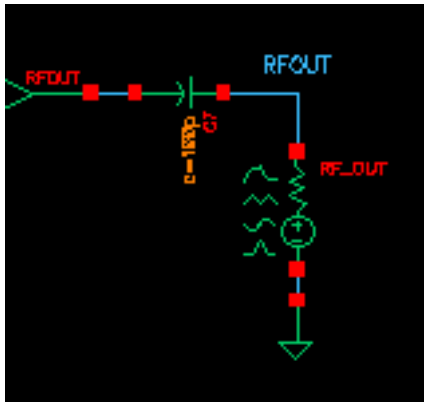
You can see the modulation wave within the output wave. It is coincidental that the bottom waveform and smooth curve in the top waveform look alike. If the individual cycles in the top waveform were sampled at a different phase, the smooth curve in the top waveform might look different.

**Note:** The appearance of this plot and of the subsequent plots depends on the section of the Waveform window in which you perform the click and drag.

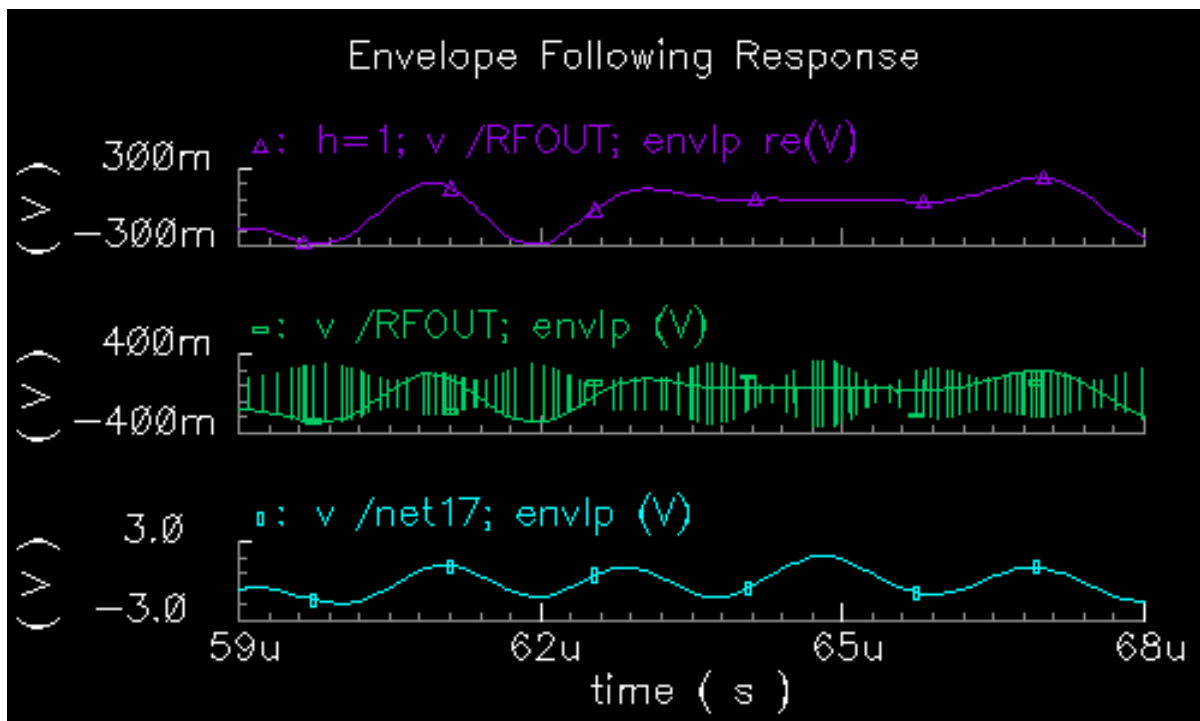
10. In the Direct Plot form, do the following:
- Highlight *harmonic time* for *Sweep*.
  - Highlight *Real* for *Modifier*.
  - Highlight *1* for *Harmonic Number*.



11. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the transmitter output net (RFOUT) as shown in the following schematic.



The waveform at the transmitter output appears at the top of the Waveform window. It shows a greater change in the signal.



## Plotting the Complete Baseband Signal

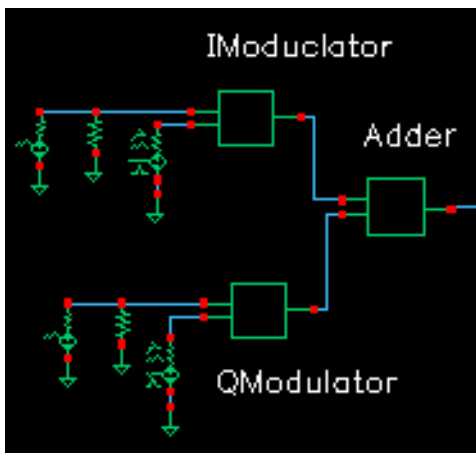
1. In the Direct Plot form, do the following:

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

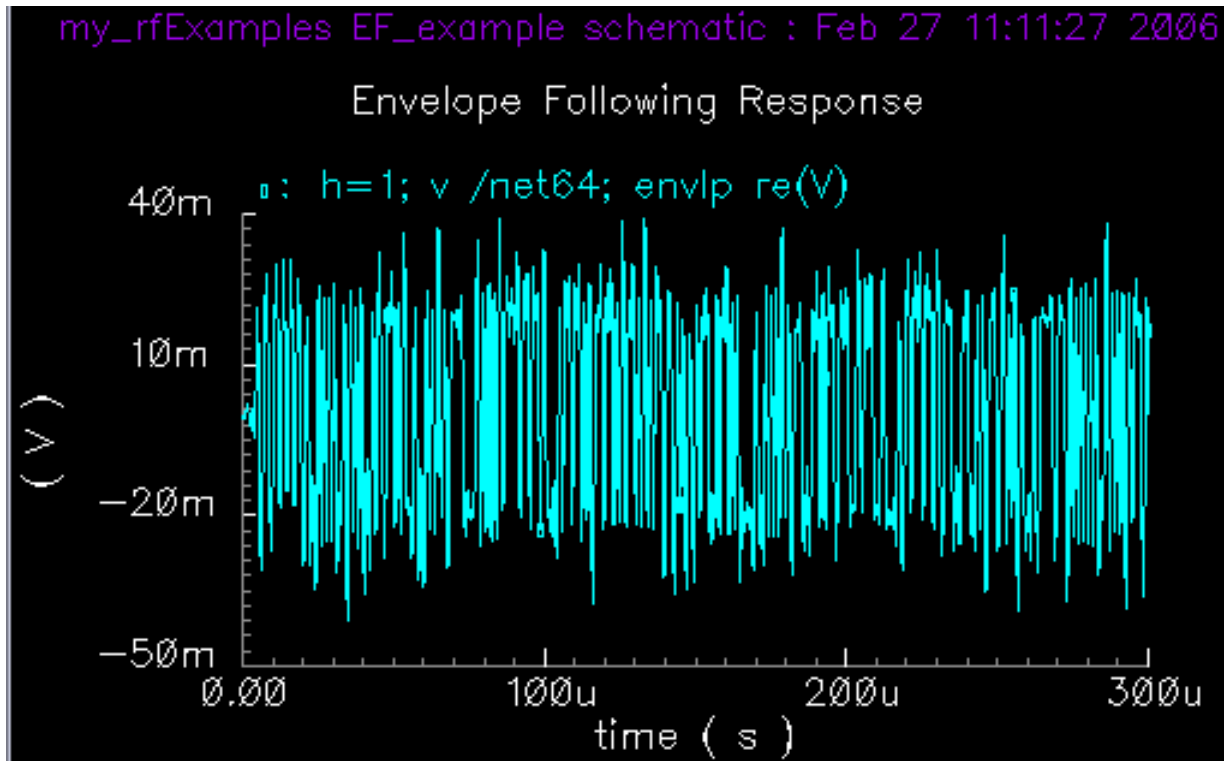
### Modeling Transmitters

---

- a. Choose *Replace* for *Plotting Mode*.
  - b. Highlight *envlp* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.
  - d. The *Select* cyclic field displays *Net* and a prompt at the bottom of the form.  
> Select Net on Schematic.
  - e. Highlight *harmonic time* for *Sweep*.
  - f. Highlight *Real* for *Modifier*.
  - g. Highlight *1* for *Harmonic Number*.
2. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the adder output net.

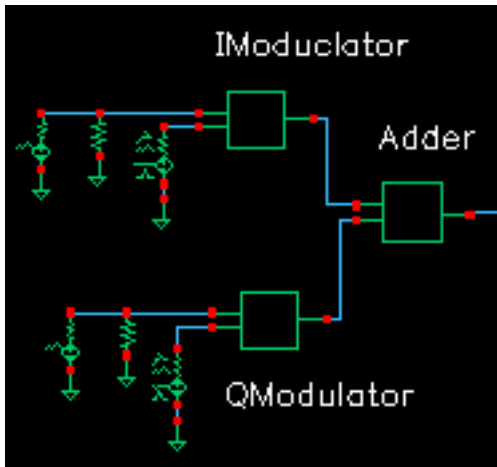


A plot for the real portion of the adder output waveform appears in the Waveform window.

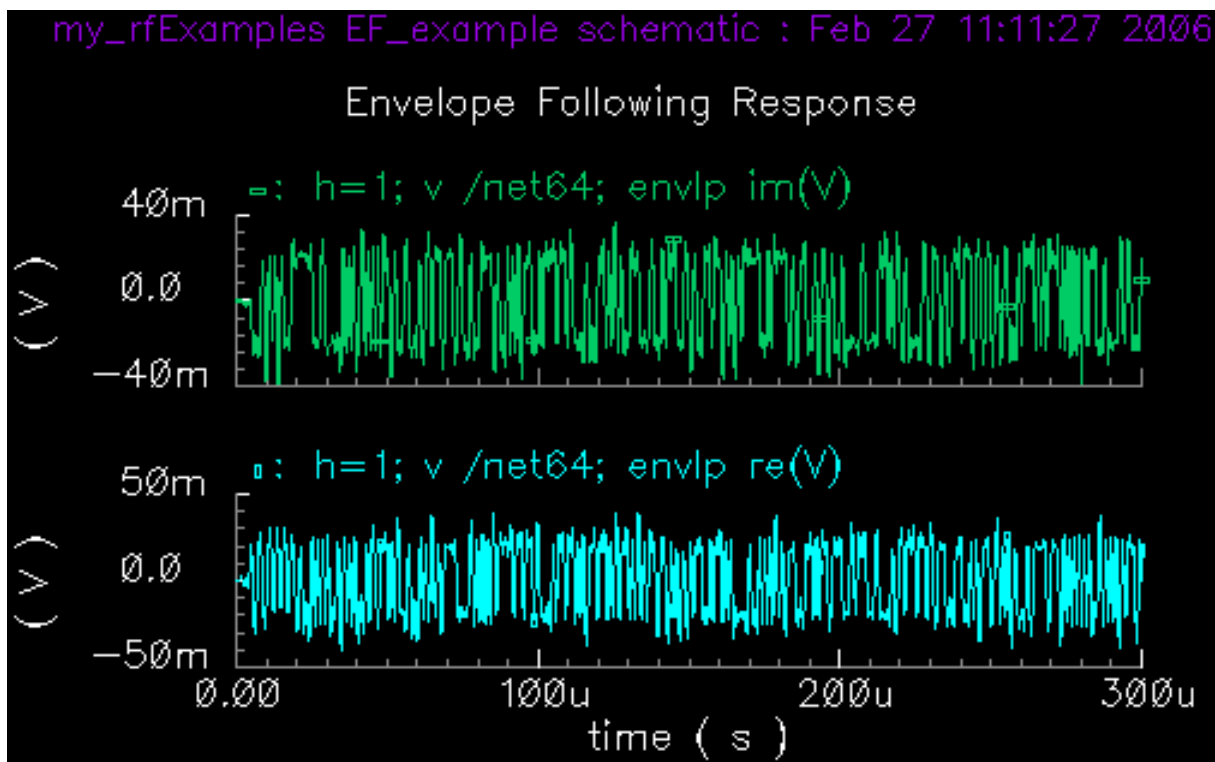


3. In the Direct Plot form, make the following changes:
  - a. Choose *Append* for *Plotting Mode*.
  - b. Highlight *Imaginary* for *Modifier*.

4. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the adder output net.

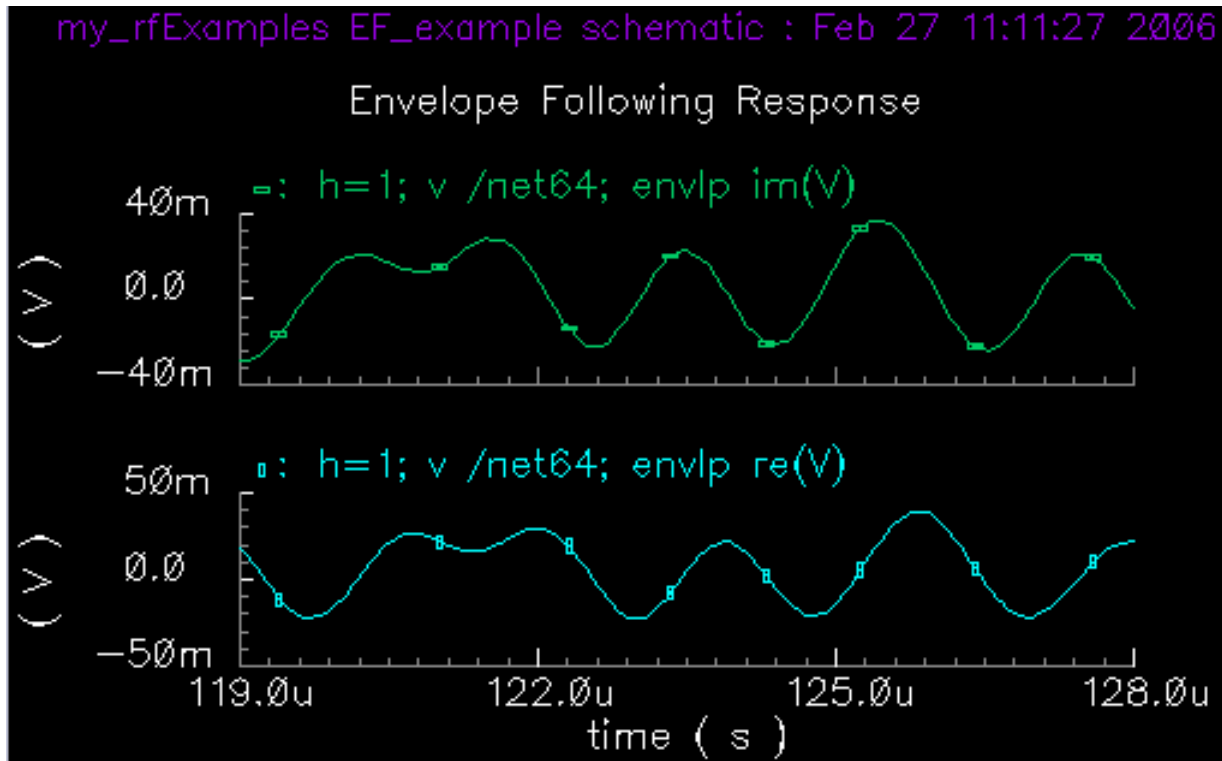


A plot for the imaginary portion of the adder output waveform is added to the Waveform window.



5. In the Waveform window, right click and drag to select a portion of the Waveform window over a narrow portion of the x-axis.

You can now clearly see both the real part and the imaginary part of the baseband signal.



## Plotting the Baseband Trajectory

This section describes how to display the baseband trajectory, the plot of one waveform against the other. The baseband waveforms recovered from the modulated RF carrier, as displayed in the previous section, do not directly reveal much information about how the transmitter affects them.

The baseband trajectory reveals much more information about the kind of distortion the transmitter introduces. The procedure described in this section displays first the input baseband trajectory followed by the output baseband trajectory. A comparison of the two trajectories reveals whether the power amplifiers in this example introduce any phase shift.

## Displaying the Input Baseband Trajectory

1. In the Waveform window, choose *Window - Reset*.
2. In the Direct Plot form, do the following:
  - a. Choose *Replace* for *Plotting Mode*.

- b.** Highlight *envlp* for *Analysis*.
- c.** Highlight *Voltage* for *Function*.
- d.** The *Select* cyclic field displays *Net* and a prompt at the bottom of the form.
  - > `Select Net on Schematic.`
- e.** Highlight *harmonic time* for *Sweep*.
- f.** Highlight *Real* for *Modifier*.
- g.** Highlight *1* for *Harmonic Number*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

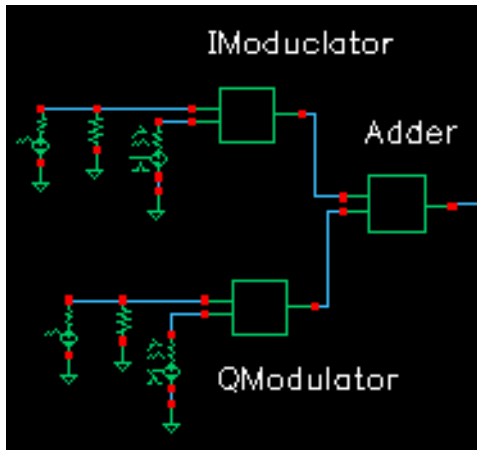
---

The completed form looks like this.

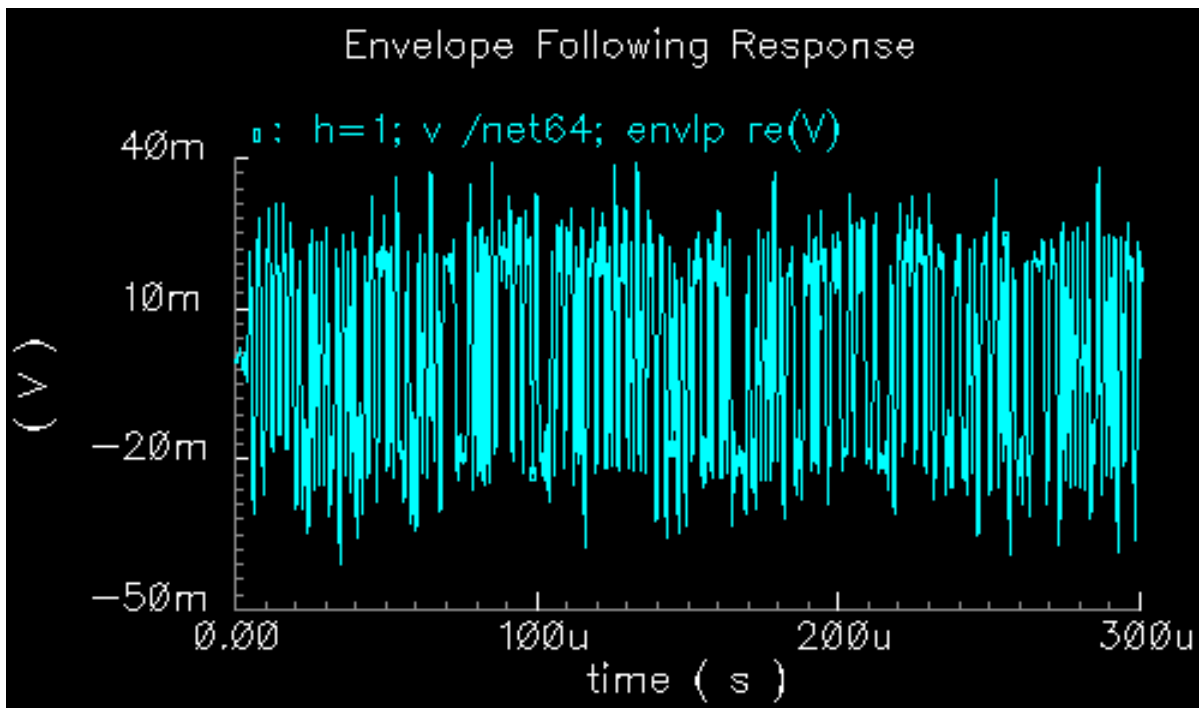
The image shows a dialog box titled "Direct Plot Form" with the following sections and controls:

- Buttons:** OK, Cancel, Help
- Plotting Mode:** Replace
- Analysis:**  envlp
- Function:**  Voltage,  Current,  Power
- Description:** Harmonic Voltage vs Time
- Select:** Net
- Sweep:**  spectrum,  harmonic time,  time
- Modifier:**  Magnitude,  Phase,  dB20,  Real,  Imaginary
- Harmonic Number:** A list box containing 0 and 1, with 1 selected.
- Buttons:** Add To Outputs (with an unchecked checkbox), Replot
- Footer:** > Select Net on schematic...

3. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the adder output net.



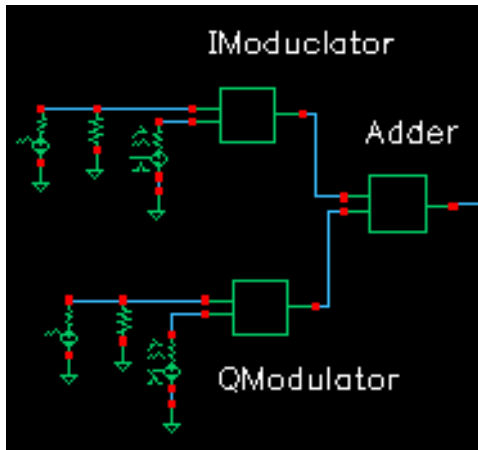
A plot for the real portion of the adder output waveform appears in the Waveform window.



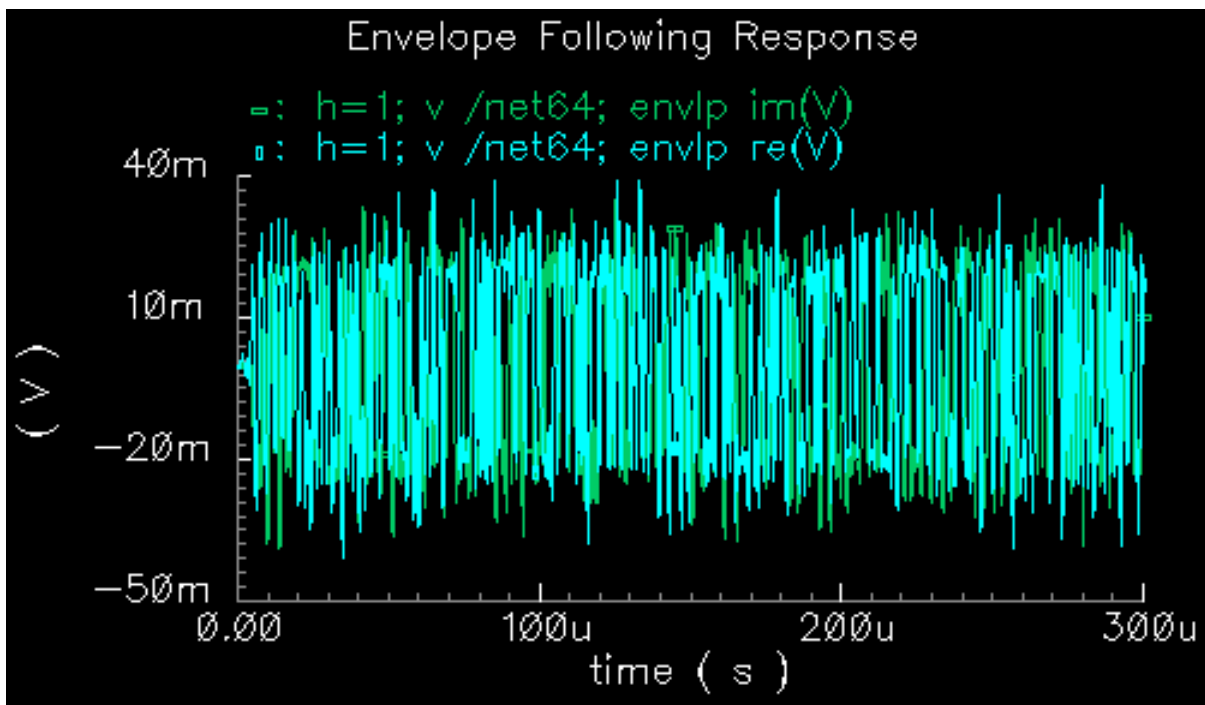
4. In the Direct Plot form, change the following:
  - a. Choose *Append* for *Plotting Mode*.
  - b. Highlight *Imaginary* for *Modifier*.



5. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the adder output net.



A plot for the imaginary portion of the adder output waveform is added to the real portion in the Waveform window.



6. In the Waveform window, choose *Axes – X Axis*.
7. The X Axis form appears.

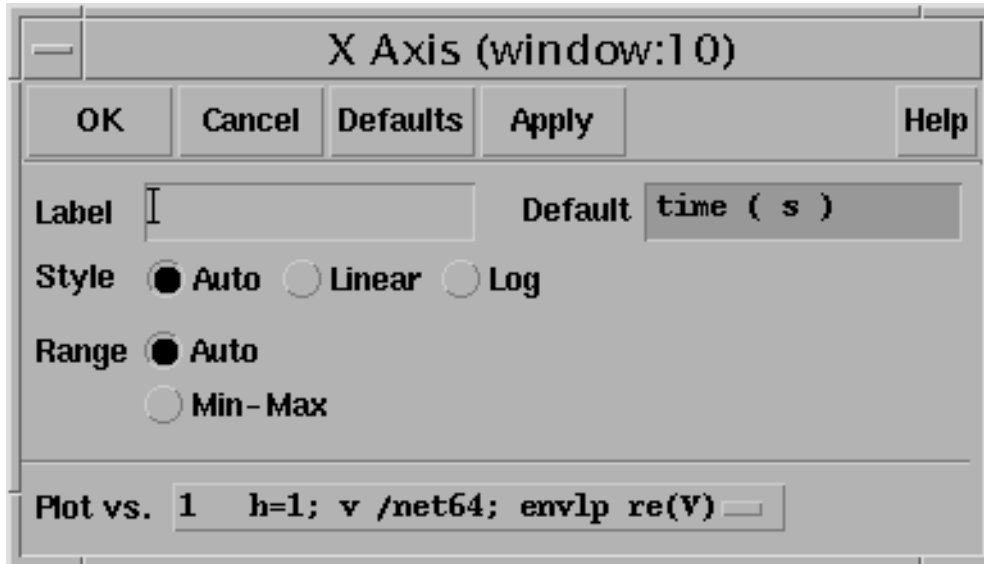
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

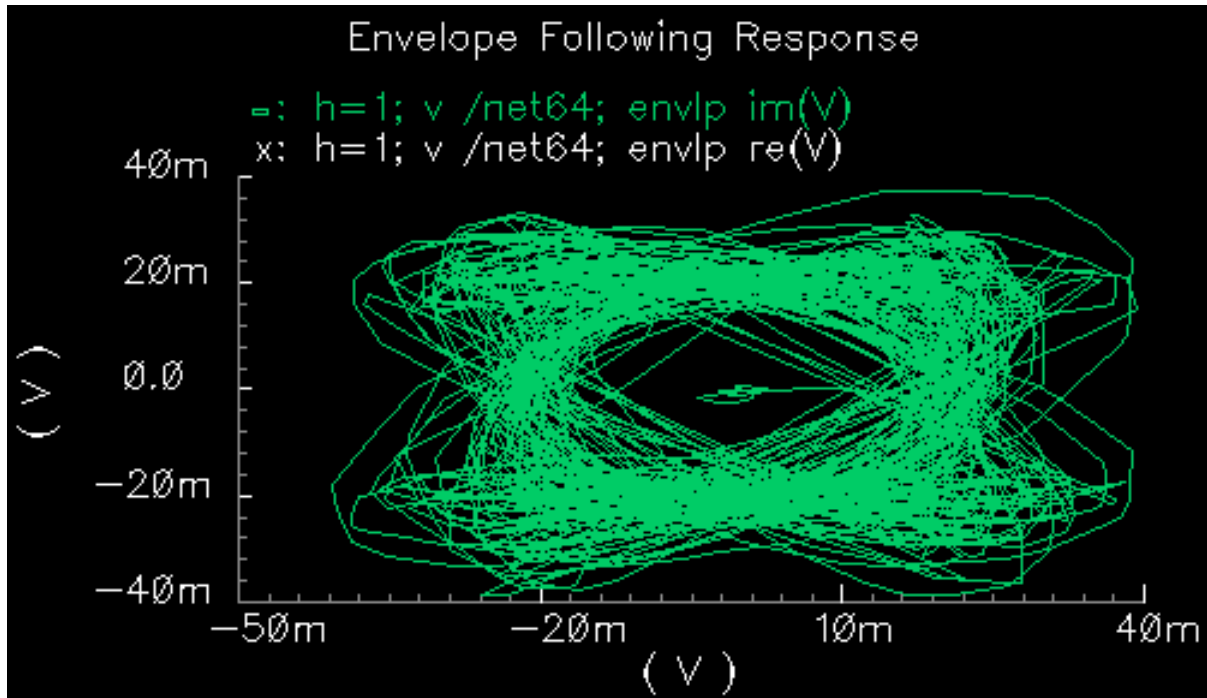
8. In the *Plot vs. cyclic* field of the X Axis form, select the real portion of the waveform. Choose the selection that contains  $re(V)$ . In this case, choose

```
1 h=1; v /net64; envlp re(V)
```



9. Click *OK*.

10. The input baseband trajectory, undistorted by the power amplifiers, appears in the Waveform window.



### Displaying the Output Baseband Trajectory

1. In the Simulation window, choose *Tools – Waveform*.  
A second Waveform window appears.
2. In the Direct Plot form, do the following:
  - a. Highlight *Replace* for *Plotting Mode*.
  - b. Highlight *envlp* for *Analysis*.
  - c. Highlight *Voltage* for *Function*.
  - d. The *Select* cyclic field displays *Net* and a prompt at the bottom of the form.  
> Select Net on Schematic.
  - e. Highlight *harmonic time* for *Sweep*.
  - f. Highlight *Real* for *Modifier*.
  - g. Highlight *1* for *Harmonic Number*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The completed form looks like this.

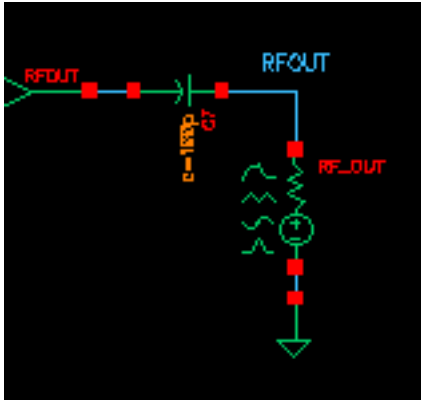
The image shows a dialog box titled "Direct Plot Form" with the following sections and controls:

- Buttons:** OK, Cancel, Help
- Plotting Mode:** Replace (dropdown menu)
- Analysis:**  envlp
- Function:**  Voltage,  Current,  Power
- Description:** Harmonic Voltage vs Time
- Select:** Net (dropdown menu)
- Sweep:**  spectrum,  harmonic time,  time
- Modifier:**  Magnitude,  Phase,  dB20,  Real,  Imaginary
- Harmonic Number:** A list box containing "0" and "1", with "1" selected.
- Buttons:** Add To Outputs (checkbox), Replot
- Footer:** > Select Net on schematic...

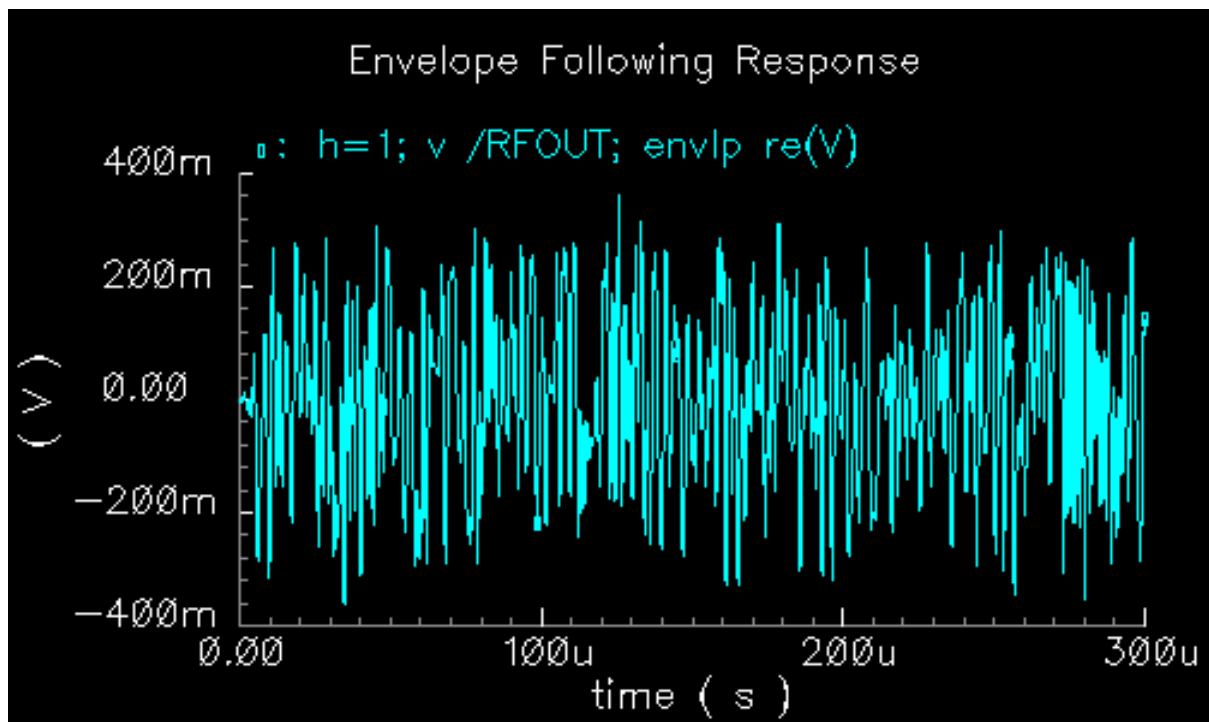
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

3. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the transmitter output net (/RFOUT).



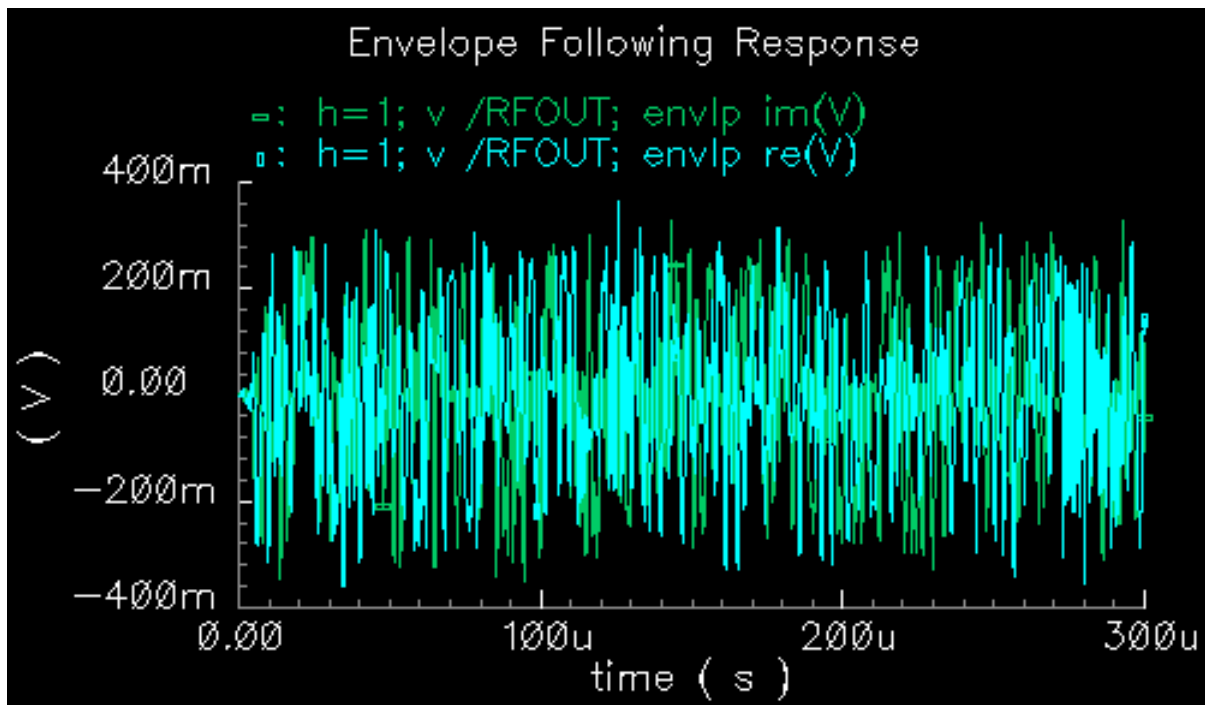
A plot of the real portion of the transmitter output waveform appears in the new Waveform window.



4. In the Direct Plot form, change the following:
  - a. Highlight *Append* for *Plotting Mode*.
  - b. Highlight *Imaginary* for *Modifier*.

5. Follow the prompt at the bottom of the Direct Plot form. In the Schematic window, click the transmitter output net (RFOUT).

A plot for the imaginary portion of the transmitter output waveform is added to the real portion in the Waveform window.



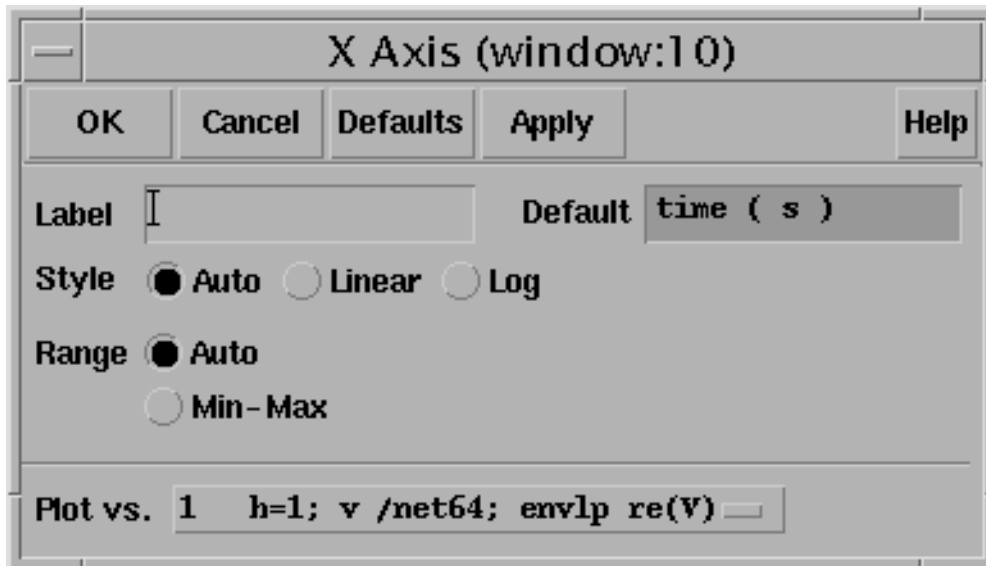
6. In the Waveform window, choose *Axes – X Axis*.
7. The X Axis form appears.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

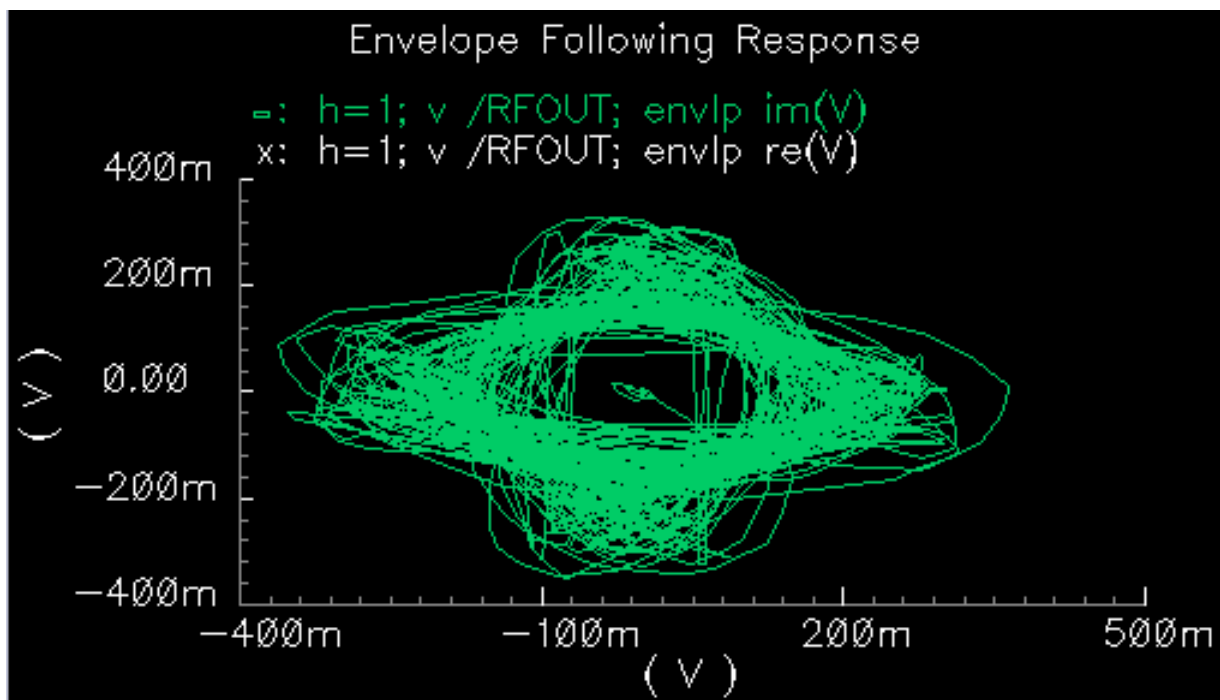
### Modeling Transmitters

- In the *Plot vs.* cyclic field of the X Axis form, select the real portion of the waveform. Choose the selection that contains  $re(V)$ . In this case, choose

```
1 h=1; v /RFOUT; envlp re(V)
```



- Click *OK*.
- The output baseband trajectory appears in the second Waveform window.



As shown in Figure , the output baseband trajectory (on the right) is the input baseband trajectory (on the left) scaled linearly and rotated.

### Input and Output Baseband Trajectories



In other words, the output baseband signal is the input baseband signal multiplied by a complex constant. The input and output waveforms look different because of the rotation, not because of some non-linear distortion. A common non-linear distortion, such as saturation, would make the outer edges of the trajectory lie on a circle.

## Measuring ACPR and PSD

Adjacent Channel Power Ratio (ACPR) is a common measure of the power a transmitter emits outside its allotted frequency band. ACPR is the ratio of the power in an adjacent band to the power in the allotted band.

$$ACPR = \frac{\text{Power in an adjacent band}}{\text{Power in the allotted band}}$$

Regardless of how you choose the frequencies and bands for the ACPR measurement, ACPR is always extracted from the power spectral density (PSD) of the transmitted signal. PSD is a frequency-by-frequency average of a set of discrete Fourier transforms (DFTs) of the baseband signal. Here, the baseband signal is the harmonic-time result of an *envlp* analysis.

### The ACPR Wizard

The ACPR Wizard simplifies the complicated calculations needed to measure ACPR. It determines the appropriate *envlp* simulation parameters and *psddb* function arguments from the information you supply in the ACPR wizard form. The ACPR wizard form contains a minimum number of clearly labeled fields so you can easily supply the required information.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

After you provide the information and press *Apply* or *OK* in the ACPR wizard, the calculated values are used to fill in the *envlp* Choosing Analyses form and the *envlp* Options form. You can then run the simulation and analyze the PSD and ACPR data.

Open the ACPR wizard in one of two ways.

In the Simulation window, choose *Tools – RF – Wizards – ACPR*

or

- In the *envlp* Choosing Analyses form, press *Start ACPR Wizard*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmitters

In either case the ACPR Wizard displays.

The screenshot shows the ACPR Wizard dialog box with the following sections and controls:

- Buttons:** OK, Cancel, Apply, Help
- Clock Name:** A text input field.
- How to Measure:** A dropdown menu set to "Net".
- Net:** A text input field and a "Select" button.
- Channel Definitions:** A dropdown menu set to "Custom".
- Main Channel Width (Hz):** A text input field.
- Adjacent frequencies are specified relative to the center of main channel:** A text label.
- Table:** A table with columns "name", "from (Hz)", and "to (Hz)". It contains two rows: "low" and "high".
- Buttons:** Add, Change, Delete
- Simulation Control:**
  - Stabilization Time (Sec):** A text input field with "0".
  - Resolution Bandwidth (Hz):** A text input field with "0" and a "Calculate" button.
  - Repetitions:** A text input field with "2".
- Windowing Function:** A dropdown menu set to "Cosine4".

## Measuring ACPR

The ACPR wizard minimizes the number of items you need to supply. Its fields are easy to understand and fill in. Once you provide the ACPR wizard field values, these values are used to calculate and fill in the field values on both the *envlp* Choosing Analyses and Options forms. The ACPR wizard computes the appropriate simulation parameters and `psdbb` arguments from the information you enter in the ACPR wizard.

To use the ACPR wizard,

1. Fill in the ACPR wizard form.
2. Click *OK* or *Apply* in the ACPR wizard to calculate field values for both the *envlp* Choosing Analyses and *envlp* Options forms.
3. Run the *envlp* analysis. When the simulation completes, the *envlp* analysis results are plotted.

This example illustrates how to use the ACPR wizard and to set up and run the Envelope analysis. This example uses the *EF\_example* circuit from your writable copy of the *rfExamples* library.

Before you start, perform the setup procedures described in [Chapter 3](#) if you have not yet set up the writable copy of the *rfExamples* library.

Set up the *EF\_example* schematic and the simulation environment.

Open the *EF\_example* schematic as described in [“Opening the EF\\_example Circuit in the Schematic Window”](#) on page 370.

Open the Simulation window as described in [“Opening the Simulation Window”](#) on page 372.

Set Up the Model Libraries as described in [“Setting Up the Model Libraries”](#) on page 373.

Edit the *PWL file names* for *PORT0* and *PORT1* as described in [“Editing PORT0 and PORT1 in the EF\\_example Schematic”](#) on page 374.

### Setting Up the ACPR Wizard and the *envlp* Analysis

1. In the Simulation window, choose *Analyses – Choose*.

The Choosing Analyses form appears.

2. In the Choosing Analyses form, highlight *envlp*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The Choosing Analyses form changes to let you specify values for the Envelope analysis.

**Envelope Following Analysis**

Engine  Shooting  Flexible Balance

Clock Name

Stop Time

Output Harmonics

Number of harmonics

Oscillator

Accuracy Defaults (empreset)

conservative  moderate  liberal

Enabled

3. In the *envlp* Choosing Analyses form, click *Start ACPR Wizard*.
4. Leave the *envlp* Choosing Analyses form open in the background.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

5. The ACPR wizard appears.

The screenshot shows the ACPR Wizard dialog box with the following sections and controls:

- Buttons:** OK, Cancel, Apply, Help
- Clock Name:** A text field with a dropdown arrow.
- How to Measure:** A dropdown menu set to "Net".
- Net:** A text field and a "Select" button.
- Channel Definitions:** A dropdown menu set to "Custom".
- Main Channel Width (Hz):** A text field.
- Adjacent frequencies are specified relative to the center of main channel:** A text label.
- Table:** A table with columns "name", "from (Hz)", and "to (Hz)". It contains two rows: "low" and "high".
- Buttons:** Add, Change, Delete
- Simulation Control:**
  - Stabilization Time (Sec):** A text field with "0".
  - Resolution Bandwidth (Hz):** A text field with "0" and a "Calculate" button.
  - Repetitions:** A text field with "2".
- Windowing Function:** A dropdown menu set to "Cosine4".

6. In the ACPR wizard, do the following.

7. In the *Clock Name* cyclic field, select *fff*.

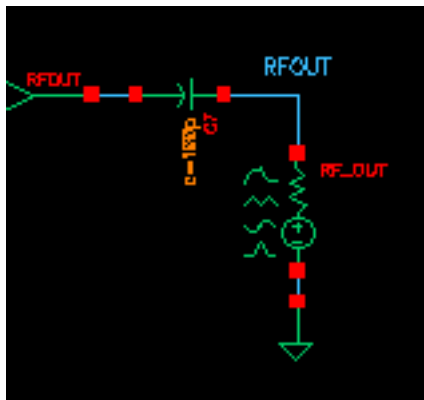
8. The clock name identifies the source of the modulated signal.

a. In the *How to Measure* cyclic field, select *Net*.

You can measure ACPR

- For a single *Net* (Select one net, *Net*)
- Between *Differential Nets* (Select two nets, *Net+* and *Net-*)

b. To select a single output net in the Schematic window, click *Select* to the right of the *Net* field. In the Schematic window, select the transmitter output net, *RFOUT*.



The output net name, */RFOUT*, displays in the *Net* field.

The top of the ACPR wizard appears below.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- c. In the *Channel Definitions* cyclic field, select the *IS-95* standard.
- d. The *Main Channel Width (Hz)* field is calculated as 1.2288M.
- e. The *adjacent frequencies* are determined and display in the list box.
- f. Note the text on the ACPR Wizard form,

*Adjacent frequencies are specified relative to the center of the main channel*

- *Channel Definitions* presets include *Custom*, *IS-95*, and *W-CDMA*
- *Main Channel Width (Hz)* specifies a frequency band in Hz

The middle of the ACPR wizard appears below.

name	from (Hz)	to (Hz)		
lower	-915K	-885K		
upper	885K	915K		

Use the edit fields below the list box and the *Add*, *Change* and *Delete* buttons to modify channel definitions in the list box.

You can hand edit or enter adjacent frequency names and *upper* and *lower* boundaries. Specify adjacent frequencies relative to the center of the main channel. All channel widths must be greater than zero.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- g. In the *Stabilization Time (Sec)* field, enter 72n.
- h.  $7.2e^{-08}$  displays in the *Stabilization Time (Sec)* field and 2 displays in the *Repetitions* field.
- i. The *Stabilization Time* is the length of time in seconds to wait before using the data for analysis. *Repetitions* is the number of times to repeat the discrete Fourier transform for averaging.
- j. Increasing the number of repetitions makes the power density curve smoother, but at the expense of longer simulation time and increased data file size.
- k. Click *Calculate* to determine the *Resolution Bandwidth (Hz)*.
- l. 7500 displays in the *Resolution Bandwidth (Hz)* field.
- m. *Resolution Bandwidth* specifies the spacing of data points on the resulting power density curve, in Hz.
- n. Reducing *Resolution Bandwidth* increases simulation time and data file size.
- o. In the *Windowing Function* cyclic field, select *Cosine4*.
- p. *Windowing Function* presets include *Blackman*, *Cosine2*, *Cosine4*, *ExtCosBell*, *HalfCycleSine*, *HalfCycleSine3*, *HalfCycleSine6*, *Hamming*, *Hanning*, *Kaiser*, *Parzen*, *Rectangular* and *Triangular*.
- q. The *Simulation Control* section at the bottom of the ACPR Wizard looks like the following.

The screenshot shows a dialog box titled "Simulation Control" with the following fields and controls:

- Stabilization Time (Sec)**: A text input field containing the value  $7.2e-08$ .
- Resolution Bandwidth (Hz)**: A text input field containing the value 7500, with a **Calculate** button to its right.
- Repetitions**: A text input field containing the value 2.
- Windowing Function**: A dropdown menu currently displaying "Cosine4".

- 9. In the APCR Wizard, click *Apply*



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

When you click *OK* or *Apply* in the ACPR wizard, values are calculated and appear in the *envlp* Choosing Analyses form.

If necessary select *Choose — Analyses* in the Simulation window to display the *envlp* Choosing Analyses form.

**Envelope Following Analysis**

**Engine**     Shooting    Flexible Balance

**Clock Name**     

**Stop Time**  

**Output Harmonics**

**Number of harmonics**     

**Oscillator**  

**Accuracy Defaults (errpreset)**

conservative    moderate    liberal

**Enabled**     

Values in the *envlp* Choosing Analyses form are as follows.

The *Clock Name* is the same in both forms. In this case *fff*.

*Stop Time* for *envlp* is calculated as 0.0002669271.

For the *envlp* analysis *Output Harmonics*, *Number of Harmonics* is selected in the cyclic field and the *Number of Harmonics* is set to 1.

- The *envlp* analysis is *Enabled*

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- ❑ The *envlp* analysis *errpreset* is *moderate*.
- ❑ Values in the *envlp* Options form are as follows.
- ❑ The *envlp* option *start* is set to blank.
- ❑ The *envlp* option *modulationbw* is calculated as 1098000.0
- ❑ The *envlp* option *strobeperiod* is calculated as 2.604167e-07.
- ❑ You can modify values on the *envlp* Choosing Analyses and Options forms but your changes are not propagated back to the ACPR wizard.
- ❑ In the *envlp* Choosing Analyses form, click *Options* to open the *envlp* Options form.
- ❑ Notice that values for the *modulationbw* and *strobeperiod* parameters are calculated and the *start* parameter is blank.

The section for the *start* and *modulationbw* parameters looks like this.

The screenshot displays a form titled "SIMULATION INTERVAL PARAMETERS" and "SIMULATION BANDWIDTH PARAMETERS".

**SIMULATION INTERVAL PARAMETERS**

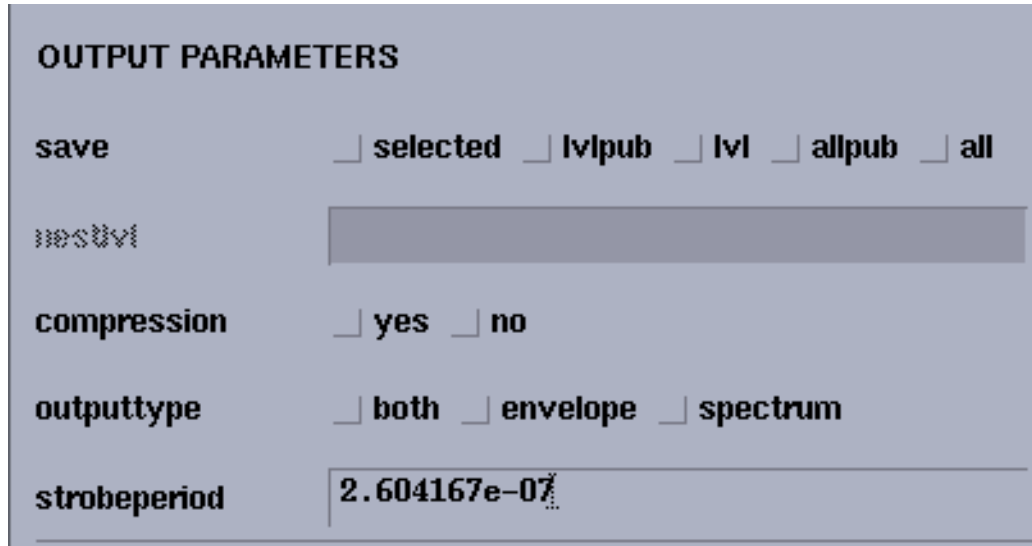
<b>start</b>	<input type="text"/>
<b>outputstart</b>	<input type="text"/>
<b>tstab</b>	<input type="text"/>

---

**SIMULATION BANDWIDTH PARAMETERS**

<b>modulationbw</b>	<input type="text" value="1098000.0"/>
---------------------	--

The section for the *strobeperiod* parameter looks like this.



The screenshot shows a dialog box titled "OUTPUT PARAMETERS". It contains several parameters with radio button options and a text input field. The parameters are:

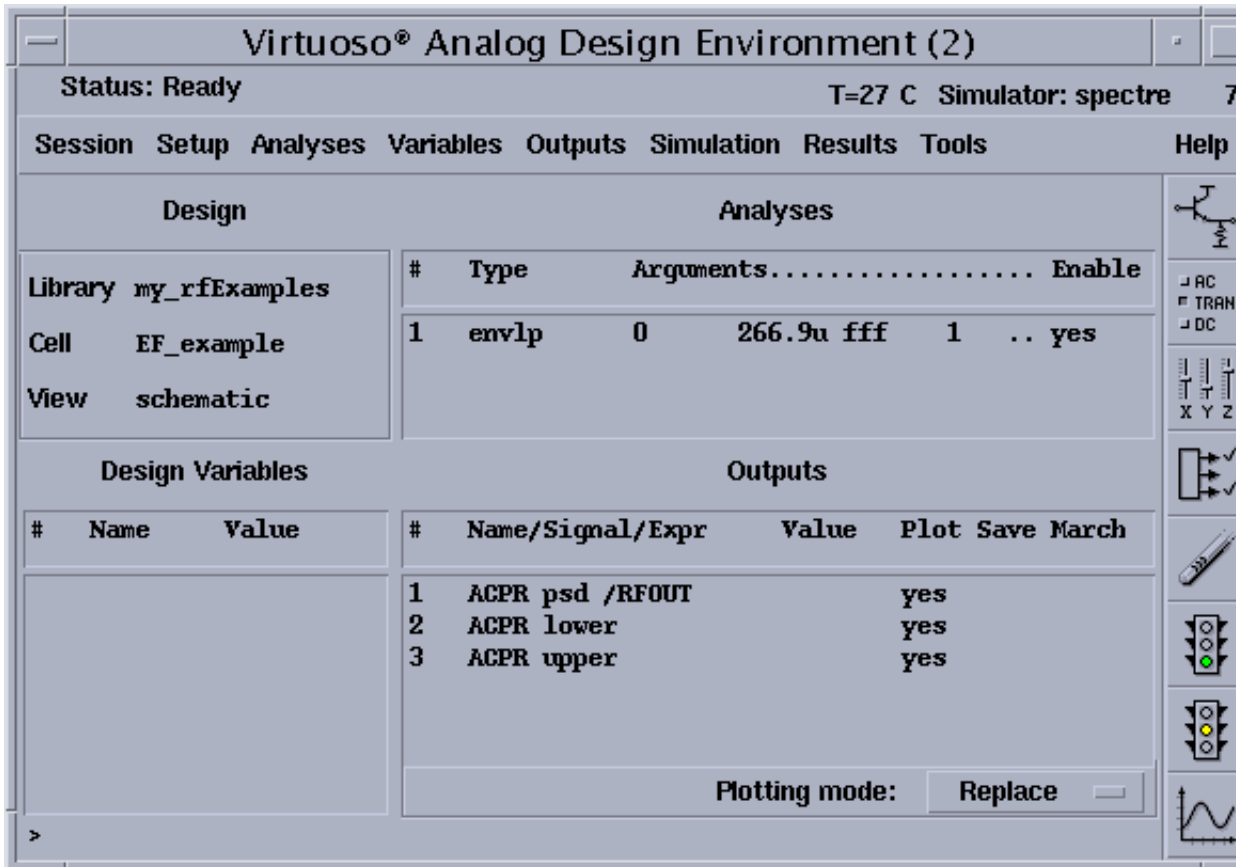
- save**:  selected  lvlpub  lvl  allpub  all
- nestlvl**:
- compression**:  yes  no
- outputtype**:  both  envelope  spectrum
- strobeperiod**:

10. Click *OK* in the Envelope Following Options form.
11. Verify that *Enabled* is highlighted and click *Apply* in the *envlp* Choosing Analyses form.
12. Click *OK* in the *envlp* Choosing Analyses form.
13. The Choosing Analyses form closes.
14. Click *OK* in the ACPR Wizard.
15. The ACPR Wizard closes.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

16. The Simulation window reflects the *Analysis* and *Outputs* information from the ACPR wizard, the *envlp* Choosing Analyses form and the calculations which resulted.



17. In the Simulation window, choose *Simulation – Netlist and Run*.

Look in the CIW for messages saying that the simulation has started and completed successfully. Watch the simulation log file for information as the simulation runs.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

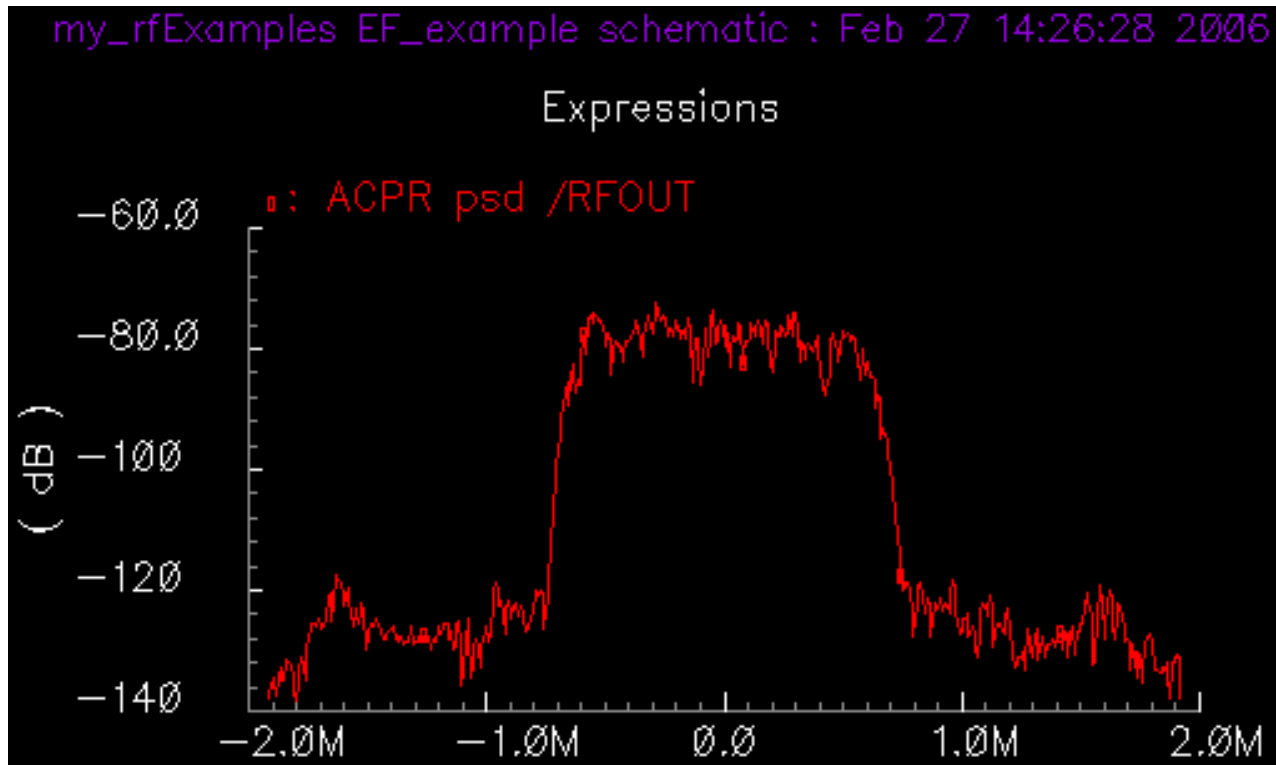
When the simulation successfully completes, an ACPR value for each channel appears in the *Value* column in the Simulation window *Output* section.

Outputs			
#	Name/Signal/Expr	Value	Plot Save March
1	ACPR psd /RFOUT	wave	yes
2	ACPR lower	-61.56	
3	ACPR upper	-60.87	

Plotting mode:

- The *ACPR psd /RFOUT Value* is a *wave* which is plotted.
- The *ACPR lower* channel value is -61.56.
- The *ACPR upper* channel value is -60.07.
- When the simulation finishes, the PSD plot displays as in Figure [7-1](#).

Figure 7-1 PSD Plot Generated by the ACPR Wizard



### Estimating PSD From the Direct Plot Form

The power spectral density (PSD) is always estimated because the information riding on the carrier is a stochastic process and the Fourier transform of a stochastic process is ill defined. No matter how you chose to define the spectral nature of a stochastic process, it always involves an averaging process. Any empirically derived average is an estimate because you can never take an infinite number of samples.

PSD is a frequency-by-frequency average of a set of discrete Fourier transforms (DFTs) of the baseband signal. Here, the baseband signal is the harmonic-time result of an *envlp* analysis.

This example shows how to estimate PSD given the results of the *envlp* analysis you just performed.

In the Simulation window, choose *Results – Direct Plot – Main Form*.

The Direct Plot form appears.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

1. In the Top of the Direct Plot form, do the following:
  - a. Choose *New Win for Plotting Mode*. This plots PSD in a new Waveform window.
  - b. Highlight *envlp* for *analysis*.
  - c. Highlight *Voltage* for *Function*.
  - d. Highlight *spectrum* for *Sweep*.
  - e. The *Power Spectral Density Parameters* section opens at the bottom of the form.
  - f. Highlight *Magnitude* for *Modifier*.
  - g. Choose *1* for *Harmonic Number*.

The top of the Direct Plot form looks like the following.

The image shows a software interface for configuring a plot. It includes several sections with radio buttons and dropdown menus:

- Plotting Mode:** A dropdown menu set to "New Win".
- Analysis:** A radio button labeled "envlp" is selected.
- Function:** Radio buttons for "Voltage" (selected), "Current", and "Power".
- Description:** The text "Description: Harmonic Voltage Spectrum" is displayed.
- Select:** A dropdown menu set to "Net".
- Sweep:** Radio buttons for "spectrum" (selected), "harmonic time", and "time".
- Modifier:** Radio buttons for "Magnitude" (selected), "dB10", and "dBm".
- Harmonic Number:** A list box containing "0" and "1", with "1" selected and highlighted.

2. In the Bottom of the Direct Plot form, enter information for the *Power Spectral Density Parameters*



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- h. Press *Get From Data* to calculate the values for the *Time Interval* values, *From* is 0.0 and *To* is 0.0002669271.
- i. Type 5M for *Nyquist half-bandwidth*.
- j. Type .1M for *Frequency bin width*.
- k. Type 3M for *Max. plotting frequency*.
- l. Type -3M for *Min. plotting frequency*.
- m. In the *Windowing* cyclic field, select *Cosine4*.
- n. In the *Detrending* cyclic field, select *None*.
- o. The *Power Spectral Density Parameters* area at the bottom of the *envlp* Direct Plot form is as follows.

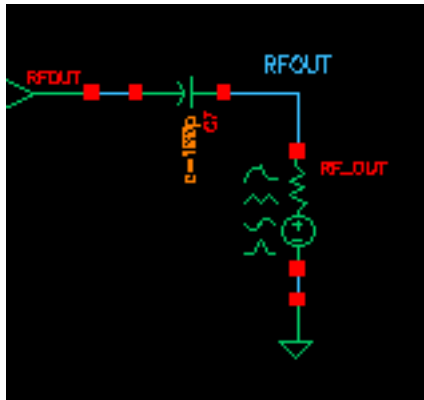
The screenshot shows a dialog box titled "Power Spectral Density Parameters". It contains several input fields and buttons. The "Time Interval" section has "From" set to 0.0 and "To" set to 2669271, with a "Get From Data" button. The "Nyquist half-bandwidth" is set to 5M, "Frequency bin width" to .1M, "Max. plotting frequency" to 3M, and "Min. plotting frequency" to -3M. The "Windowing" dropdown is set to "Cosine4" and the "Detrending" dropdown is set to "None". At the bottom, there is an "Add To Outputs" checkbox (unchecked) and a "Replot" button. A footer bar contains the text "> Select Net on schematic...".

Parameter	Value
From	0.0
To	2669271
Nyquist half-bandwidth	5M
Frequency bin width	.1M
Max. plotting frequency	3M
Min. plotting frequency	-3M
Windowing	Cosine4
Detrending	None

p. Following the prompt at the bottom of the form,

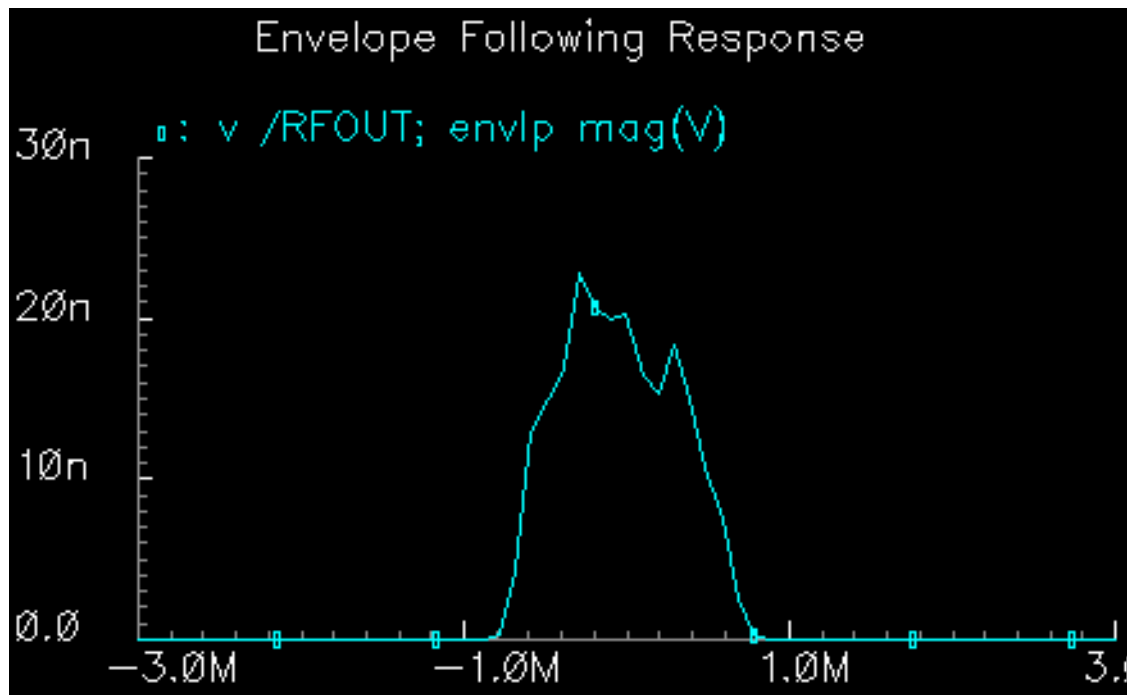
> Select Net on schematic.

Click the transmitter output net (*/RFOUT*).



The PSD plot displays.

Figure 7-2 PSD mag (V) from the envlp Direct Plot Form



To plot the modified PSD plot,

1. In the Direct Plot form, make the following changes:

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- a. Choose *New Win* for *Plotting Mode*.
- b. Highlight *dB10* for *Modifier*.

The top of the Direct Plot form looks like this.

**Plotting Mode**

**Analysis**

envlp

**Function**

Voltage  Current  
 Power

**Description: Harmonic Voltage Spectrum**

**Select**

**Sweep**

spectrum  harmonic time  time

**Modifier**

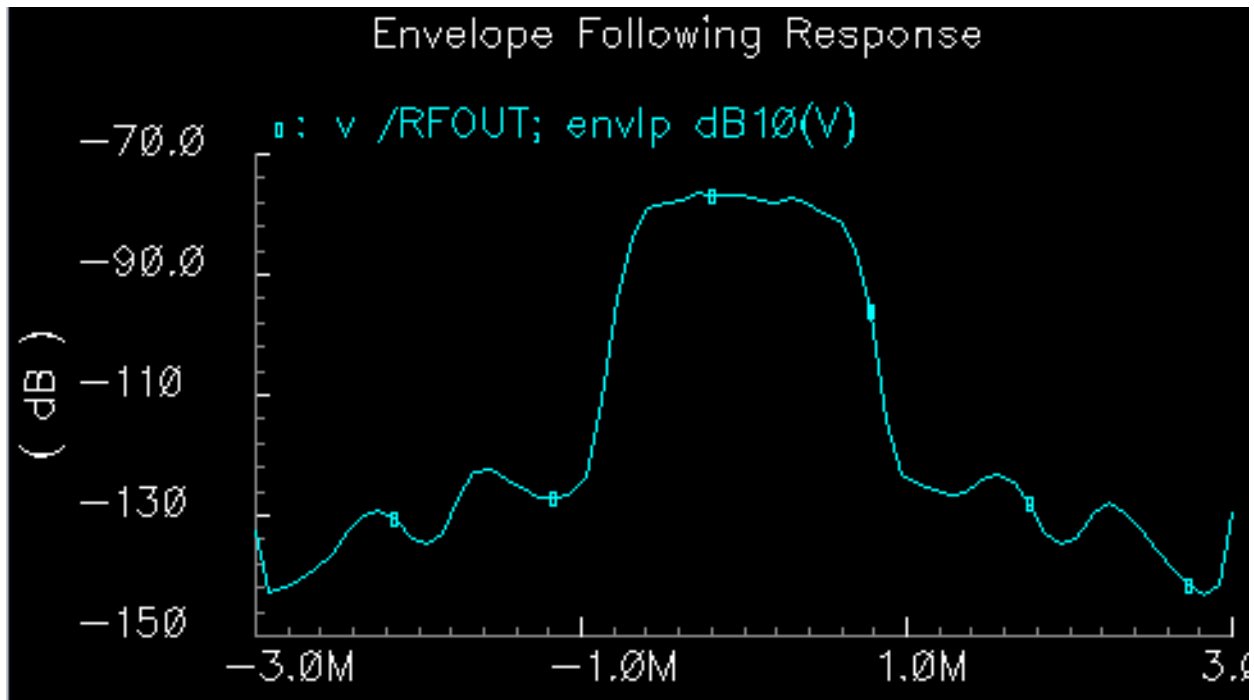
Magnitude  dB10  dBm

**Harmonic Number**

0  
1

- c. Select the RFOUT net in the Schematic window.
- d. The modified psd plot displays, as shown in Figure 7-3.

**Figure 7-3 Estimated PSD from the envlp Direct Plot Form**



Compare Figure 7-3 with the PSD plot shown in Figure 7-2 on page 426.

## Reference Information for ACPR and PSD Calculations

The process outlined in this section is complex largely because the *envlp* analysis parameters are not directly related to the Waveform Calculator *psdbb* function arguments.

The *envlp* parameters include nyquist half-bandwidth, frequency bin width and time interval.

The *psdbb* function arguments are the total number of samples, the window size and the bin-width. All three parameters are in terms of the number of DFT time samples.

However, to make optimum use of the simulation data, it is necessary that the simulation parameters and the *psdbb* function arguments be compatible.

## The Power Spectral Density (PSD) Parameters

When you select *spectrum* for *Sweep* in the Direct Plot form, the *Power Spectral Density Parameters* section opens at the bottom of the Direct Plot form. The *Power Spectral Density Parameters* section displays the analysis parameters that control the PSD estimate.

The *Power Spectral Density Parameters* section is shown in Figure 7-3.

### PSD Parameters on the envlp Direct Plot Form

The screenshot shows a dialog box titled "Power Spectral Density Parameters". It contains several input fields and buttons. The "Time Interval" section has "From" and "To" fields with values "0.0" and "2669271" respectively, and a "Get From Data" button. Below this are four frequency-related fields: "Nyquist half-bandwidth" (5M), "Frequency bin width" (.1M), "Max. plotting frequency" (3M), and "Min. plotting frequency" (-3M). There are also dropdown menus for "Windowing" (set to "Cosine4") and "Detrending" (set to "None"). At the bottom, there is an "Add To Outputs" checkbox (unchecked) and a "Replot" button. A status bar at the very bottom contains the text "> Select Net on schematic...".

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The PSD Parameters are described in Table 7-1.

**Table 7-1 Power Spectral Density Parameters From the Direct Plot Form**

PSD Parameter	Description
<i>Time Interval</i>	<p>The <i>Time Interval</i> parameters specify the time record to analyze in the frequency domain. The time interval should be long enough to support the required frequency resolution.</p> <p>Use <i>Get From Data</i> to calculate the <i>From</i> and <i>To</i> values from the ACPR wizard data.</p>
<i>Nyquist half-bandwidth</i>	<p>The <i>Nyquist half-bandwidth</i> parameter is half the sampling frequency used in the DFTs. Make the nyquist half-bandwidth large enough to prevent aliasing. The true spectrum of the baseband signal should be negligible at this frequency and beyond it.</p>
<i>Frequency bin width</i>	<p>The <i>Frequency bin width</i> parameter specifies the required frequency resolution.</p> <p>When this value is too small; the resulting PSD looks noisy and the PSD has a jagged appearance.</p> <p>When this value is too large; the resulting PSD might be softened when the spectrum should have sharp edges.</p> <p><i>Windowing Function</i> presets include <i>Blackman</i>, <i>Cosine2</i>, <i>Cosine4</i>, <i>ExtCosBell</i>, <i>HalfCycleSine</i>, <i>HalfCycleSine3</i>, <i>HalfCycleSine6</i>, <i>Hamming</i>, <i>Hanning</i>, <i>Kaiser</i>, <i>Parzen</i>, <i>Rectangular</i> and <i>Triangular</i>.</p>
<i>Windowing</i>	<p>The <i>Windowing</i> parameter specifies which windowing function to apply before performing the DFTs.</p>
<i>Detrending</i>	<p>The <i>Detrending</i> parameter has one of three values: <i>None</i>, <i>Mean</i>, or <i>Linear</i>.</p>

If necessary, the waveform is first interpolated to generate evenly spaced data points in time. The data point spacing is the inverse of the DFT sampling frequency. The PSD is computed by

- Breaking the time interval up into overlapping segments
- Multiplying each segment, time point by time point, by the specified *Windowing* function.

Windowing reduces errors caused by a finite time record. It is impossible to work with an infinite time record. Direct use of an unwindowed finite time record is equivalent to multiplying the infinite record by a rectangular pulse that lasts as long as the data record. Multiplication in the time domain corresponds to convolution in the frequency domain. The Fourier transform of a rectangular pulse is a *sinc* function. Considering the frequency domain convolution, the side lobes of the sinc function cause parts of the true spectrum to leak into the frequency of interest, that is, the frequency of the main lobe. Ideally, the sinc function would be a Dirac delta function but that requires an infinite time record. Good window functions have smaller side lobes than the sinc function.

The DFT is performed on each windowed segment of the baseband waveform. At each frequency, the DFTs from all segments are averaged together. Fewer segments means fewer data points in the average at a particular frequency. The length of each segment is inversely proportional to the *Frequency bin width*, which is why a small *Frequency bin width* produces a jagged PSD. A smaller *Frequency bin width* means a longer time segment.

Fewer long segments fit into the given time interval so there are fewer DFTs to average together. In the extreme, there is only one segment and no averaging. Without averaging, the PSD is the square of the magnitude of the DFT of a stochastic process. At the other extreme, large *Frequency bin widths* produce lots of points to average at each frequency but there are fewer frequencies at which to average because fewer large bins fit into the Nyquist frequency. The PSD is smoother but it does not have as much resolution.

PSD is always estimated because the information riding on the carrier is a stochastic process and the Fourier transform of a stochastic process is ill defined. No matter how you chose to define the spectral nature of a stochastic process, it always involves an averaging process. Any empirically derived average is an estimate because you can never take an infinite number of samples.

The Waveform Calculator *psdbb* function performs a discrete Fourier transform (DFT) on the voltage curve which produces the power spectral density (PSD) curve. Integrate the PSD curve to calculate power in the channel.

### **The Waveform Calculator *psdbb* Function**

The Waveform Calculator *psdbb* function, which estimates PSD, derives the function parameters it requires from the PSD parameters and values you supply in the *envlp* Direct Plot form.

*Nyquist half-bandwidth*

- *Frequency bin width*
- *Time Interval*

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

See Table 7-1 for more information about the PSD parameters on the Direct Plot form.

The Waveform Calculator *psdbb* function parameters are

- The total number of samples
- The window size
- The bin-width

All three parameters are in terms of the number of DFT time samples.

The calculations defined in Table 7-2 generate the *psdbb* parameters. The calculated *psdbb* parameters are printed in the CIW window.

**Table 7-2 Parameter Values Calculated by the *psdbb* Calculator Function**

<b>psdbb Calculation</b>	<b>Source of the Data</b>
$L = T_o - From$	<i>T<sub>o</sub></i> and <i>From</i> are the values from the <i>Time Interval</i> <i>T<sub>o</sub></i> and <i>From</i> fields on the <i>envlp</i> Direct Plot form.
$f_{max}$	<i>Nyquist half-bandwidth</i> and <i>Frequency bin width</i> are values from these fields on the <i>envlp</i> Direct Plot form.
$\#bins = \text{floor}(L * binwidth)$	$\#bins \geq 1$ . Here, <i>floor</i> means to take the integer part of, i.e. truncate to the nearest integer.
$2^m * (\#bins) > 2 * L * f_{max}$	Compute the smallest <i>m</i> .
$window\ size = 2^m$	
$\#bins * window\ size$	Compute the number of samples.

You might want to use the *psdbb* function directly when strobing time-harmonic results to eliminate interpolation error. Use of the *psdbb* function is described in the waveform calculator documentation.

### Calculating ACPR

This section describes how to Calculate the ACPR (Adjacent Channel Power Ratio) by hand. The ACPR Wizard performs these calculations for you.

ACPR is measured with respect to any  $x_1$  and  $x_2$ , as  $y_1 - y_2$ . You can calculate the ACPR for any two x-axis values by subtracting their associated y-axis values.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

For example, given the following two spectral parameters,

- The adjacent channel is 2.5 MHz from the carrier
- You choose to define ACPR in terms of power at just two frequencies,
  - 2.5 MHz
  - 0 MHz

Use the cursor to determine the RF output power at 2.5 MHz and 0 MHz frequencies. As you slide the cursor along the curve, you read the X and Y values off the top of the Waveform window:

X: 2.5M      Y: -115.8      v /RFOUT;envlp mag(V)

and

X: 0      Y: -77.33      v /RFOUT;envlp mag(V)

So

- The power at 2.5 MHz equals -115.8 dB
- The power at 0 MHz equals -77.33 dB

Remember that the horizontal scale is frequency offset from the carrier fundamental. The difference between these measurements equals -38.47 dB.

$$ACPR = -115.8 - (-77.33) = -38.47$$

For this definition of ACPR, ACPR = -38.47 dB.

Other definitions of ACPR are possible. You can compute them by adjusting the spectral parameters and applying the waveform calculator to the spectral plot.

### Calculating PSD

For a constant baseband signal, calculate power spectral density as shown in Table 7-3.

**Table 7-3 Power Spectral Density for a Constant Baseband Signal**

---

Baseband signal

$$1 + j \times q$$

where

$$j = \sqrt{-1}$$

**Table 7-3 Power Spectral Density for a Constant Baseband Signal**

Passband signal	$i \times \cos(w \times t) - q \times \sin(w \times t)$
Baseband signal power	$i \times i + q \times q$ <i>volts</i> $\times$ <i>volts</i>
Passband signal power	$i \times \frac{i}{2} + q \times \frac{q}{2} = (1/2) \times (\text{baseband} \text{ power})$

Envelope Spectral analysis computes these values per Hz. You can also express these values as

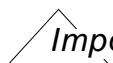
$$(\text{rms} \text{ passband volts}) \times (\text{rms} \text{ passband volts}) / (\text{Hz})$$

Envelope Time-Harmonic analysis computes peak volts because they can be directly compared with the modulating signals.

For this example

- You see a waveform displayed in the Waveform window as  $V^2 / (\text{Hz})$  versus frequency.
- You can think of this as

$$(\text{rms} \text{ passband volts}) \times (\text{rms} \text{ passband volts}) / (\text{Hz})$$

 **Important**

The displayed spectrum is the estimated PSD of the complex envelope divided by two. The division by two is included because the envelope is expressed in units of peak carrier volts, but power in the carrier equals the square of the peak divided by two. It is convenient to express the envelope in peak units because you can then directly compare it against an input baseband signal.

**PSD and the Transmitted Baseband Signal**

When you measure ACPR, it is crucial that you drive the transmitter with the proper baseband signals. The baseband signals driving the transmitter dominate the transmitted PSD. In most cases, the baseband signals are produced by digital filters so the digital filters constrain the

spectrum of the input baseband signal. Distortion in the transmitter causes the spectrum to grow where it should not, hence the need for an ACPR measurement.

It is not practical to model digital filters in Spectre RF because Spectre RF cannot simulate state variables inside Verilog<sup>®</sup>-A modules. Consequently, for now, you must pre-compute and store the baseband inputs and then read them into the Spectre RF analysis through *ppwlf* sources found in the *analogLib*, as shown in “Computing the Spectrum at the Adder Output” on page 435. The *ppwlf* sources also read SPW format, so you can also generate and record the input baseband waveforms using SPW.

The *rfLib* contains three sets of stored baseband waveforms, *cdma*, *dqpsk*, and *gsm*. These waveforms were created with the baseband signal generators in the *measurement* category of the *rfLib*.

If you want to measure ACPR with the noise floor much more than 40 dB below the peak of the output power spectral density, you must create baseband drive signals with a noise floor at or below the required noise floor. If you use a DSP tool such as SPW to create the signals, the filters in the baseband signal generator must operate perhaps hundreds of times faster than those in the actual generator.

Sometimes an ACPR specification exceeds the ACPR of the baseband drive signals. To see if the transmitter meets specifications in that event, it must be driven with an unrealistic baseband signal. Otherwise the signals do not have enough resolution. The noise floor depends heavily on interpolation error.

The Fourier analysis used to compute the power spectral density uses evenly spaced time points. If data does not exist at one of the Fourier time points, the Fourier algorithm must interpolate to create the missing Fourier time point.

You can strobe the harmonic time results to eliminate interpolation of the output but you cannot eliminate interpolation of the baseband drive signals. The only way you can reduce interpolation errors at the input is to use ultra-high-resolution drive signals so that no matter where the interpolation occurs, the error is small. For now, you must generate the ultra-high-resolution drive signals yourself.

### **Computing the Spectrum at the Adder Output**

The following results were obtained by

1. Generating high resolution baseband signals in SPW
2. Storing the high resolution baseband signals
3. Reading the high resolution baseband signals into an *envlp* analysis through the *pwlf* sources

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The SPW FIR filters operate at 300 times the chip rate.

Figure [7-3](#) shows the spectrum at the output of the adder. In this circuit, the RF signal is undistorted at the adder output. The spectrum was computed three times. Each time with a different value for *number of samples*.

The envlp analysis was run with the following parameters:

*reltol*             $1e^{-5}$   
*strobeperiod*    100n

**Note:** If the *strobeperiod* does not equal an integer number of clock cycles, it is internally truncated to an integer number of clock cycles.

For the `psdbb` function

*Time interval*    100u to 1m  
*Window size*     1024  
*Windowing*        *Hanning*

**Note:** *Window size* must be a power of 2. If you enter a value that is not a power of 2, the `psdbb` function truncates the value to the nearest power of 2.

With *strobeperiod* set to 100 ns (100 clock cycles), the simulation produced 9000 evenly spaced samples inside the time interval. The spectra were calculated using 8500, 8750, 9000, 9250, and 9500 points for the *number of samples*. Figure [7-3](#) clearly shows interpolation errors when the number of samples used in the `psdbb` function does not equal the actual number of samples.

Adder Output Spectrum Computed with Three Different Numbers of Samples

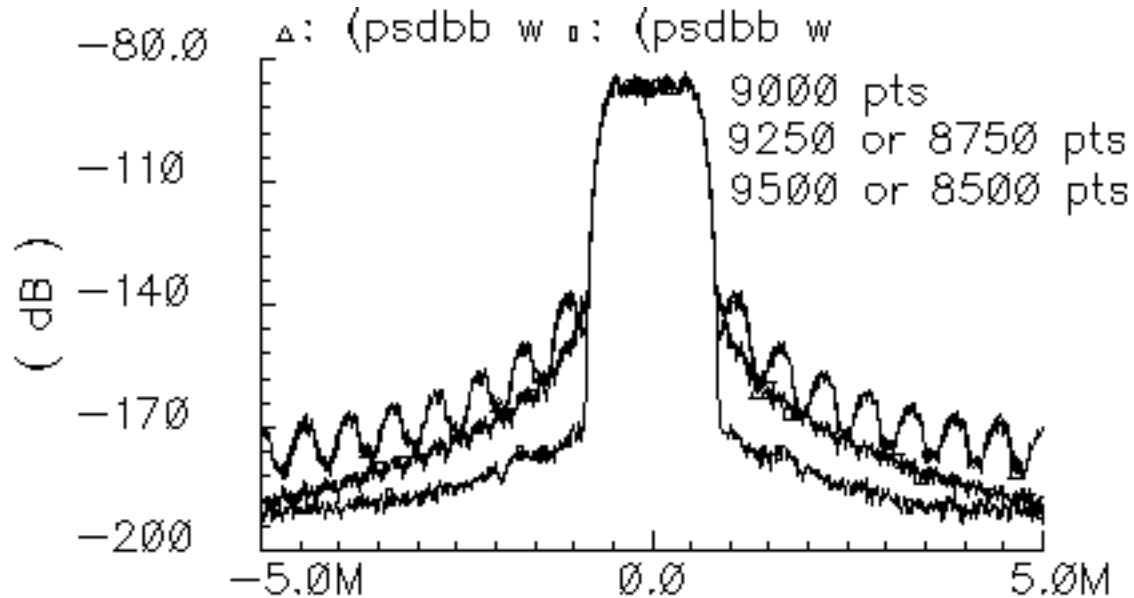
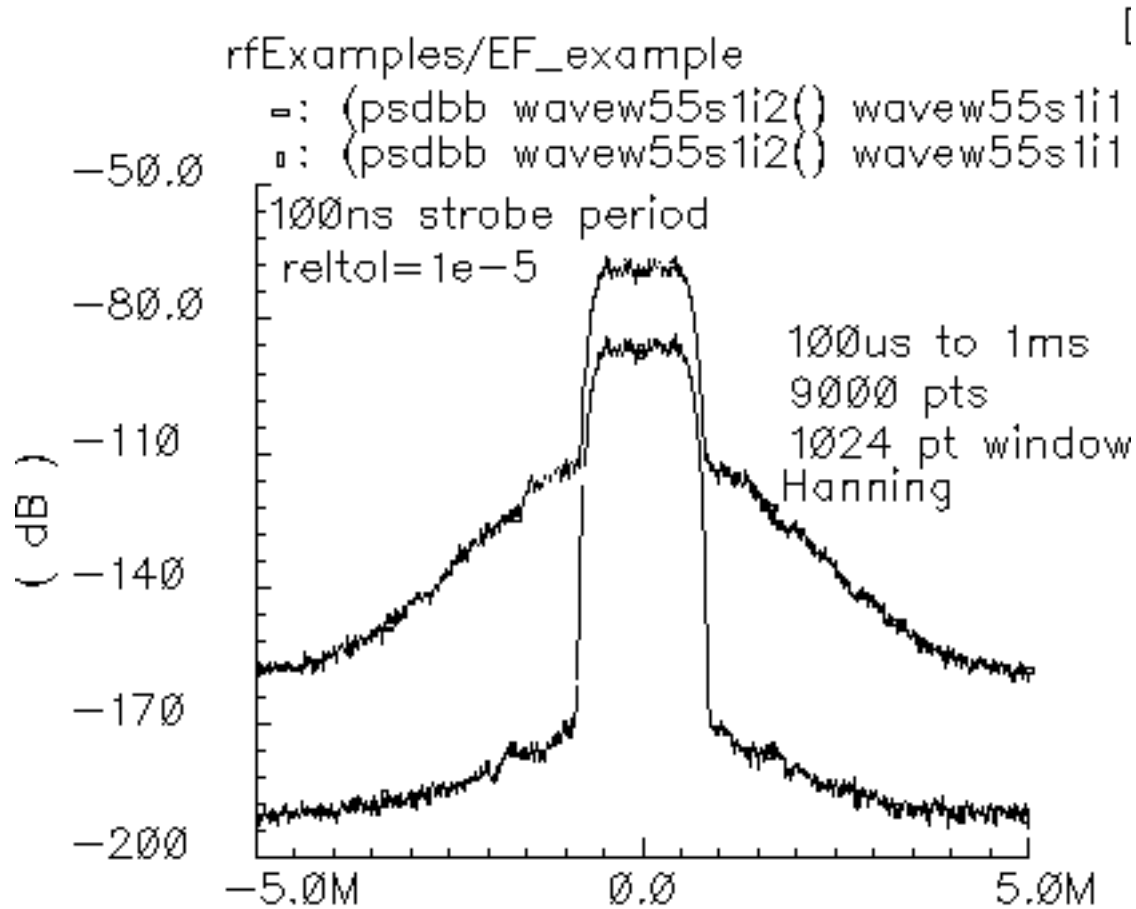


Figure 7-4 compares the input and output PSD plots using the high resolution drive signals and the following set of `psd bb` parameters:

<i>From</i>	100u
<i>To</i>	1m
Number of Samples	9000
Window Size	1024
Window Type	Hanning

In Figure 7-4, the upper PSD plot is the power amplifier output and clearly shows spectral regrowth when compared to the input PSD plot.

Figure 7-4 Comparison of Input and Output PSD plots



## Measuring Load-Pull Contours and Load Reflection Coefficients

This section describes how to generate load pull contours and how to determine whether you must redesign the input matching network for the optimal load.

A *load-pull contour* is a set of points on a Smith chart representing all the loads that dissipate a given amount of power. Load-pull contours help you match a load to a power amplifier for maximum power transfer. Just as a topographical map shows a hiker where the mountain peak lies and how steep the climb is, load-pull contours show which load dissipates the most power and how sensitive that power is to small load perturbations.

To properly use Load-Pull results, you must understand how the Load-Pull feature defines load reflection coefficients. Load reflection coefficients are computed using the PSS analysis.

The *load reflection coefficients* are computed from the load impedance, which is computed as the ratio of the voltage across the load to the current flowing into the load at the RF carrier frequency. Suppose the RF carrier is 1 GHz and the load waveforms are distorted. The impedance is not computed from small-signal perturbations about an operating point. Rather, the impedance is computed as the ratio of the 1 GHz components of the load voltage and current waveforms simulated by a PSS analysis. Because the load is passive and linear, the load reflection coefficient computed in this manner equals the small-signal, or incrementally computed, load impedance.

This is not necessarily true for the *input reflection coefficient* because the input circuitry can be non-linear and it can also contain input offset voltages and currents. However, because matching networks are usually designed only for the RF fundamental, defining the reflection coefficient as the ratio of fundamental Fourier components of the large signals is often justified.

### Creating and Setting Up the Modified Circuit (EF\_LoadPull)

This example tells you how to

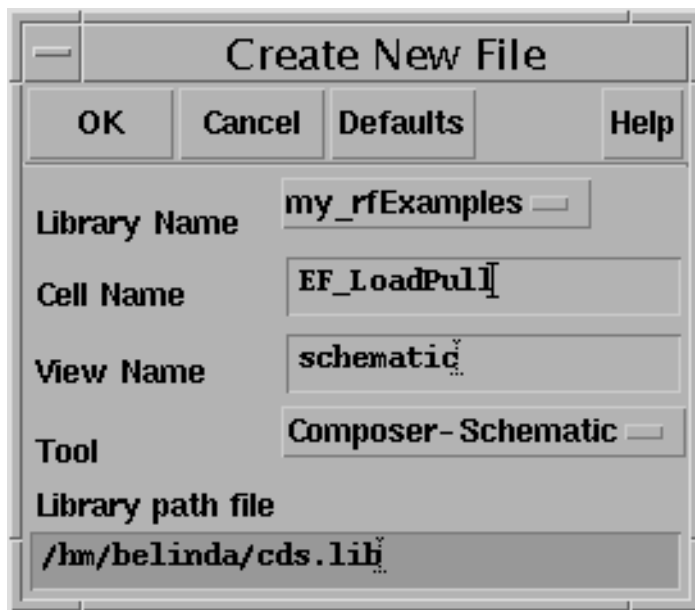
- Create the *EF\_LoadPull* schematic for this example. The *EF\_LoadPull* schematic is a modified copy of the *EF\_example* schematic from the *rfExamples* library.
- Set up and run the necessary PSS and Parametric analyses
- Measure load-pull contours for the modified *EF\_LoadPull* schematic
- Measure load reflection coefficients for the modified *EF\_LoadPull* schematic

Before you start, make sure you have performed the setup procedures for the writable *rfExamples* library, as described in [Chapter 3](#).

### Creating a New Empty Schematic Window

1. In the CIW choose *File—New—Cellview*.

The Create New File form appears.



2. In the Create New File form, do the following:

- a. Choose *my\_rfExamples* for *Library Name*.

Select *my\_rfExamples*, the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

In the *Cell Name* field, enter *EF\_LoadPull*.

- b. In the *View Name* field, enter *schematic*.
- c. In the *Tool* cyclic field, select *Composer-Schematic*.
- d. Click *OK*.

A new, empty Schematic window named *EF\_LoadPull* opens.



### Opening and Copying the EF\_example Circuit

Now open another schematic window containing the *EF\_example* circuit. Then copy the *EF\_example* circuit into this empty *EF\_LoadPull* Schematic window.

1. In the CIW, choose *File – Open*.

The Open File form appears.

2. In the Open File form, do the following:
  - a. Choose *my\_rfExamples* for *Library Name*.

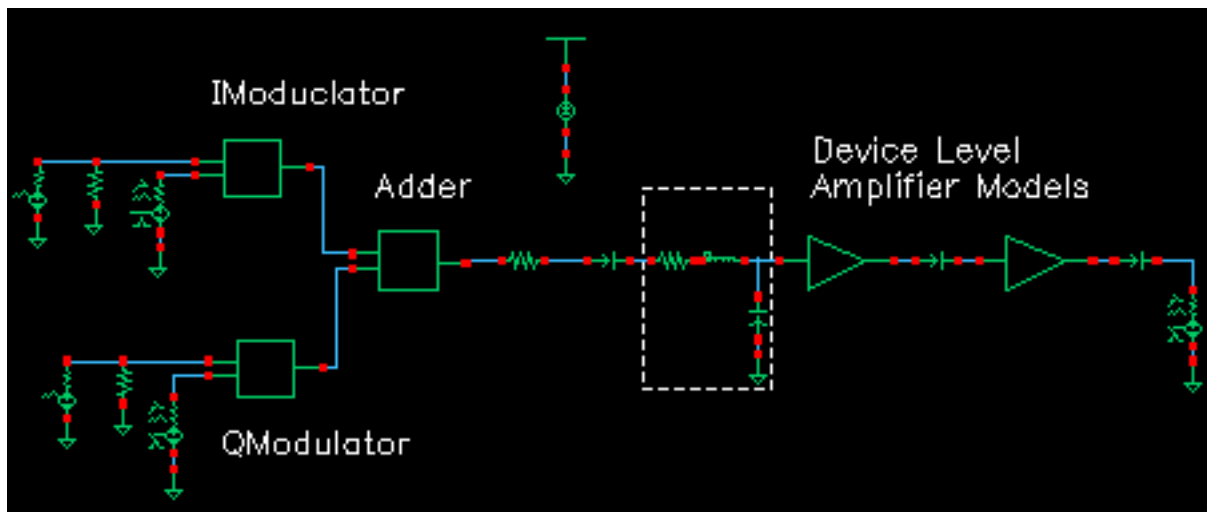
Select the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

In the *Cell Name* list box, highlight *EF\_example*.

- b. Choose *schematic* for *View Name*.
- c. Highlight *edit* for *Mode*.
- d. Click *OK*.

The Schematic window appears with the *EF\_example* schematic as shown in Figure 7-4. This is a simple direct-conversion transmitter with ideal I/Q modulators.

### The EF\_example Schematic



3. In the *EF\_example* Schematic window, choose *Edit - Copy* and follow the prompts at the bottom of the Schematic window.

**4.** Following the prompt,

> point at object to copy

left click and drag to create a box around the entire *EF\_example* circuit.

The *EF\_example* components are highlighted in yellow.

**5.** Following the prompt,

> point at reference point for copy

click inside the outlined elements.

**6.** Following the prompt,

> point at destination point for copy

move the cursor to the empty *EF\_LoadPull* Schematic window and click there. This drags a copy of the *EF\_example* circuit into the empty Schematic window.

The *EF\_LoadPull* Schematic window now contains a copy of the *EF\_example* schematic.

**7.** If necessary, choose *Window - Fit* to center the *EF\_LoadPull* circuit in the Schematic window.

**8.** In the *EF\_example* Schematic window, choose *Window - Close*.

The *EF\_example* Schematic window closes.

### **Opening the Simulation Window for the *EF\_LoadPull* Schematic**

- Open the Simulation window from the *EF\_LoadPull* Schematic window as described for the *EF\_example* schematic in [“Opening the Simulation Window”](#) on page 372.

### **Setting up the Model Libraries for the *EF\_LoadPull* Schematic**

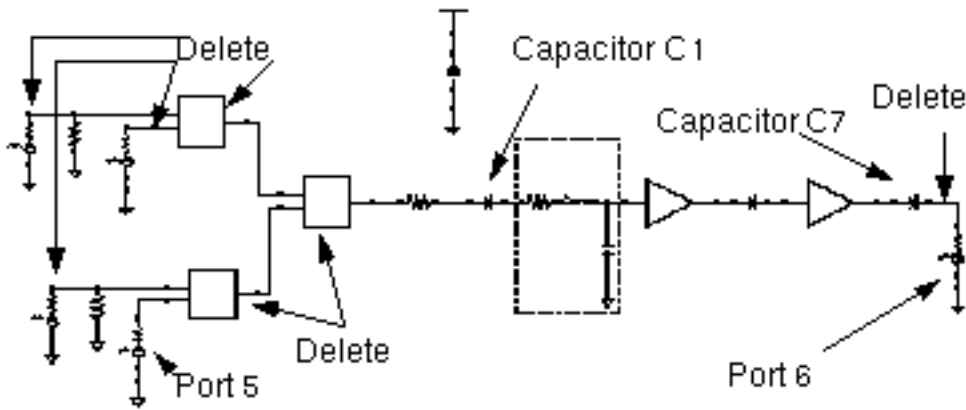
- Set up the Model Libraries for the *EF\_LoadPull* Schematic as described for the *EF\_example* schematic in [“Setting Up the Model Libraries”](#) on page 373.

### **Editing the *EF\_LoadPull* Schematic**

#### ***Delete Components and Wires from the *EF\_LoadPull* Schematic***

In the *EF\_LoadPull* Schematic window, delete components and their associated connecting wires as shown in Figure [7-5](#).

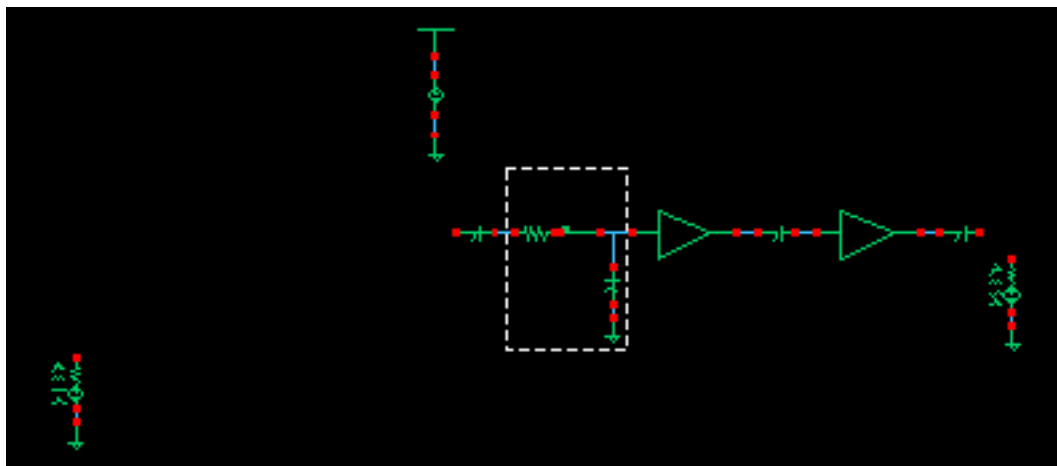
Figure 7-5 Components and Wires to Delete for the EF\_LoadPull Schematic



If you need assistance with methods for editing the schematic, see the *Virtuoso® Schematic Composer™ User Guide*.

1. In the *EF\_LoadPull* Schematic window, choose *Edit – Delete*.
2. Click a component or wire to delete it.
3. Press the *Esc* key when you are done deleting.
4. After deleting the indicated components and wires, the *EF\_LoadPull* Schematic should look like [Figure 7-6](#) on page 443.

Figure 7-6 *EF\_LoadPull* Schematic After Deleting Components



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

5. In the *EF\_LoadPull* Schematic window, choose *Edit – Move* to move both *PORT5* and *PORT6* as shown in [Figure 7-7](#) on page 445.
6. Move *PORT5* closer to capacitor *C1*.
7. Move *PORT6* away from capacitor *C7* to make room for the *PortAdaptor* component you add later.
8. Left click and drag to create a rectangle around *PORT5* and its *GND*.
9. *PORT5* and the *Gnd* are highlighted in yellow.
10. Click inside the yellow line and drag the outlined components to move them close to capacitor *C1* as shown in [Figure 7-7](#) on page 445.
11. Left click and drag to create a rectangle around *PORT6* and its *GND*.
12. *PORT6* and the *Gnd* are highlighted in yellow.
13. Click inside the yellow line and drag the components to move them away from capacitor *C7* as shown in [Figure 7-7](#) on page 445. This makes room for the *PortAdaptor*.
14. Choose *Window - Fit* to center the edited schematic in the window.
15. Place the *PortAdaptor*
  - a. In the *EF\_LoadPull* Schematic window, choose *Add – Instance*.

The Add Instance form appears.

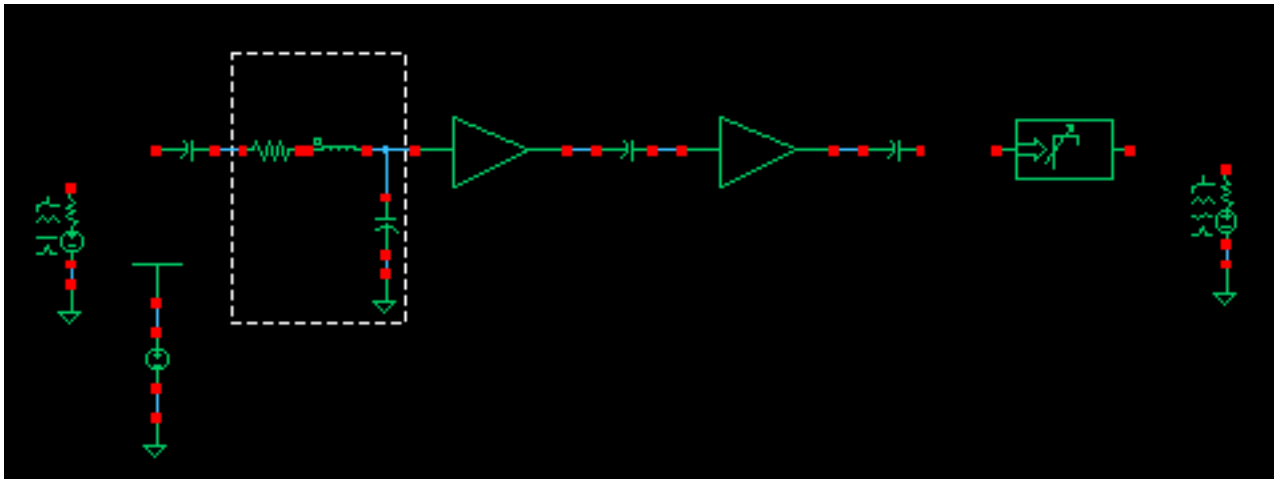
Hide	Cancel	Defaults	Help	
Library	rfExamples	Browse		
Cell	portAdapted			
View	symbol			
Names				
Array	Rows	1	Columns	1
Rotate	Sideways	Upside Down		

- b. In the Add Instance form *Library* field, type `rfExamples`.
- c. In the *Cell* field, type `portAdapter`.
- d. In the *View* field, type `Symbol`.

As you move your cursor from the form to the *EF\_LoadPull* Schematic window, a copy of the *portAdapter* moves with the cursor. Left click to place the *portAdapter* as shown in [Figure 7-7](#) on page 445.

After you place the *portAdapter* in the schematic, press the *Esc* key to remove the *portAdapter* symbol from your cursor and close the Add Instance form.

**Figure 7-7 The EF\_LoadPull Schematic**

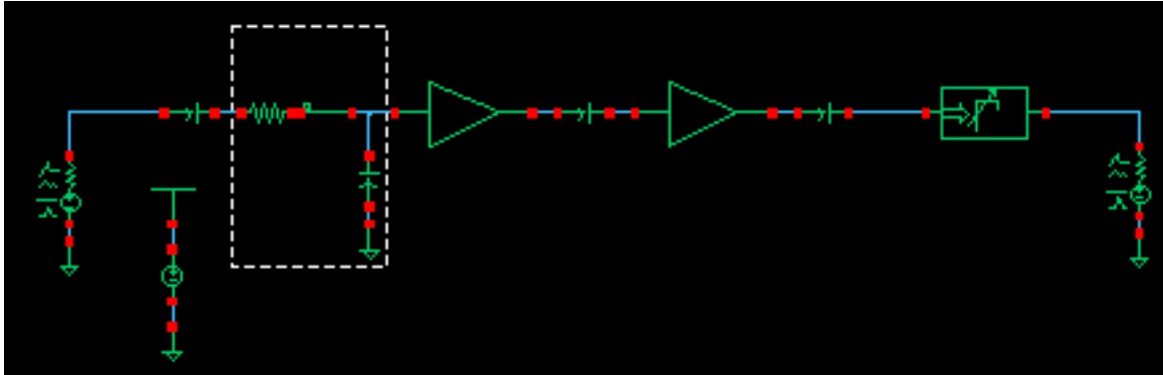


### **Wire the Schematic**

In the *EF\_LoadPull* Schematic window, you need to connect *PORT5* to Capacitor *C1* and connect the *portAdapter* between Capacitor *C7* and *PORT6*. To do so,

1. In the Schematic window, choose *Add - Wire (Narrow)* and do the following.
2. Click the terminal on *PORT5* then click the terminal on *C1*.
3. Click the terminal on *C7* then click the terminal on the *PortAdaptor*.
4. Click the terminal on the *PortAdaptor* then click the terminal on *PORT6*.
5. Press the *Esc* key to stop wiring.
6. The edited schematic looks like the one in [Figure 7-7](#).

### The EF\_LoadPull Schematic



#### **Edit CDF Properties for both the PortAdapter and Port6**

In the *EF\_LoadPull* Schematic, the *portAdapter* and its terminating port, *Port6*, must have the same reference resistance.

1. In the *EF\_LoadPull* Schematic window, select the *PortAdapter*.
2. In the *EF\_LoadPull* Schematic window, choose *Edit – Properties – Objects*.

The Edit Object Properties form appears with information for the *Port Adapter* displayed.

3. In the Edit Object Properties form, do the following and click *Apply*.
  - a. Type *frf* for *Frequency*.
  - b. Type *theta* for *Phase of Gamma (degrees)*.
  - c. Type *mag* for *Mag of Gamma (linear scale)*.
  - d. Type *r0* for *Reference Resistance*.

The completed form looks like this.

CDF Parameter	Value
Frequency	frf
Phase of Gamma (degrees)	theta
Mag of Gamma (linear scale)	mag
Reference Resistance	r0
Gamma Phase Offset (deg)	0
Gamma Mag Offset (linear)	0

- In the *EF\_LoadPull* Schematic window, select *Port6*.

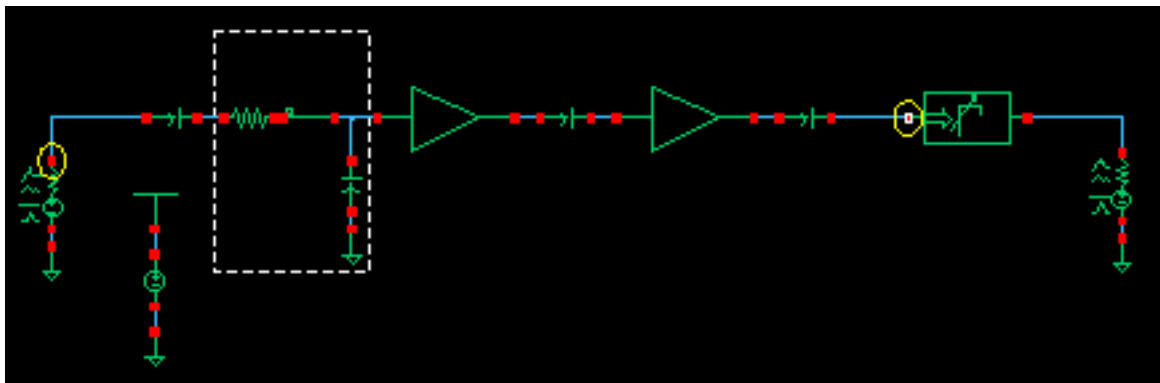
The Edit Object Properties form changes to display data for *Port6*.

In the Edit Object Properties form for *Port6*, type *r0* for *Resistance* and click *OK*.

Select Outputs To Save

- In the Simulation window for the *EF\_LoadPull* Schematic, choose *Outputs – To Be Saved – Select on Schematic*.
- In the *EF\_LoadPull* Schematic window, click each terminals that is circled in Figure 7-8.

**Figure 7-8 Outputs to Save in the *EF\_LoadPull* Schematic**



After you click a terminal, it is circled in the schematic as shown in Figure 7-8. The selected outputs are also displayed in the Simulation window Outputs area as shown in Figure 7-9.

**Figure 7-9 Outputs Area in the Simulation Window**

Outputs					
#	Name/Signal/Expr	Value	Plot	Save	March
1	PORT3/PLUS		no	yes	no
2	I3/out		no	yes	no

### ***Editing Variables***

1. In the Simulation window, choose *Variables – Copy from Cellview*.

The *Design Variables* list box in the Simulation window changes to reflect the copied variables from the schematic as shown in Figure 7-10.

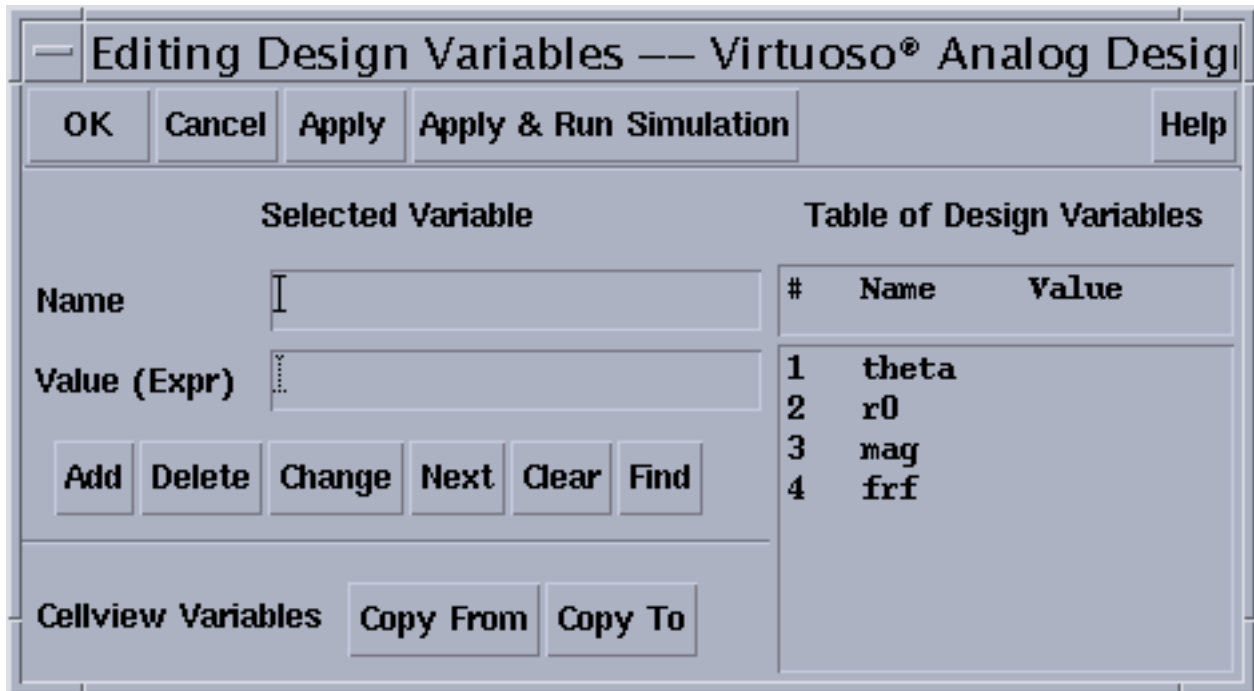
**Figure 7-10 Design Variables Area Showing Variable Names**

Design Variables		
#	Name	Value
1	theta	
2	r0	
3	mag	
4	frf	

2. In the Simulation window, choose *Variables - Edit*.



The Editing Design variables form appears.



3. In the Editing Design Variables form, do the following.
  - a. Highlight one of the variables in the *Table of Design Variables* list box.  
The variable name displays in the *Name* field.
  - b. Type the variable's associated value from the following list into the *Value (Expr)* field.
    - theta 0
    - r0 50
    - mag 0
    - frf 1G
  - c. Click *Change*.
  - d. When you have given values to all four variables, click *OK*.

The Design Variables list box in the Simulation window now displays both variable names and the value associated with each name.

Design Variables		
#	Name	Value
1	theta	0
2	r0	50
3	mag	0
4	frf	1G

### **Save the Changes to the EF\_LoadPull Schematic**

- ▶ In the Schematic window, choose *Design – Check and Save* to save the current state of the *EF\_LoadPull* schematic.

## **Setting Up and Running the PSS and Parametric Analyses**

This example tells you how to

- Set up the swept PSS analysis to sweep the design variable *theta*.
- Set up the Parametric analysis to sweep the design variable *mag*.
- Run the parametric and swept PSS analyses.
- Plot the results.

### **Setting Up the PSS Simulation**

1. In the Simulation window, choose *Analyses – Choose*.  
The Choosing Analyses form appears.
2. In the Choosing Analyses form, highlight *pss*.
3. The form changes to display options for PSS simulation.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

4. In the Choosing Analyses form, do the following
  - a. If information for *frf* is not displayed in the *Fundamental Tones* list box, type `frf` in the *Name* field below the list box, type `1G` in the *Expr* field below the list box and click *Clear/Add*.
  - b. Click *Update From Schematic* to update the values in the *Fundamental Tones* list box from those in the schematic.
  - c. Before you can *Update From Schematic*, you must have performed a *Design - Check and Save* on the design in the Schematic window.
  - d. Click *Auto Calculate* to automatically calculate and enter a value in the *Beat Frequency* field. `1G` displays in the *Beat Frequency* field.
  - e. Choose *Number of harmonics* for *Output harmonics* and type `9` in the adjacent field.

The top of the PSS analysis form looks like this.

**Periodic Steady State Analysis**

Engine       Shooting    Flexible Balance

**Fundamental Tones**

#	Name	Expr	Value	Signal	SrcId
1	fff	1G	1G	Large	PORT3
2	frf	1G	1G	Large	

Beat Frequency       Auto Calculate
   
 Beat Period

**Output harmonics**

Number of harmonics

- f. Highlight *moderate* for *Accuracy Defaults* (*errpreset*).
- g. Highlight the *Sweep* button and choose *Variable* in the associated cyclic field.
- h. Type *theta* for *Variable Name* (or click *Select Design Variable*, highlight *theta* in the form that appears and click *OK*).
- i. Choose *Start – Stop* for *Sweep Range*.
- j. Type 0 for the *Start* value and 359 for the *Stop* value.
- k. Choose *Linear* for *Sweep Type*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- I. Highlight *Number of Steps* and type 20 in the adjacent field.
- m. Verify that *Enabled* is highlighted.

The bottom of the PSS analysis form looks like this.

**Accuracy Defaults (empreset)**  
 conservative  moderate  liberal  
Additional Time for Stabilization (tstab)   
Save Initial Transient Results (saveinit)  no  yes

Oscillator

Sweep  Frequency Variable?  no  yes  
Variable  Variable Name

**Sweep Range**  
 Start-Stop Start  Stop   
 Center-Span

**Sweep Type**  
 Linear  Step Size   
 Logarithmic  Number of Steps

Add Specific Points

Enabled

n. Click *Apply*. Then click *OK*.

### Setting Up the Parametric Analysis and Performing the Analyses

1. In the Simulation window, choose *Tools – Parametric Analysis*.

The Parametric Analysis form appears.

2. In the Parametric Analysis form, do the following:

- a. Type *mag* for *Variable Name*.
- b. Choose *From/To* in the *Range Type* cyclic field.
- c. Type 0 for *From* and 0.95 for *To* in the adjacent fields.
- d. Choose *Linear* in the *Step Control* cyclic field.
- e. Type 10 in the adjacent *Total Steps* field.

The completed form looks like this.

Parametric Analysis – spectre(1): my\_rfExamples test schematic

Running mag=0.1055556 8 rema

Tool Sweep Setup Analysis

Sweep 1 Variable Name mag Add Specification

Range Type From/To From 0 To 0.95

Step Control Linear Total Steps 10

3. In the Parametric Analysis form, choose *Analysis – Start*.

4. Look in the CIW for a message that says the simulation has completed successfully.

### Displaying Load Contours

1. In the Simulation window, choose *Results – Direct Plot – Main Form*.

The Direct Plot form appears.

2. In the Direct Plot form, do the following
  - a. Select *Replace* in the *Plotting Mode* cyclic field.
  - b. Highlight *pss* for *Analysis*.
  - c. Highlight *Power Contours* for *Function*.

The top of the Direct Plot form looks like the following.

The image shows a screenshot of the 'Direct Plot Form' dialog box. At the top, there is a title bar with the text 'Direct Plot Form'. Below the title bar are three buttons: 'OK', 'Cancel', and 'Help'. The main area of the dialog is divided into three sections. The first section is 'Plotting Mode', which has a dropdown menu currently set to 'Replace'. The second section is 'Analysis', which contains a list of radio buttons, with 'pss' selected. The third section is 'Function', which contains a grid of radio buttons. In this section, 'Power Contours' is selected. Other options in the 'Function' section include Voltage, Current, Power, Voltage Gain, Current Gain, Power Gain, Transconductance, Transimpedance, Compression Point, IPN Curves, Harmonic Frequency, Power Added Eff., Power Gain Vs Pout, Comp. Vs Pout, Node Complex Imp., and THD.

3. Highlight *Magnitude* for *Power Modifier*.
4. Leave *Maximum Power* and *Minimum Power* blank.
5. Type 9 for *Number of Contours*, if necessary.
6. Type 50.0 for *Reference Resistance*, if necessary.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

7. Following the prompt at the bottom of the form,

> Select Output Harmonic on this form...

highlight 1 1G for *Output Harmonic*.

You would select a different harmonic if the PSS fundamental frequency were smaller than 1 GHz. For example, if another part of the circuit is driven at 1.5 Hz, the fundamental is 500 MHz. But because the part of the circuit where you want to do load-pull analysis operates at 1 GHz, in order to plot the correct contours, the contours associated with 1 GHz, you must specify the second harmonic of the fundamental,  $2 \times 500 \text{ MHz} = 1 \text{ GHz}$ .

The *Select cyclic field* displays *Single Power/Refl Terminal* and the prompt at the bottom of the form displays

> Select Instance Terminal on Schematic.

The bottom of the Direct Plot form looks like the following.

Select **Single Power/Refl Terminal**

Power Modifier  Magnitude  dB10  dBm

Maximum Power

Minimum Power

Reference Resistance

Number of Contours

Close Contours

Output Harmonic

0	0
<b>1</b>	<b>1G</b>
2	2G
3	3G
4	4G
5	5G

Add To Outputs  Replot

> Select Instance Terminal on schematic...

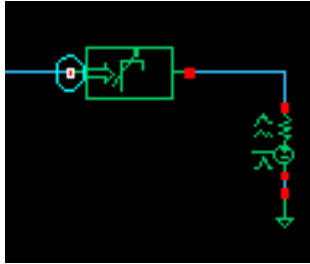


8. Follow the prompt at the bottom of the Direct Plot form,

> Select Instance Terminal on Schematic

by clicking, in the schematic, the terminal at the port adapter as shown in Figure [7-11](#).

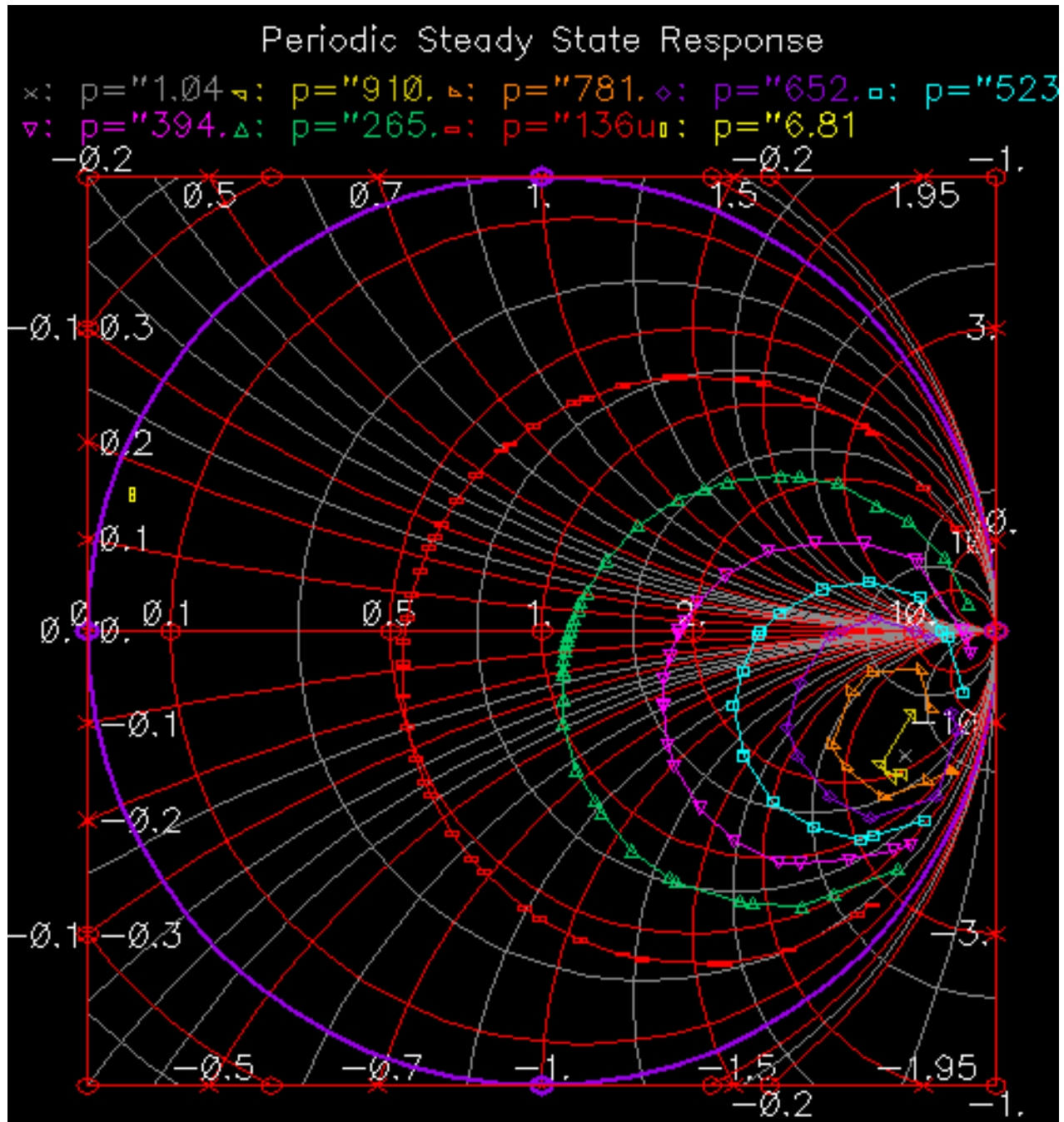
**Figure 7-11 Terminal at Port Adapter**



**Note:** Subsequent instructions call this terminal the *terminal at the port adapter*.

After you select the terminal, the plot for the load contours appears in the Waveform window. Figure [7-11](#) shows the entire plot.

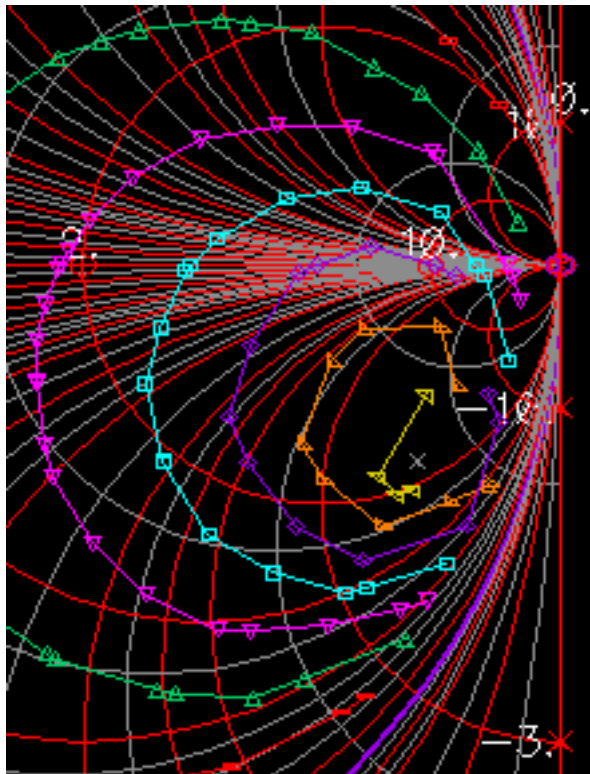
### Complete Load Contour Plot



A small x appears at the maximum power point, which in this case lies near the center of the smallest (yellow) constant power contour. See [Figure 7-12](#) on page 459 for a detailed view.

Figure 7-12, an enlarged area taken from Figure 7-11, which more clearly shows the  $x$  that marks the maximum power point (inside the yellow contour).

**Figure 7-12 Enlarged View Showing Max Power Point (X)**



If you place the cursor on the  $x$ , you can read the following information across the top of the Waveform window.

Real: 2.479      Imag: -4.737      Freq: -360      p="1.04m"; Constant Power Contours

This indicates that a normalized load impedance of about  $2.48 - j4.737$  dissipates the most power. The  $x$  appears at the maximum power point.

### Adding the Reflection Contours to the Plot

You might want to maximize load power subject to a constraint on the magnitude of the amplifier's input reflection coefficient. Such a constraint can prevent unstable interactions with the preceding stage.

You can overlay load-pull contours with contours of constant input reflection coefficient magnitude. The optimal load corresponds to the reflection coefficient that lies on both the

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

*largest power* load-pull contour and on a constant input reflection coefficient contour that is within the constraint. Here *largest power* means the contour corresponding to the largest amount of power delivered to the load.

To use the topographical map analogy again, this is like overlaying constant elevation contours with constant temperature contours. The optimum objective is similar to trying to find the highest point on the mountain where the temperature is above 60 degrees.

In the Direct Plot form, do the following:

1. If necessary, in the Simulation window, choose *Results – Direct Plot – Main Form*.  
The Direct Plot form appears.
2. Select *Append* in the *Plotting Mode* cyclic field.
3. Highlight *pss* for *Analysis*.
4. Highlight *Reflection Contours* for *Function*.

The top of the completed form looks like this.

The image shows a software interface window titled "Direct Plot" with the following settings:

- Plotting Mode:** A dropdown menu set to "Append".
- Analysis:** A radio button selection where "pss" is selected.
- Function:** A list of radio button options where "Reflection Contours" is selected. Other options include Voltage, Current, Power, Voltage Gain, Current Gain, Power Gain, Transconductance, Transimpedance, Compression Point, IPN Curves, Power Contours, Harmonic Frequency, Power Added Eff., Power Gain Vs Pout, Comp. Vs Pout, Node Complex Imp., and THD.

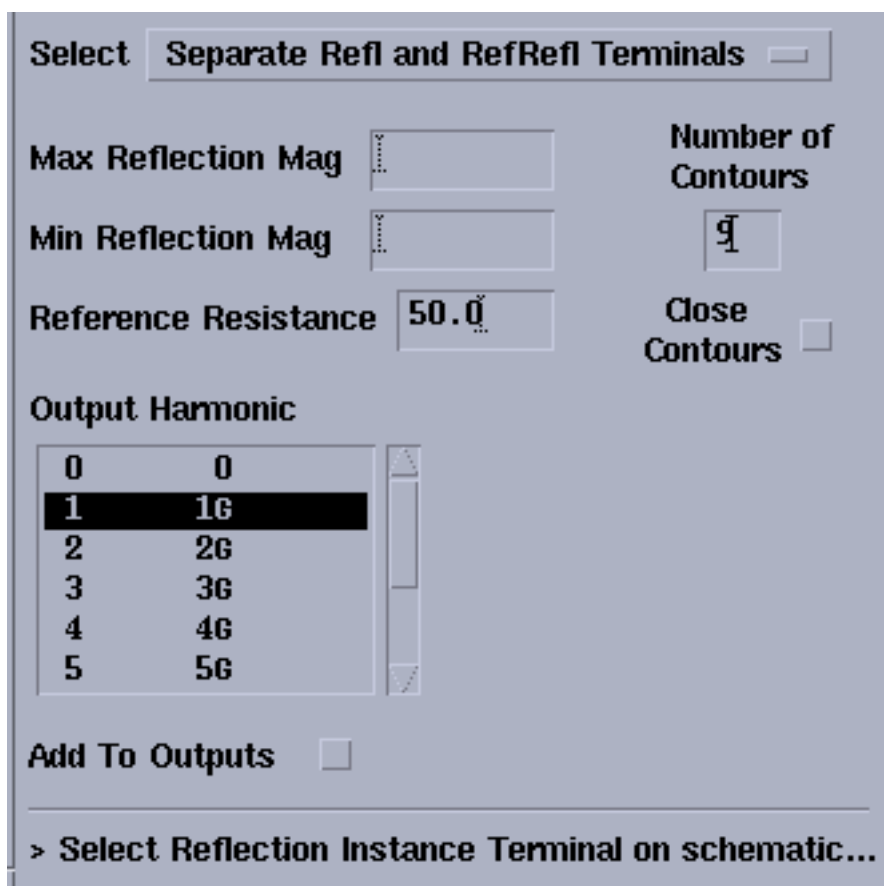
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

5. Type 9 for *Number of Contours*, if necessary.
6. Type 50.0 for *Reference Resistance*, if necessary.
7. Highlight 1 1G for *Output Harmonic*.
8. In the *Select* cyclic field, select *Separate Refl and RefRefl Terminals*. The prompt at the bottom of the form changes to

> Select Reflection Instance Terminal on Schematic

The bottom of the completed form looks like this.



The screenshot shows a dialog box with the following fields and controls:

- Select**: A dropdown menu showing "Separate Refl and RefRefl Terminals".
- Max Reflection Mag**: A text input field.
- Min Reflection Mag**: A text input field.
- Reference Resistance**: A text input field containing "50.0".
- Number of Contours**: A spin box containing the value "9".
- Close Contours**: A checkbox that is currently unchecked.
- Output Harmonic**: A list box with the following items:

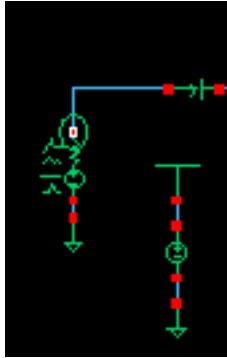
0	0
1	1G
2	2G
3	3G
4	4G
5	5G

The row "1 1G" is highlighted.
- Add To Outputs**: A checkbox that is currently unchecked.
- At the bottom, a prompt reads: "> Select Reflection Instance Terminal on schematic...".

9. Follow the prompt at the bottom of the Direct Plot form,  
> Select Reflection Instance Terminal on Schematic

In the Schematic window, click the terminal at port 5, as shown in Figure 7-13.

Figure 7-13 Terminal at Port 5



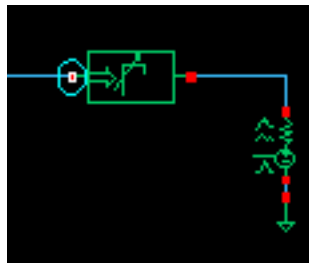
**Note:** Subsequent instructions call this terminal the *terminal at Port 5*.

The prompt on the bottom of the Direct Plot form is

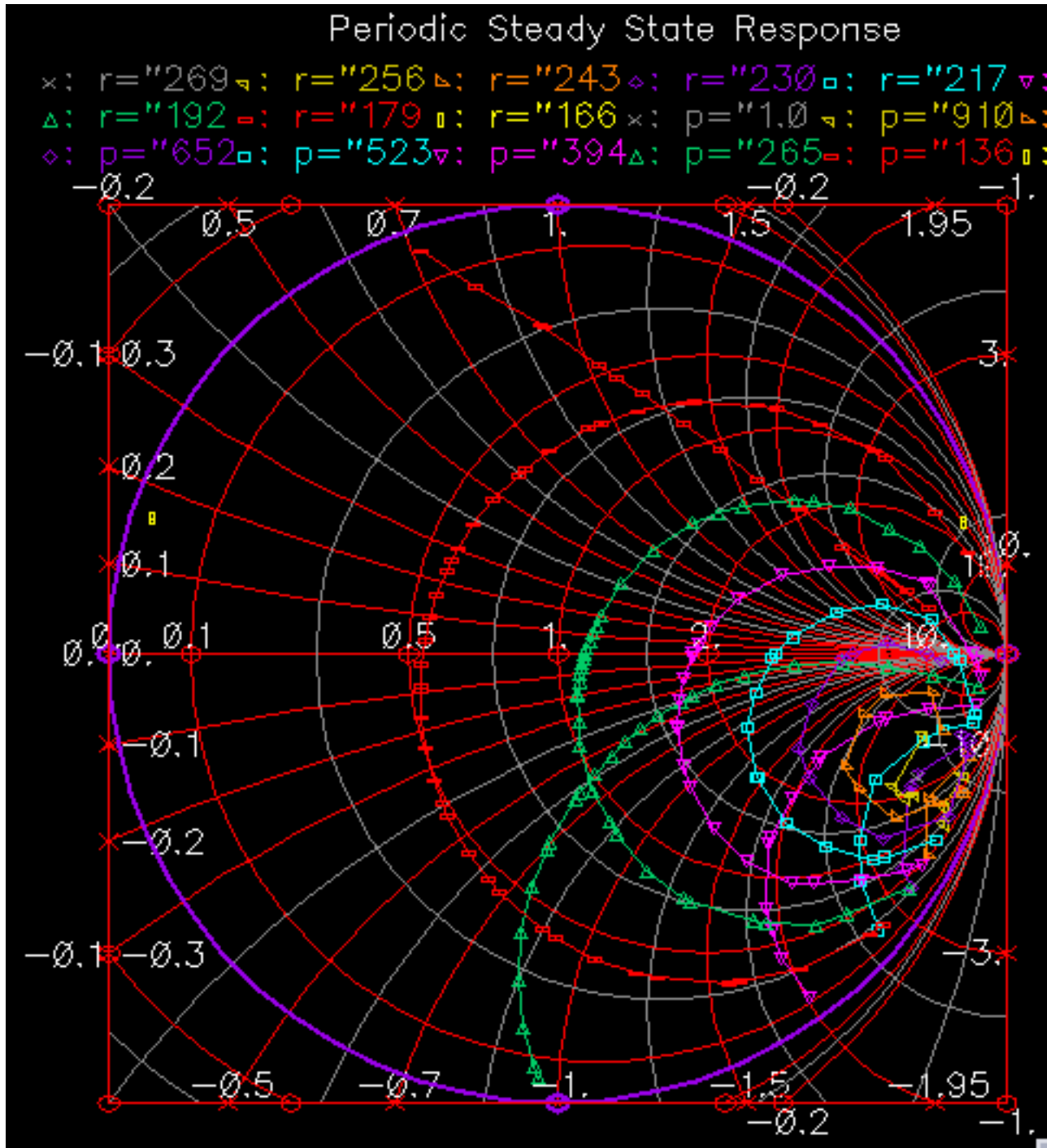
```
> Select Reflection Instance Terminal on Schematic....
```

Follow the prompt. In the Schematic window, click the *terminal at the port adapter* (See [7-14](#)).

Figure 7-14 Terminal at Port Adapter



The plot for reflection contours is added to the Waveform window.



Let's assume that for stability, the input reflection coefficient should be less than or equal to 0.218. For the curves shown above, the optimal normalized load impedance is approximately

3.908 -j5.073. Assuming you were free to draw smaller constant power contours, the true constrained optimal power point would occur when the constant power contour is just tangent to the constant input reflection coefficient contour of 0.218.

In the Direct Plot Form for *Reflection Contours, Separate Refl & RefRefl Terminals* is selected to measure *Reflection Contours* as contours of constant reflection (magnitude) as seen from the power amplifier's input (measured at *port 5*, the input port) with respect to the reflection of the power amplifier's load (measured at the input to the port adapter.)

In this case,

- In response to the first prompt,

> Select Reflection Instance Terminal on Schematic

Select the input to the power amplifier, the terminal of *port 5* in the schematic. (See [Figure 7-13](#) on page 462)

- In response to the second prompt

> Select Reflection Instance Terminal on Schematic

Select the output of the power amplifier, the input terminal to the *port adapter* in the schematic. (See [Figure 7-14](#) on page 462)

So the *port adapter's* reflection (both `mag` and `phase`) is swept, which is the same as sweeping the power amplifier's load. In the Smith chart you see input reflection plots as seen by *port 5*.

The reflection contour plot shows the effect of the power amplifier's load on the input reflection while the power contour plot shows the effect of the power amplifiers's load on the power gain.

For reflection contours, the following terminal selections apply.

For *Single Refl/RefRefl Terminal* select one terminal.

One reflection coefficient of that terminal is computed as

$$\text{gamma1} = \frac{Z - Z0}{Z + Z0}$$

where *Z* is the large signal impedance at the fundamental.

The resulting Smith chart plots contours where `gamma1` is constant.

For *Single Refl/RefRefl Term and ref Term* select two terminals.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

This is the differential case for number 1. The different voltages of the two nets are used in the large signal impedance calculation.

- For *Separate Refl and RefRefl Terminals* select two terminals

Two reflection coefficients for each terminal are computed as

$$\gamma_1 = \frac{Z_1 - Z_0}{Z_1 + Z_0}$$

$$\gamma_2 = \frac{Z_2 - Z_0}{Z_2 + Z_0}$$

where  $Z_1$  and  $Z_2$  are the large signal impedance for each terminal at the fundamental.

The resulting Smith chart plots  $\gamma_2$  and contours where  $\gamma_1$  is constant.

For *+ - Refl and + - Ref Refl Terminals* select 4 terminals

This is the differential case for number 3.

For power contours, the following terminal selections apply.

For *Single Power/Ref Terminals* select one terminal.

The power ( $P_1$ ) and the reflection coefficient ( $\gamma_1$ ) of that terminal are computed. The resulting Smith chart plots  $\gamma_1$  and contours where  $P_1$  is constant.

For *Single Power/Refl Term and ref Term*

This is the differential case for number 1.

For *Separate Power and Refl Terminals* select 2 terminals

The power of the first terminal ( $P_1$ ) and the reflection coefficients of the second terminal ( $\gamma_2$ ) are computed. The resulting Smith chart plots  $\gamma_2$  and contours where  $P_1$  is constant.

For *+ - Power and + - Refl Terminals*

This is the differential case for the third case, *Separate Refl and RefRefl Terminals*.

Are Constant Power Contours the Same as Constant Gain Contours?

In general, *constant power contours* are not the same as *constant power gain contours*. The simulator does not maintain any input impedance match while it sweeps the load to generate load-pull contours. The resulting contours are not constant power gain contours, but are only constant power contours. However, if the input reflection coefficient and input power do not change much over the load sweep range, the generated contours are good

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

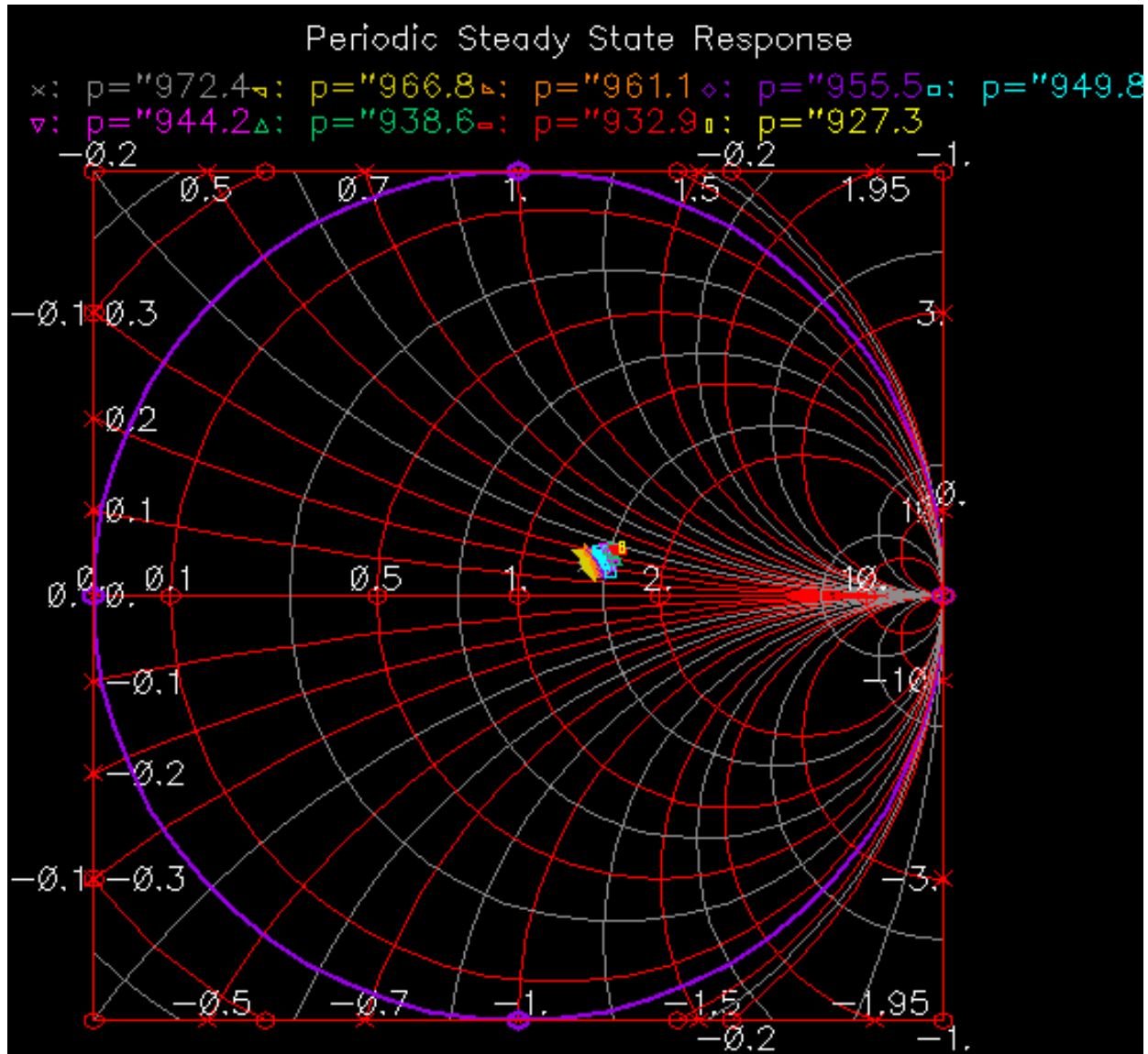
### Modeling Transmitters

---

approximations of constant gain contours. By plotting constant power contours in the input reflection coefficient plane, you can see how both change when you sweep load reflection coefficient. If the input power does not vary with load, the constant power contours are also constant power gain contours. If the input power does vary with the load, the constant power contours tell you how much you have to change the input matching network during the load sweep to maintain a constant input power.

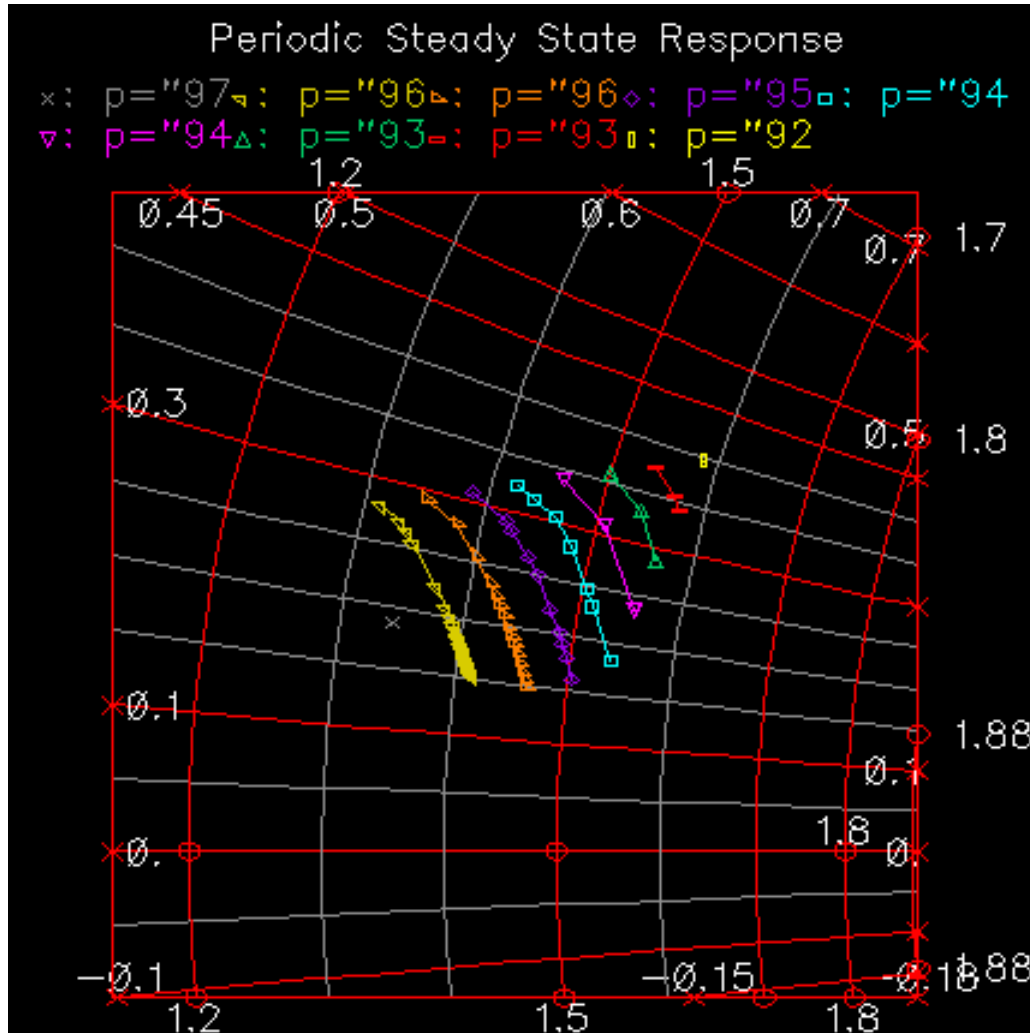
1. In the Direct Plot form, do the following:
  - a. Highlight *Replace* for *Plot Mode*.
  - b. Highlight *Power Contours* for *Function*.
  - c. Highlight *Single Power/Ref Terminal*.
  - d. Highlight *Magnitude* for *Power Modifier*.
  - e. Type 9 for *Number of Contours*, if necessary.
  - f. Type 50.0 for *Reference Resistance*, if necessary.
  - g. Highlight 1 G for *Output Harmonic*.
2. In the Schematic window, click the terminal at Port 5 (See [Figure 7-13](#) on page 462).
3. The plot of input reflection coefficients appears as in [Figure 7-15](#) on page 467.

Figure 7-15 Input Reflection Coefficients Generated by Sweeping Load



4. In the Waveform window, choose *Zoom – ZoomIn* and then click and drag with the mouse to form a rectangle that includes the area you want to see. The magnified view is shown in Figure 7-15.

### Magnified View of Input Reflection Coefficients



The real part of the input impedance varies from about 1.3 to 1.6 in normalized units and the imaginary part varies from 0.15 to 0.4 in normalized units. The input power varies from 927 nW to 967 nW. You find these numbers by placing the cursor on the end points of the outside contours. You read the impedance and power/reflection for that cursor location at the top of the window.

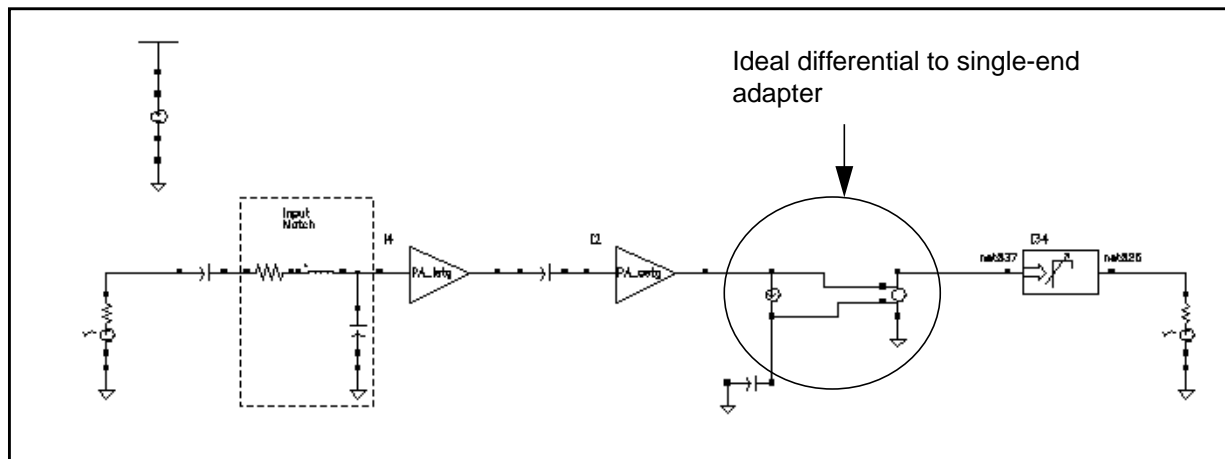
The numbers vary slightly as the load is swept. The objective here is to determine how much the input reflection coefficient and the input power vary as the load varies. If the input quantities vary significantly, then the generated load pull contours do not coincide with the constant power gain contours because the input power does not remain constant. The amount of variation in the input reflection coefficient tells you by how much you would have

to re-tune the input matching network to maintain constant input power as you sweep the load. You have to decide whether this much variation matters.

## Moving to Differential Mode

Up until now, the examples used in this chapter have been single-ended. Since many circuits are differential, Figure 7-15 shows a simple way to modify the example circuit to make it differential.

### Simple Differential Circuit



If your circuit has one or more differential ports, you can translate between single-ended and differential circuitry using linear-dependent controlled sources. The extra circuitry added to Figure 7-15 shows a modified version of the circuit where the output has been modified to make it differential. The transition from single-ended to differential output was made using the controlled sources  $F0$  and  $E1$  (circled in Figure 7-15).

The controlled sources,  $cccs$  and  $vcvs$ , are available in *analogLib*. The gain of  $cccs$  is  $-1$  to be consistent with the polarity of the  $vcvs$ . The  $vcvs$  ( $E1$ ) is the *Name of voltage source* parameter in the  $cccs$ .

### Using S-Parameter Input Files

In this example, you create an S-parameter data file then you enter it back into the original circuit. The S-parameter data file replaces that portion of the circuit originally used to create the S-parameters. To enter tabulated S-parameters from some other source, record the data in the format shown in the S-parameter data file and select the appropriate numerical options in the *nport*.

This example shows you how to

- Set up and simulate the *sparamfirst* circuit and create an S-parameter data file (`sparam.practice`) for the *sparamfirst* schematic
- Add an *n2port* device in the frequency domain to a modified copy of the *sparamfirst* schematic.
- Run another simulation of the modified *sparamfirst* circuit where the *n2port* component reads in the S-parameter data file
- Plot and compare the two sets of data produced.
- The output of the original branch of the circuit
  - The output of the modified branch of the circuit where the *n2port* component replaced the components in the original branch. The *n2port* reads in the `sparam.practice` S-parameter file you created.

When you superimpose the two plots, it is clear that the S-parameter file produces the same plot as the original components.

## Setting Up the EF\_example Schematic for the First Simulation

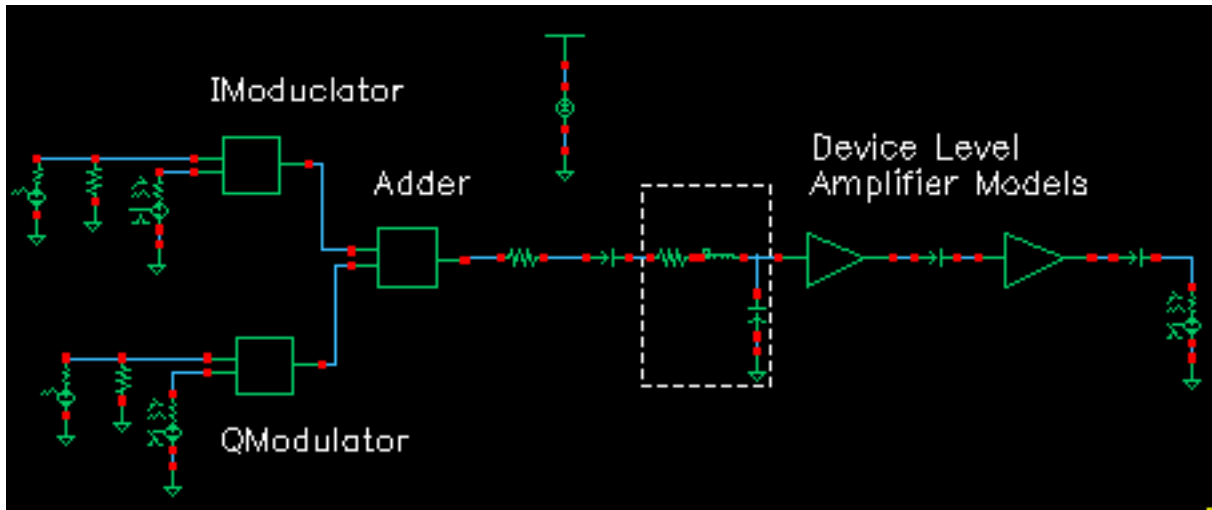
This example requires that you

- Open the *EF\_example* schematic
- Create a new, empty cellview named *sparamfirst*
- Copy part of the *EF\_example* schematic and paste it in the *sparamfirst* schematic
- Simulate the new schematic.
- This first simulation generates the S-parameters of a linear time-invariant part of the original circuit (*EF\_example*). The S-parameter file is stored so you can use it in the second simulation to import S-parameters.

### Opening the EF\_example Schematic

1. Open the *EF\_example* schematic as described in [“Opening the EF\\_example Circuit in the Schematic Window”](#) on page 370.

The *EF\_example* schematic is the original schematic used in the first Envelope example in this chapter. Leave the *EF\_example* schematic open in the background.



### Creating and Editing the New Schematic

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

The Create New File form appears.

2. In the New File form, do the following:

- a. Choose *my\_rfExamples* for *Library Name*, the editable copy of *rfExamples*.

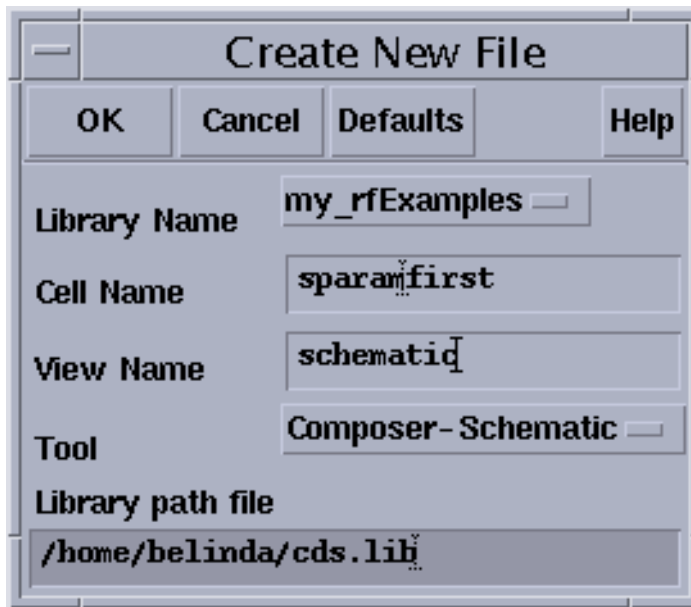
Select the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

In the *Cell Name* field, type *sparamfirst*, a name for the new view.

- b. In the *View Name* field, type *schematic*.

- c. Choose *Composer-Schematic* for *Tool*.

d. Click *OK*.

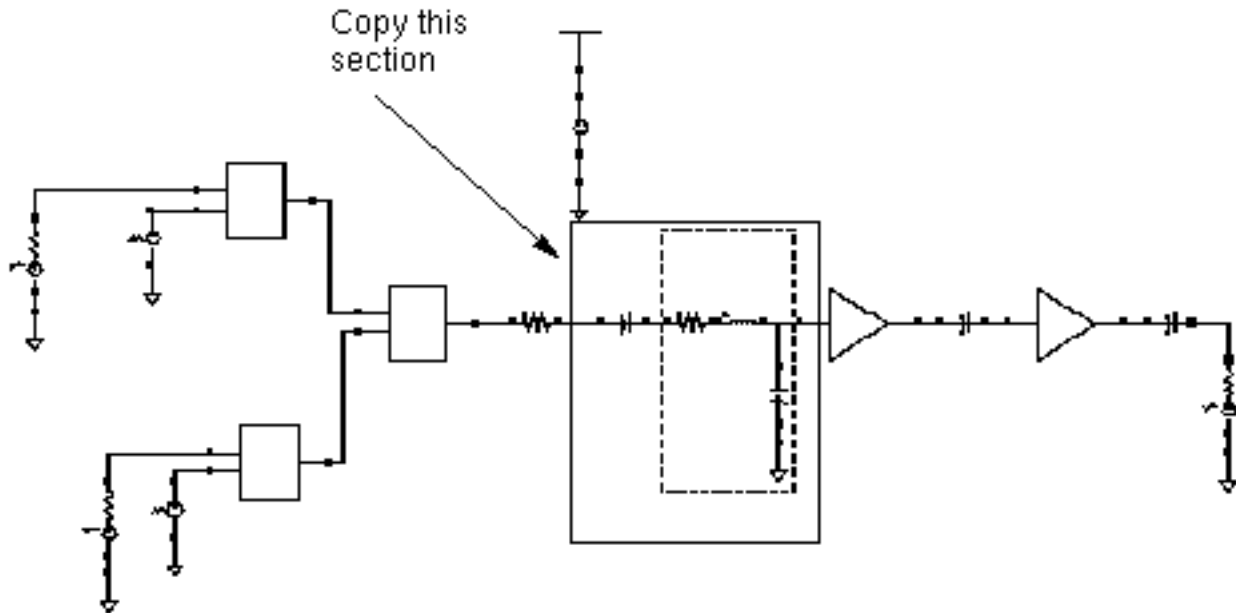


After you click *OK*, the new empty Schematic window for *sparamfirst* appears.

3. In the *EF\_example* Schematic window, copy the part of the schematic shown in Figure [7-15](#) and paste it into the new Schematic window.



### Portion of EF\_example Schematic to Copy



a. In the *EF\_example* Schematic window, choose *Edit - Copy* and follow the prompts at the bottom of the Schematic window.

b. Following the prompt,

> point at object to copy

left click and drag to create a box around the part of the *EF\_example* circuit indicated in [7-15](#).

The selected *EF\_example* components are highlighted in yellow.

Following the prompt,

> point at reference point for copy

click inside the outlined elements.

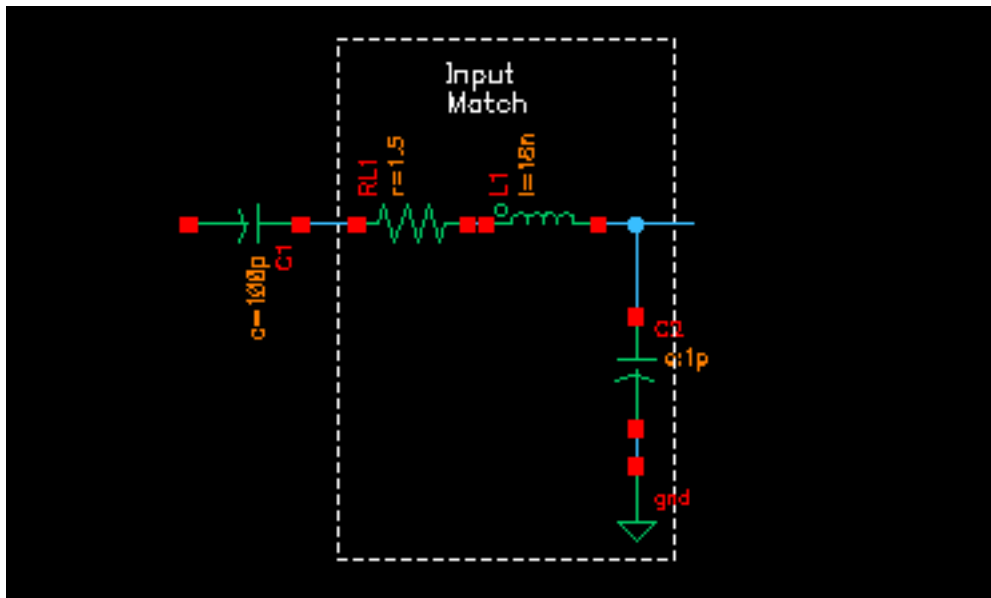
Following the prompt,

> point at destination point for copy

move the cursor to the *sparamfirst* schematic window and click there. This copies the selected part of the *EF\_example* circuit into the empty Schematic window.

The empty Schematic window now contains the portion of the *EF\_example* schematic shown in [7-15](#).

If necessary, choose *Window - Fit* to center the copied section of the *EF\_example* circuit in the *sparamfirst* Schematic window.



4. In the original *EF\_example* Schematic window, choose *Window - Close*.
5. The *EF\_example* Schematic window closes.

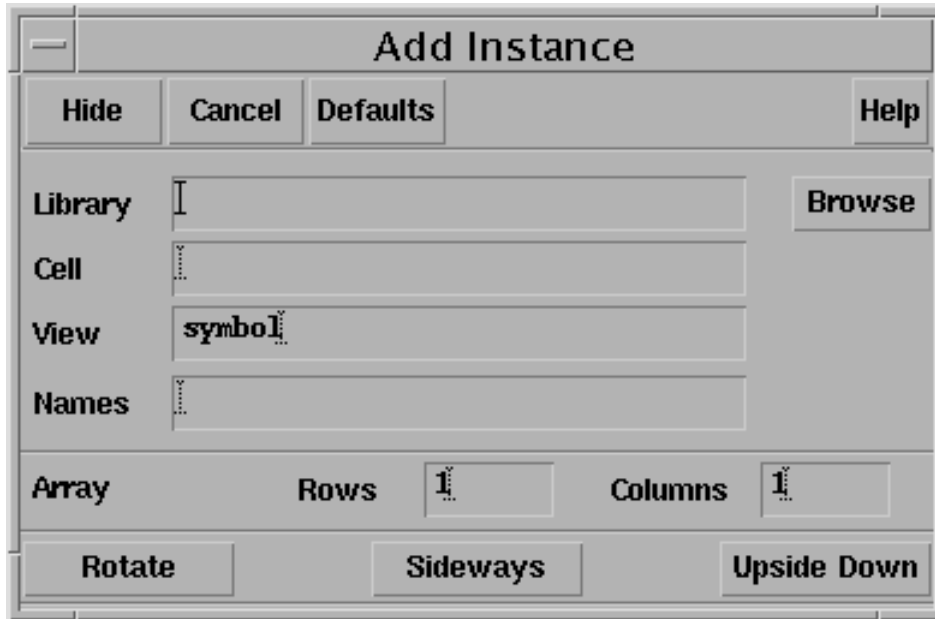
## Adding Components to the Schematic

1. In the *sparamfirst* Schematic window, choose *Add – Instance*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

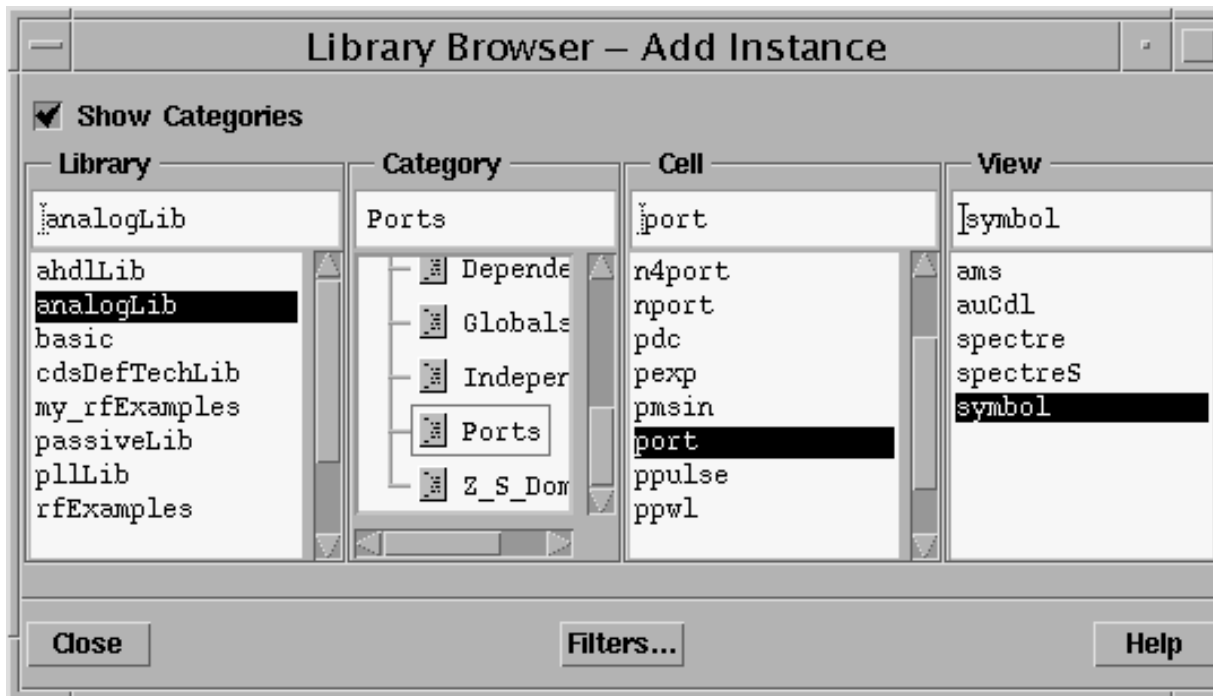
### Modeling Transmitters

The Add Instance form appears.



2. In the Add Instance form, click *Browse*.

The Library Browser – Add Instance form appears.



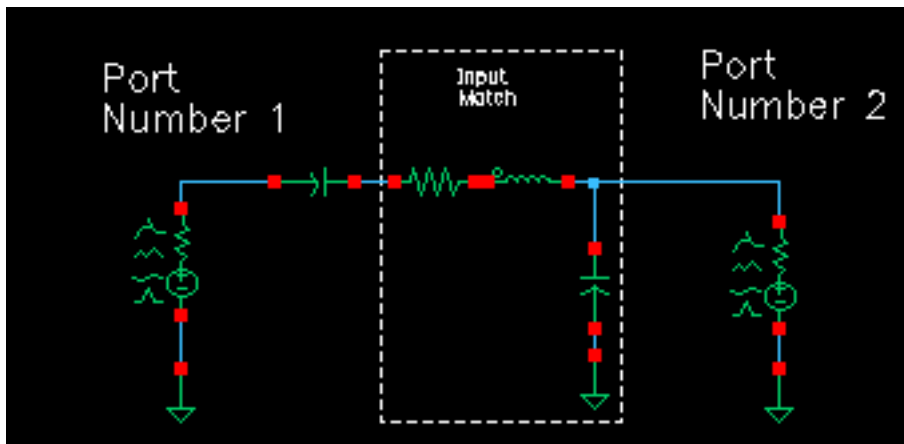
3. In the Library Browser – Add Instance form, do the following:

- a. Select the *analogLib* and highlight *Show Categories*.
- b. Scroll down and select *Sources* in the *Category* list box.  
The *Category* list box expands to display several subcategories under *Sources*.
- c. Scroll down and select the *Ports* subcategory under *Sources*.
- d. Select *port* for *Cell*.
- e. Select *symbol* for *View*.

A copy of the port component moves with the cursor into the new Schematic window.

Place two *port* symbols in the *sparamfirst* Schematic window as shown in Figure 7-15.

#### Placement and Port Numbering in the *sparamfirst* Schematic



4. In the Library Browser – Add Instance form, do the following:

- a. Select *analogLib* for *Library*.
- b. Select *Everything* for *Category*.
- c. Select *gnd* for *Cell*.
- d. Select *symbol* for *View*.

5. Place two *gnd* components in the Schematic window as shown in Figure 7-15. Press *Esc* when you are done.

6. In the Schematic window, select *Add – Wire (narrow)* and wire up the new components in the *sparamfirst* Schematic window. Press *Esc* when you are done.
7. In the new Schematic window, choose *Edit – Properties – Objects*.
8. For each of the *port* components in the new schematic, do the following in the Edit Object Properties form:
  - a. Select the *port*.
  - b. Type the appropriate number in the *Port number* field.

Figure 7-15 shows the appropriate number to type. Leave the Port resistance at its default of 50 Ohms.
  - c. Click *Apply*.
9. In the new Schematic window, choose *Design – Check and Save*.

### Setting Up the *sparamfirst* Schematic

1. In the *sparamfirst* Schematic window, open the Simulation window as described in [“Opening the Simulation Window”](#) on page 372.
2. In the Simulation window, set up the model libraries as described in [“Setting Up the Model Libraries”](#) on page 373.
3. Running the SP Simulation

Run an SP analysis to write out an S-parameter file, `sparam.practice`, that describes the *sparamfirst* circuit.

In the Simulation window, choose *Analyses – Choose*.

The Choosing Analyses form appears.

1. In the Choosing Analyses form, do the following:
  - a. Highlight *sp* for *Analysis*.
  - b. Highlight *Frequency* for *Sweep Variable*.
  - c. Highlight *Start – Stop* for *Sweep Range*.
  - d. Type 100K for *Start* and 10G for *Stop*.
  - e. Choose *Automatic* for *Sweep Type*.

- f. Highlight *no* for *Do Noise*.
2. In the Choosing Analyses form, click *Options*.
3. The S-parameter Options form appears.
4. In the OUTPUT PARAMETERS section of the S-parameter Options form, type the path for the output S-parameter file in the *file* field. Click *Apply*. Click *OK*.

**OUTPUT PARAMETERS**

**file**

**datafmt**  **spectre**  **touchstone**

5. In the Choosing Analyses form, click *Apply*. Click *OK*
6. In the Simulation window, select *Simulation – Netlist and Run*.
7. Check the CIW for a message that says the simulation completed successfully.

The sp analysis wrote an S-parameter file, `sparam.practice`, to the directory you indicated.

### The S-Parameter File

Use the S-parameter output file, `sparam.practice`, you created as input for the next simulation. You can use a text editor to open and examine the format of the S-parameter output file. You can use this format to create S-parameter files from other sources.

The top of the `sparam.practice` S-parameter file looks like this.

```
; S-parameter data file /hm/belinda/sparam.practice.
; Tue Feb 28 18:10:27 2006
; Number of ports is 2

reference resistance
    port2 = 50.000000
    port1 = 50.000000

format freq:  s1:1(real,imag)  s2:1(real,imag)
              s1:2(real,imag)  s2:2(real,imag)

1.00000000e+05:  0.99996,      -0.00628293      4.02664e-05,      0.00628293
                4.02664e-05,      0.00628293      0.99996,      -0.00634576
1.25892541e+05:  0.999936,      -0.00790956      6.38165e-05,      0.00790956
```

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

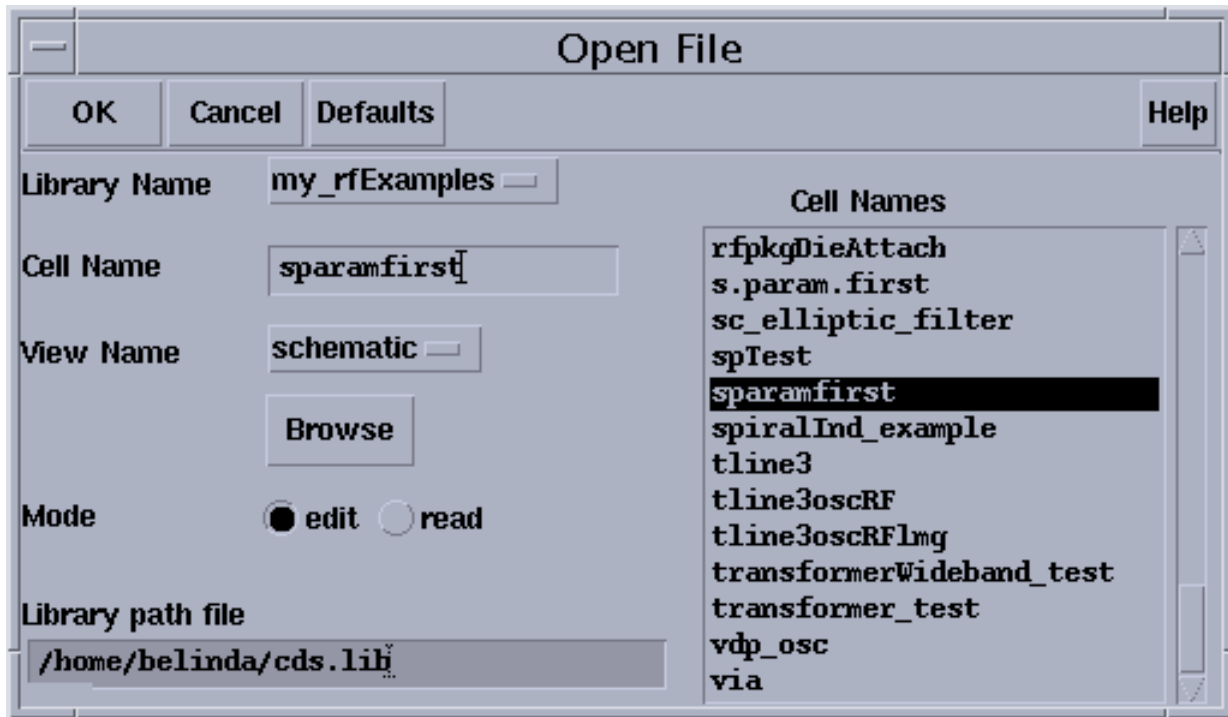
## Modeling Transmitters

```
1.58489319e+05: 6.38165e-05, 0.00790956 0.999936, -0.00798866
0.999899, -0.00995718 0.000101139, 0.00995717
0.000101139, 0.00995717 0.999898, -0.0100567
```

### Setting Up and Running the Second sp Simulation

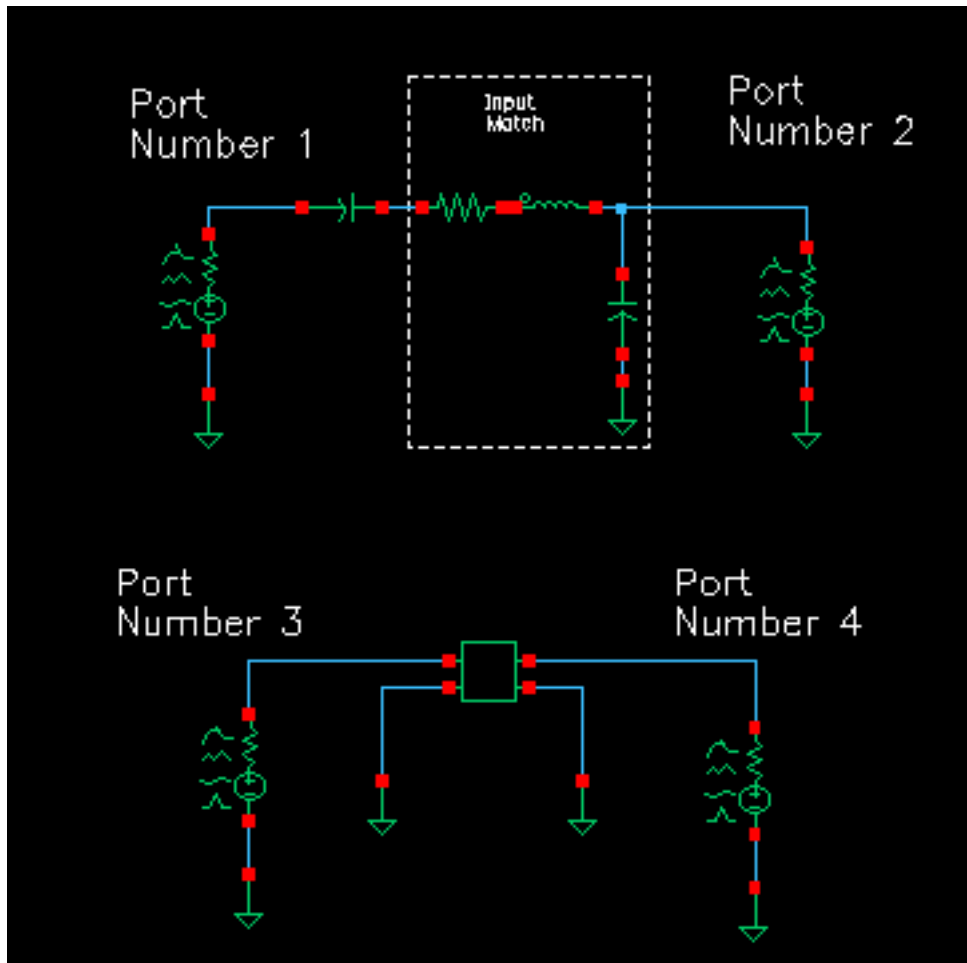
In this simulation, use the S-parameter output file, `sparam.practice`, from the first simulation as input to the second simulation. The S-Parameter file models the components in the `sparamfirst` Schematic.

If necessary, open the `sparamfirst` schematic.



1. Add components to the `sparamfirst` schematic you created for the SP analysis as shown in Figure 7-16.

**Figure 7-16 Wired sparamfirst Schematic**



The components to add are described in the following table. See [“Adding Components to the Schematic”](#) on page 474 for information on adding components to a schematic.

**Table 7-4 Components to Add to the sparamfirst Schematic**

Library	Category	Subcategory	Cell	View	Number to Add
analogLib	Sources	Ports	n2port	symbol	1
analogLib	Sources	Ports	port	symbol	2
analogLib	Everything	None	gnd	symbol	4

- Wire the components as shown in Figure [7-16](#).



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

3. Edit the properties on the two new ports to assign port numbers 3 and 4 as shown in Figure [7-16](#). Then edit the properties on the new *n2port*.
4. In the Schematic window select the *port* on the lower left.
5. In the Schematic window, choose *Edit – Properties – Objects*.
6. In the Edit Object Properties form, do the following:
  - a. Type 3 in the *Port number* field.
  - b. [Figure 7-16](#) on page 480 shows the appropriate number to type. Leave the *Port resistance* at the default of 50 Ohms.
  - c. Click *Apply*.
  - d. In the Schematic window, select the *port* on the lower right.
  - e. Type 4 in the *Port number* field.
  - f. Leave the *Port resistance* at the default of 50 Ohms.
  - g. Click *Apply*.
7. In the Schematic window, select the *n2port*.

The Edit Object Properties form changes to display information for the *n2port*.

8. In the Edit Object Properties form, do the following and click *OK*:
  - a. In the *S-parameter data file* field, type the absolute path to the S-parameter file you created in the first simulation.  
`/hm/belinda/sparam.practice`
  - b. Choose *rational* for *Interpolation method*.  
The form changes to let you add additional information.
  - c. Type `.001` for *Relative error*.
  - d. Type `1e-6` for *Absolute error*.
  - e. Type `6` for *Rational order*.
  - f. Select *no* for *Thermal Noise*.
  - g. Select *spectre* for *S-parameter data format*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

The completed Edit Object Properties form for the *n2port* looks like this.

CDF Parameter	Value	Display
S-parameter data file	belinda/sparam.practice	off <input type="checkbox"/>
Multiplier		off <input type="checkbox"/>
Scale factor		off <input type="checkbox"/>
Interpolation method	rational <input type="checkbox"/>	off <input type="checkbox"/>
Relative error	.001	off <input type="checkbox"/>
Absolute error	1e-6	off <input type="checkbox"/>
ROM data file		off <input type="checkbox"/>
Rational order	6	off <input type="checkbox"/>
No. of Harmonics for PSS		off <input type="checkbox"/>
Thermal Noise	no <input type="checkbox"/>	off <input type="checkbox"/>
Use smooth data windowing	<input type="checkbox"/>	off <input type="checkbox"/>
S-parameter data format	spectre <input type="checkbox"/>	off <input type="checkbox"/>
Thermal noise model	<input type="checkbox"/>	off <input type="checkbox"/>

9. In the Schematic window, choose *Design – Check and Save*.
10. In the Simulation window, choose *Simulation – Netlist and Run*.

### Plotting Results

1. In the Simulation window, choose *Results – Direct Plot – Main Form*.

The Direct Plot form appears.

2. In the Direct Plot form, do the following:
  - a. Choose *Replace* for *Plotting Mode*.
  - b. Highlight *sp* for *Analysis*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

- c. Highlight *SP* for *Function*.
- d. Highlight *Z-Smith* for *Plot Type*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The Direct Plot form looks like the following.

The image shows a dialog box titled "Direct Plot Form" with a standard Windows-style title bar. At the top, there are three buttons: "OK", "Cancel", and "Help". Below the buttons, the "Plotting Mode" is set to "Replace" with a dropdown arrow. The "Analysis" section contains a radio button labeled "sp" which is selected. The "Function" section contains a grid of radio buttons for various parameters: SP (selected), ZP, YP, HP, GD, VSWR, NFmin, Gmin, Rn, m, NF, Kf, B1f, GT, GA, GP, Gmax, Gmsg, Gumx, ZM, NC, GAC, GPC, LSB, and SSB. The "Description" field is labeled "S-Parameter". The "Plot Type" section has radio buttons for "Rectangular", "Z-Smith" (selected), "Y-Smith", and "Polar". Below this, there is a grid of buttons for S-parameters: S11, S12, S13, S21, S22, S23, S31, S32, S33, and a button labeled "S" followed by two input fields, both containing the number "1". At the bottom, there is a checkbox labeled "Add To Outputs" which is unchecked. A footer message reads "> To plot, press Sij-button on this form...".

- e. Click *S11*.

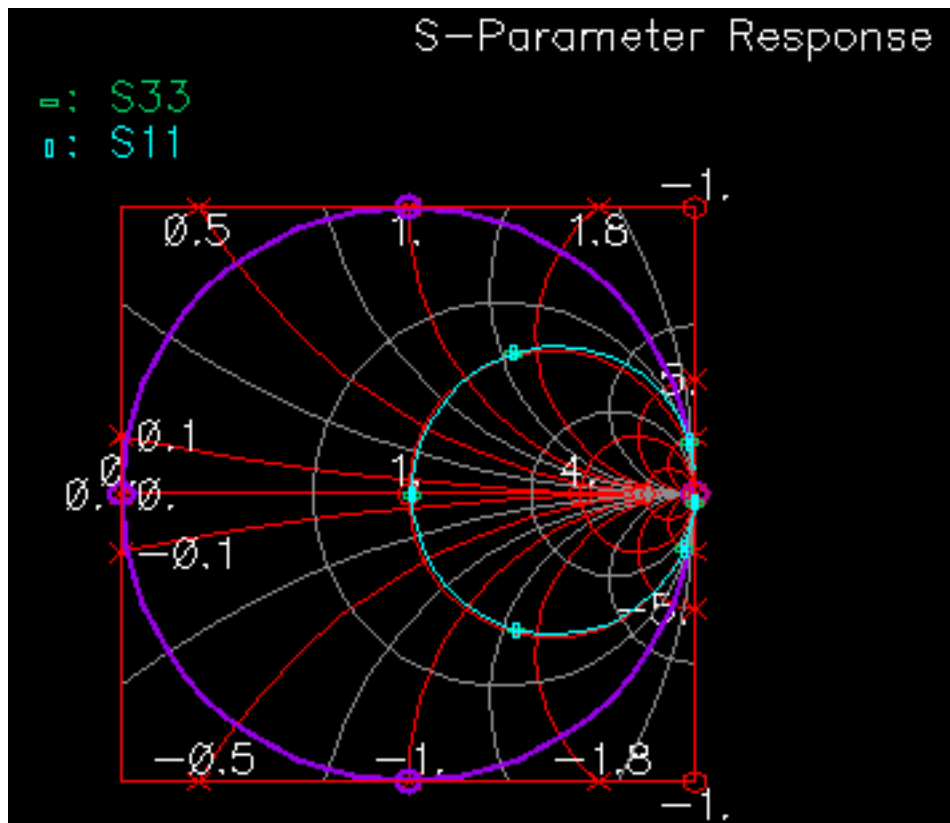
The plot for *S11* appears in the Waveform window.

- 3. In the Direct Plot form, do the following:

- a. Choose *Append* for *Plot Mode*.

- b. Click *S33*.

The plot for *S33* is appended to the *S11* plot. The two plots lie one on top of the other which shows that the two plots are identical. Thus the results produced by the first simulation are the same as those produced by the second simulation which used the S-parameter input file.



You can also check other plots for equivalency. For example, you can do the following to plot *S21* and *S43*.

1. Choose *Replace* for *Plotting Mode*.
2. Plot *S21* as described in "[Plotting Results](#)" on page 482.
3. Plot *S43* as follows.

- a. Choose *Append* for *Plotting Mode*.
- b. Select 4 in the first cyclic field to the right of S.
- c. Select 3 in the second cyclic field to the right of S.
- d. Click S to create the plot.

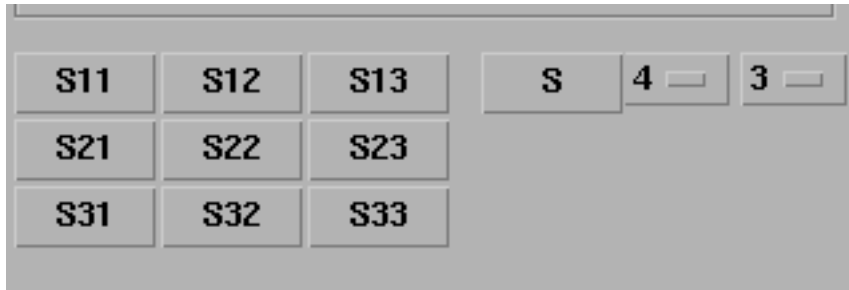
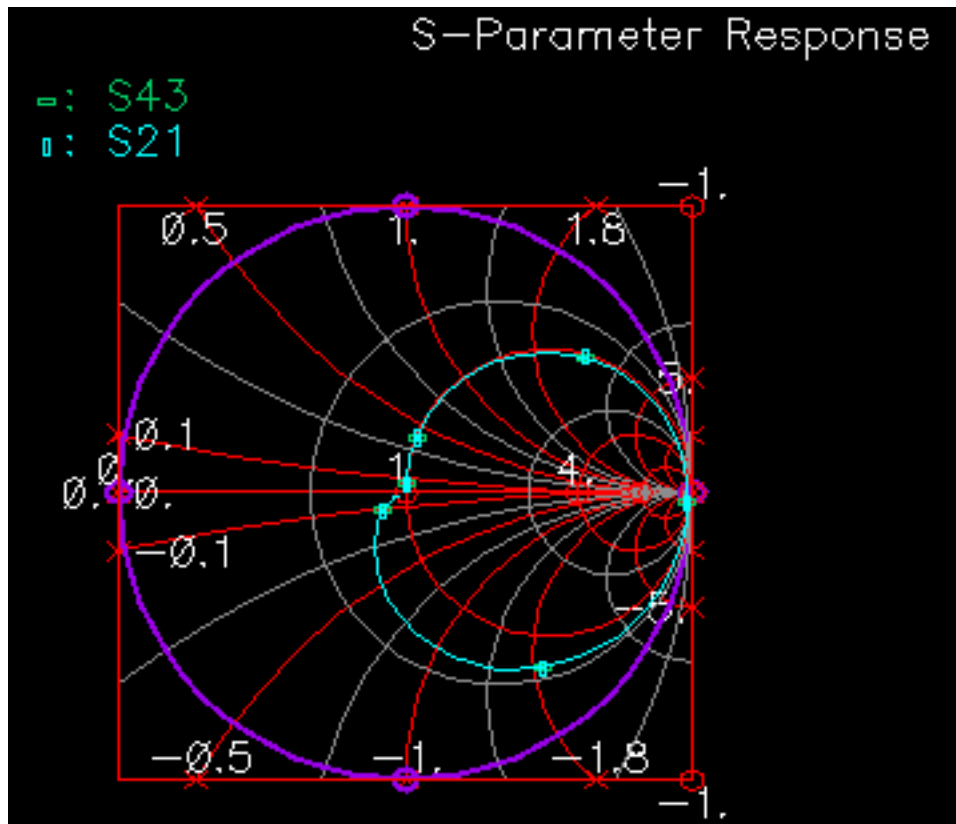


Figure 7-4 is the plot produced by appending S431 to the S21 plot.

### Plot Showing S21 and S43



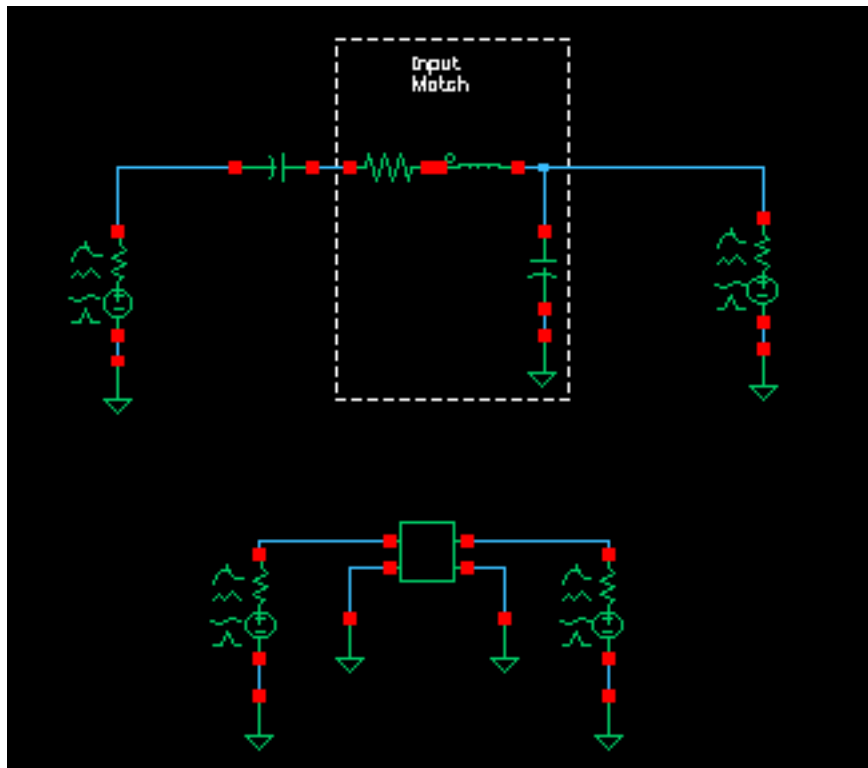
## Using the S-Parameter Input File with a Spectre RF Envp Analysis

Use the `sparam.practice` S-parameter format data file as input to an Envelope analysis. Again, it compares results of an *envlp* analysis using the S-parameter file to results of an *envlp* analysis using the original components.

### Setting Up the Schematic

1. Open the *EF\_example* schematic. Use *File – Open* in the CIW.
2. Open the *sparamfirst* schematic and move it to the background.

Figure 7-17 The *sparamfirst* Schematic



3. In the *EF\_example* schematic. choose *Design – Save As* in the Schematic window.  
The Save As form appears.
4. In the Save As form, do the following:

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

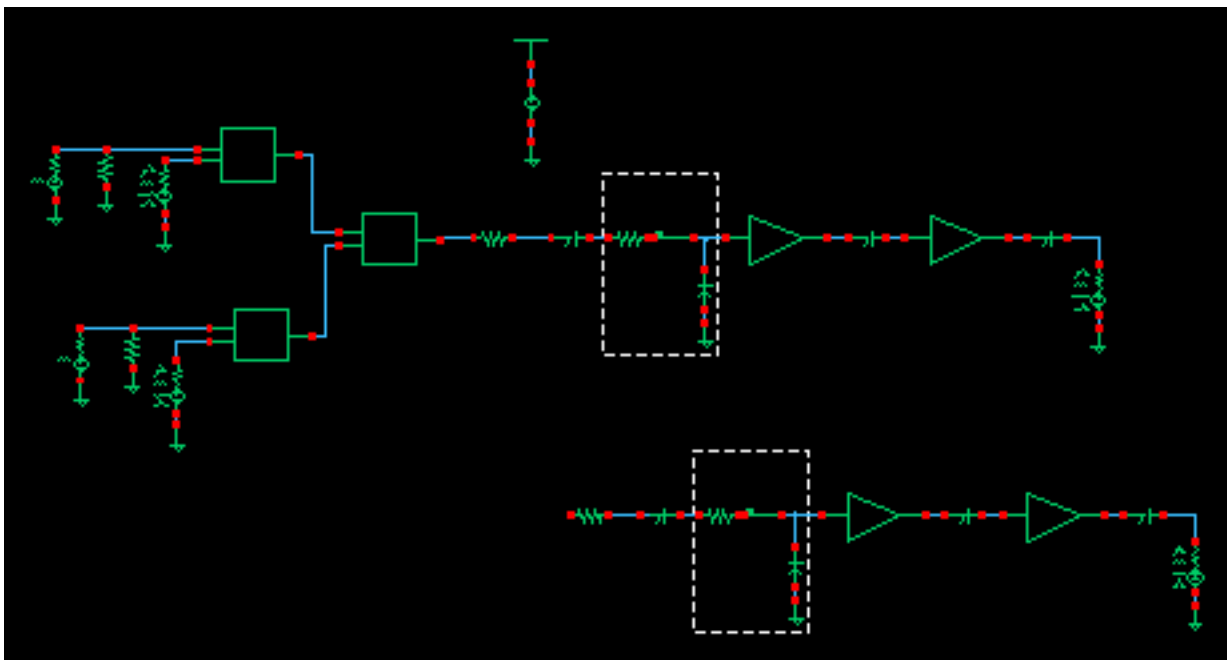
### Modeling Transmitters

- a. In the *Library Name* field, type the name of your local, editable copy of the *rfExamples* library.
- b. In the *Cell Name* field, type *EF\_example\_copy*, the name for the copy of the *EF\_example* schematic.
- c. Click *OK*.

You now have a copy of the *EF\_example* schematic called *EF\_example\_copy*.

In the *EF\_example* Schematic window., choose *Window – Close* to close the *EF\_example* schematic.

5. Open the *EF\_example\_copy* schematic.
6. In the *EF\_example\_copy* Schematic, copy everything to the right of the adder and paste the copy below the original.

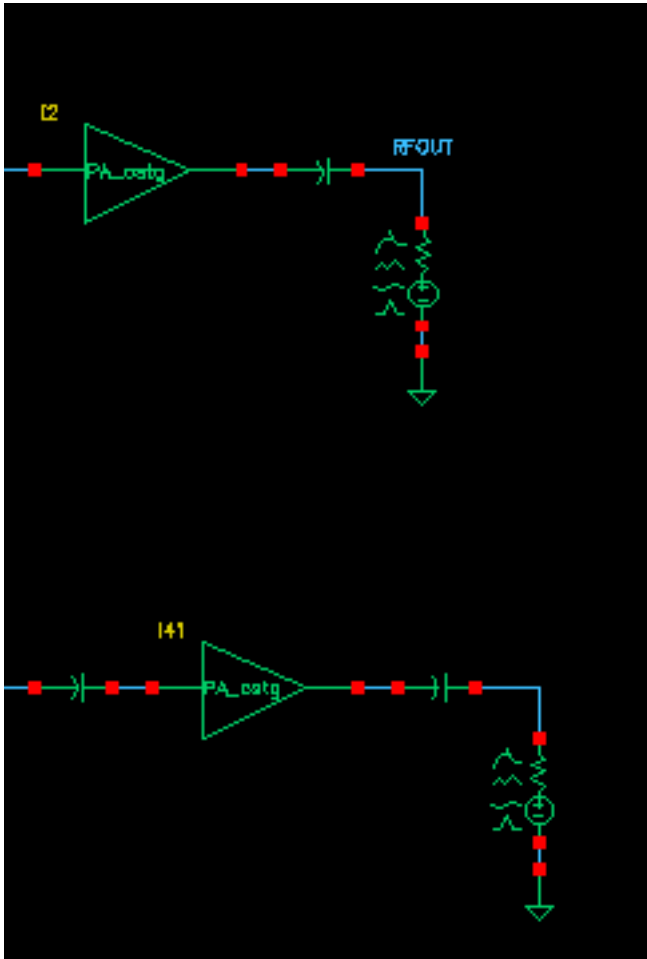


#### **Important**

Be sure to remove the *RFOUT* label from the duplicate branch or the two branches are shorted together at their outputs.

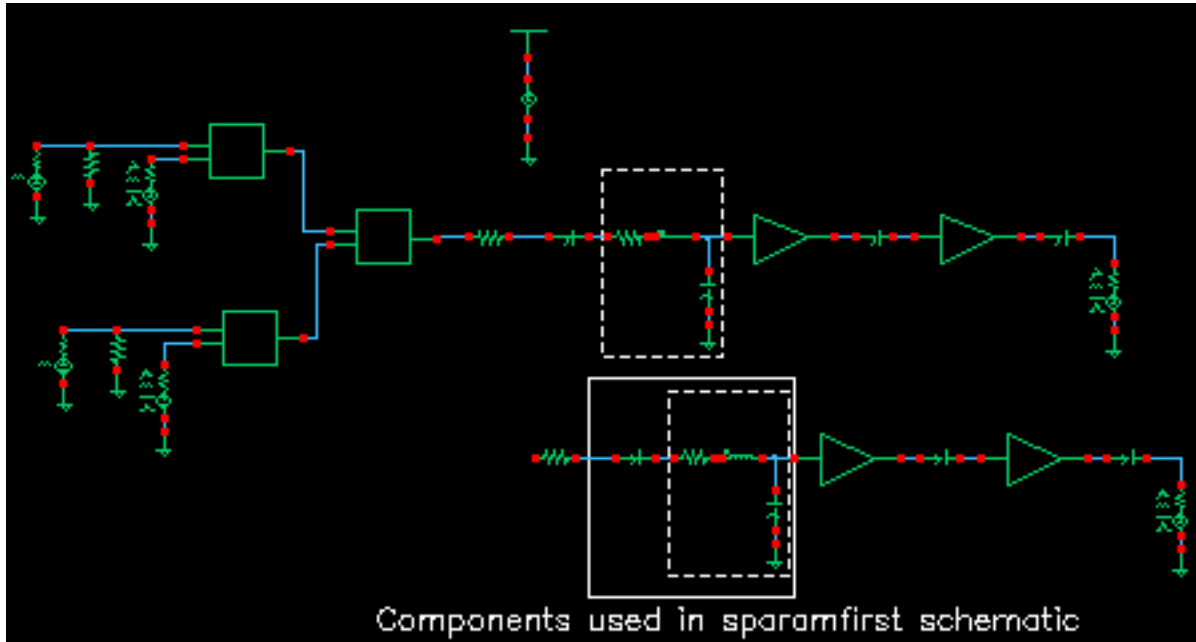


7. To delete RFOUT from the bottom branch, select *Edit — Delete*, highlight *RFOUT* and then press *Esc*.



8. In the lower branch of the *EF\_example\_copy* schematic, delete the circuitry you used to create the S-parameter file in the *sparmfirst* schematic, see Figure 7-18.

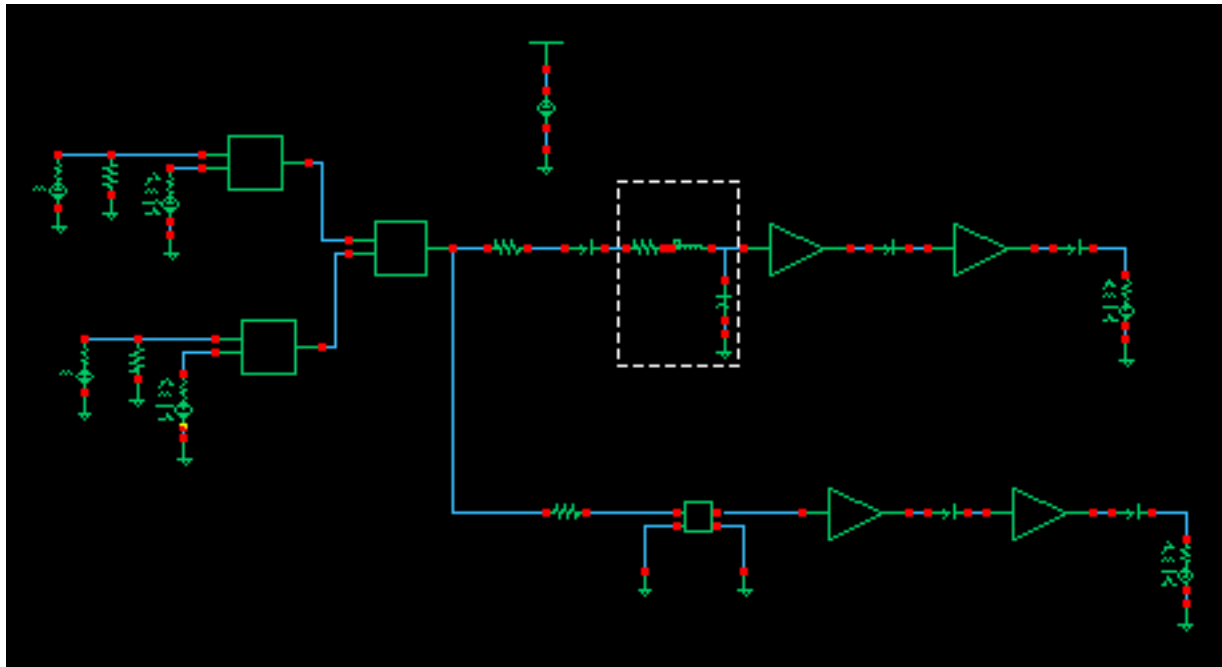
Figure 7-18 Components to Delete from the EF\_example\_copy Schematic



9. Choose *Edit – Delete* and follow the prompts at the bottom of the *EF\_example\_copy* schematic window.
10. Following the prompt,  
> point at object to delete  
left click and drag to create a box around the indicated part of the *EF\_example\_copy* circuit.  
  
The selected *EF\_example\_copy* components are deleted.
11. Press *Esc* to stop deleting.
12. Replace the deleted components in the *EF\_example\_copy* schematic with the *n2port* component and the two attached *gnd* (ground) components from the *sparamfirst* schematic. The *n2port* and *gnd* components are shown in [Figure 7-17](#) on page 487.
  - a. In the *sparamfirst* schematic window, choose *Edit – Copy* to copy the *n2port* and the two attached *gnd* components. Follow the prompts at the bottom of the *sparamfirst* schematic window.  
  
**Note:** (Close the *sparamfirst* schematic window when you are done.)
  - b. As you move the cursor into the *EF\_example\_copy* Schematic window, a copy of the *n2port* and *gnd* components moves with the cursor. Place the components in the *EF\_example\_copy* Schematic window as shown in [Figure 7-18](#)

13. In the Schematic window, select *Add – Wire (narrow)* and wire up the new components in the *EF\_example\_copy* Schematic window. Press *Esc* when you are done.

**The EF\_example\_copy Schematic with n2port Component in Place**



14. In the Schematic window, select the *n2port* component and choose *Edit – Properties – Object*. Then verify that the *n2port* component has the following properties.
- The absolute path to the S-parameter data file displays in the S-parameter data file field. In this example,  
`/hm/belinda/sparam.practice`
  - Interpolation method* is *Rational*.
  - Relative error* is `.001`.
  - Absolute error* is `1e-6`.
  - Rational order* is `6`.
  - Thermal Noise* is *no*.
  - S-parameter data format* is *spectre*.

The completed Edit Object Properties form for the *n2port* looks like this.

CDF Parameter	Value	Display
S-parameter data file	belinda/sparam.practice	off <input type="checkbox"/>
Multiplier		off <input type="checkbox"/>
Scale factor		off <input type="checkbox"/>
Interpolation method	rational <input type="checkbox"/>	off <input type="checkbox"/>
Relative error	.001	off <input type="checkbox"/>
Absolute error	1e-6	off <input type="checkbox"/>
ROM data file		off <input type="checkbox"/>
Rational order	6	off <input type="checkbox"/>
No. of Harmonics for PSS		off <input type="checkbox"/>
Thermal Noise	no <input type="checkbox"/>	off <input type="checkbox"/>
Use smooth data windowing	<input type="checkbox"/>	off <input type="checkbox"/>
S-parameter data format	spectre <input type="checkbox"/>	off <input type="checkbox"/>
Thermal noise model	<input type="checkbox"/>	off <input type="checkbox"/>

15. In the Schematic window, choose *Design – Check and Save*.

### Setting Up and Running the Simulation

Set up and run an *env/p* analysis as described in the sections listed here.

- [“Opening the Simulation Window”](#) on page 372
- [“Setting Up the Model Libraries”](#) on page 373
- [“Editing PORT0 and PORT1 in the EF example Schematic”](#) on page 374
- [“Setting Up an Envelope Analysis”](#) on page 376.

When you set up the *envlp* analysis, use 30u for the *Stop Time* value.

The completed *envlp* Choosing Analyses form looks like this.

The screenshot shows the "Envelope Following Analysis" dialog box. It has a title bar "Envelope Following Analysis". Under "Engine", there are two unchecked checkboxes: "Shooting" and "Flexible Balance". Below this is a section with a "Clock Name" field containing "fff" and a "Select Clock Name" button. The "Stop Time" field contains "30u". Under "Output Harmonics", there is a "Number of harmonics" field with a dropdown arrow and the value "1". A "Start ACPR Wizard" button is located at the bottom right of this section. Below this is an "Oscillator" checkbox, which is unchecked. The "Accuracy Defaults (empreset)" section has three radio buttons: "conservative" (unchecked), "moderate" (checked), and "liberal" (unchecked). At the bottom left, the "Enabled" checkbox is checked. At the bottom right, there is an "Options..." button.

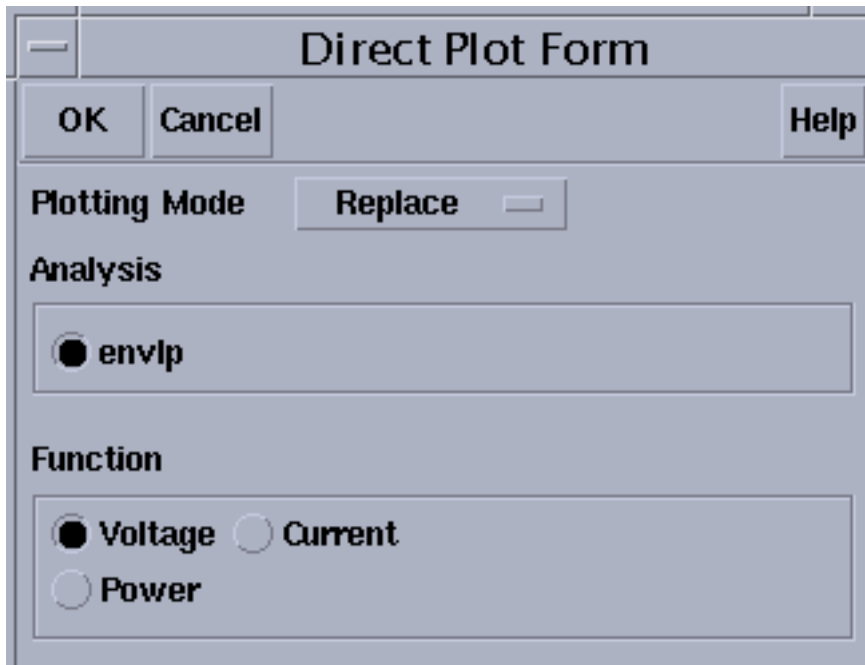
- In the Simulation window, choose *Simulation – Netlist and Run*.  
Check the CIW for a message that says the simulation completed successfully.

### Displaying the *envlp* Following Results

1. In the Simulation window, choose *Results – Direct Plot – envlp*.  
The Direct Plot form appears.

2. In the Direct Plot form, choose *Replace* for *Plotting Mode*.
3. Highlight *envlp* for *analysis*.
4. Highlight *Voltage* for *Function*.

The top of the form looks like this.



5. Highlight *Harmonic Time* for *Sweep*.
6. Highlight *Real* for *Modifier*.
7. Choose *1* for *Harmonic Number*.

The bottom of the form looks like this.

**Description: Harmonic Voltage vs Time**

Select

**Sweep**

spectrum  harmonic time  time

**Modifier**

Magnitude  Phase  dB20  
 Real  Imaginary

**Harmonic Number**

0  
1

Add To Outputs

> Select Net on schematic...

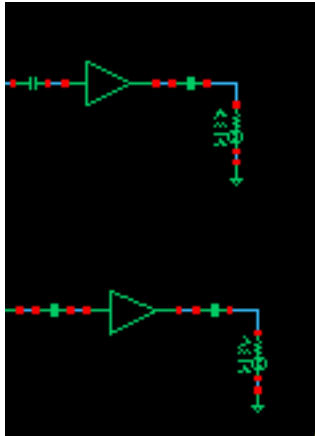
The prompt on the bottom of the Direct Plot form is

> Select Net on Schematic....

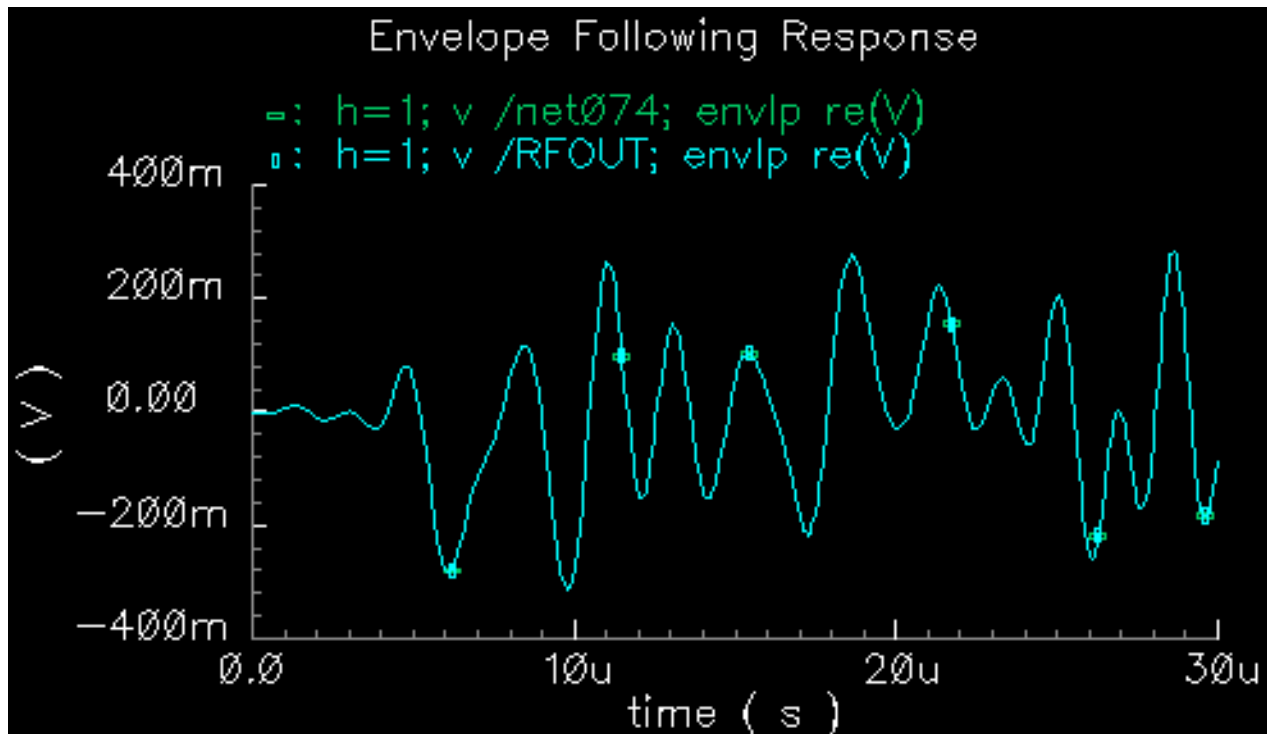
8. Follow the prompt. In the Schematic window, click one of the output nodes as shown in [“Output Nodes”](#) on page 496.
9. Change the *Plot Mode* to *Append*.
10. Click the other output node as shown in [“Output Nodes”](#) on page 496.

You can also confirm that the imaginary parts of the waveforms match.

Figure 7-19 Output Nodes



The coincident plots in the Waveform window show that the output is the same at both nodes. When the S-parameters are correct, the *n2port* device accurately simulates the circuitry represented by the S-parameters.





## Measuring AM and PM Conversion with Modulated PAC, AC and PXF Analyses

This section describes AM and PM small signal characterization in Spectre RF. AM/PM conversion measurements allow you to investigate the amplitude and phase characterization of RF circuits. The modulated PAC, AC and PXF analyses calculate conversion gain and other characteristics between AM, PM, and SSB sources and AM, PM, and SSB outputs.

AM/PM conversion computes transfer functions and gain measurements involving AM and PM inputs and outputs. In general, there are three possible types of inputs and outputs for which you might want to compute transfer functions.

- Unmodulated or single sideband (SSB)
- Amplitude modulated sinusoids (AM)
- Phase modulated sinusoids (PM)

### The Modulated Analysis Settings

In total, a modulated analysis can calculate 9 cross conversion metrics from 3 types of inputs (AM, PM and SSB) to 3 types of outputs (AM, PM and SSB). Use the *Modulated Analysis* section in the PAC and PXF Choosing Analyses forms to set up your analyses.

*Input Type* (for PAC analysis) and *Output Type* (for PXF analysis) allow you to choose whether to measure all 9 modulated conversions (choose *SSB/AM/PM*) or only 3 conversions (choose *SSB*).

*Output Modulated Harmonic List* (for PAC output modulations) and *Input Modulated Harmonic List* (for PXF modulated sources) specify a vector of harmonic indexes. You can type the indexes separated by spaces or you can choose them from a scrolling list.

*Input Modulated Harmonic* (for PAC) and *Output Modulated Harmonic* (for PXF) specify a single harmonic index for PAC input source modulation or for PXF output modulation. You can type the index or you can choose it from a scrolling list. This choice appears when you select the *SSB/AM/PM Input* or *Output Type*.

For *SSB Input Type* (PAC) or *Output Type* (PXF), specify an *Output Upper Sideband* or *Input Upper Sideband*. You can type the sideband index or you can choose it from a scrolling list.

For modulated PXF analysis, select the *Output Probe Instance* or the *Positive* and *Negative voltage Output Nodes* from the schematic.

This example illustrates how to

- Create the *EF\_AMP* schematic, a modified copy of the *EF\_example* schematic.
- Save a copy of the *EF\_AMP* schematic in the *my\_rfExamples* library. You use this schematic for other examples later.
- Set up and run the necessary PSS, modulated PAC and modulated PXF analyses.
- Compute transfer functions and gain measurements and display the resulting information with the Direct Plot form.
- After you run the simulation, use the Direct Plot form to plot the modulated analysis results.

## Creating the EF\_AMP Circuit

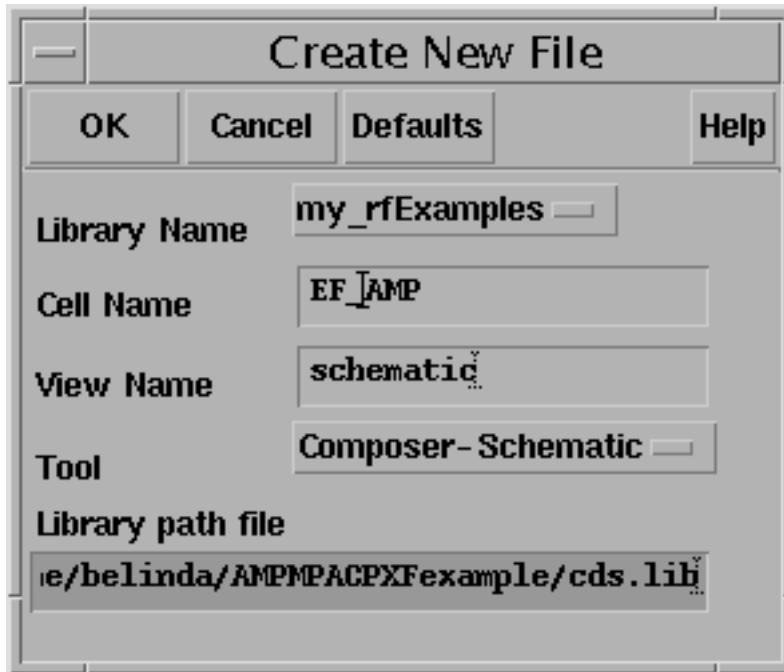
Create a modified copy of the *EF\_example* schematic, *EF\_AMP* and save the *EF\_AMP* schematic in the *my\_rfExamples* library you created. You use the *EF\_AMP* schematic in the Jitter example.

### Create a New Empty Schematic Window

Open the *EF\_example* schematic in a schematic window. Open another empty schematic window, name it *EF\_AMP* and copy the *EF\_example* schematic into the *EF\_AMP* schematic window.

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

The Create New File form appears.



2. In the Create New File form, do the following:

a. Choose *my\_rfExamples* for *Library Name*.

Select *my\_rfExamples*, the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

In the *Cell Name* field, enter *EF\_AMP*.

b. In the *View Name* field, enter *schematic*.

c. In the *Tool* cyclic field, select *Composer-Schematic*.

d. Click *OK*.

A new, empty Schematic window named *EF\_AMP* opens.

### Open and Copy the *EF\_example* Schematic

Open the *EF\_example* schematic and copy it into the empty *EF\_AMP* Schematic window. Then edit *EF\_AMP* before simulating.

1. In the CIW, choose *File – Open*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The Open File form appears.

2. In the Open File form, do the following:

a. Choose *my\_rfExamples* for *Library Name*.

Select the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

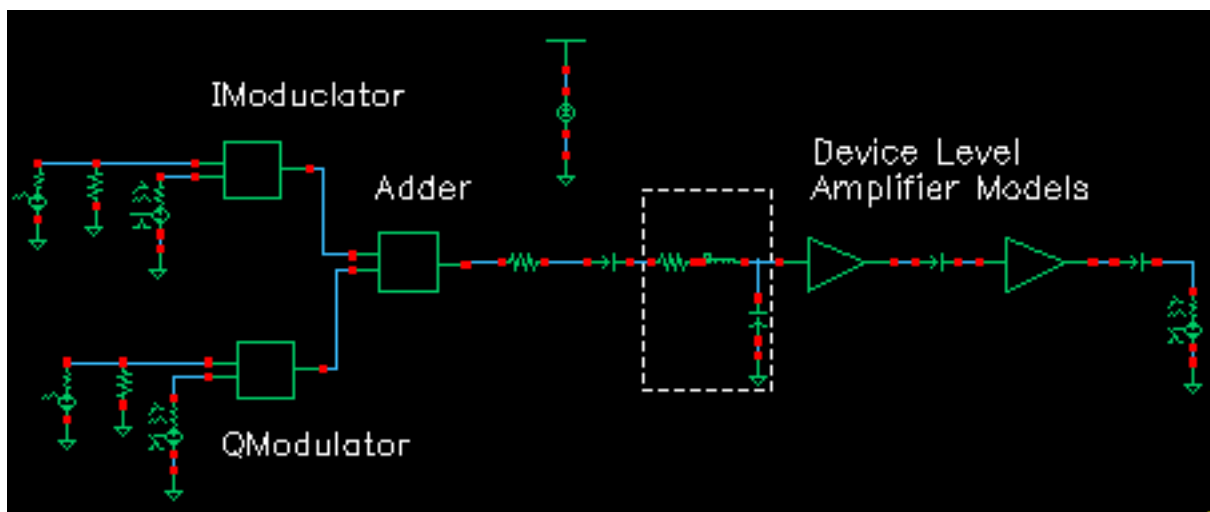
In the *Cell Name* list box, highlight *EF\_example*.

b. Choose *schematic* for *View Name*.

c. Highlight *edit* for *Mode*.

d. Click *OK*.

The Schematic window appears with the *EF\_example* schematic.



3. In the *EF\_example* Schematic window, choose *Edit - Copy* and follow the prompts.

4. Following the prompt at the bottom of the Schematic window,

> point at object to copy

left click and drag to create a box around the entire *EF\_example* circuit.

The *EF\_example* components are highlighted.

5. Following the prompt,

> point at reference point for copy

click inside the outlined elements.

**6.** Following the prompt,

> point at destination point for copy,

move the cursor to drag a copy of the entire circuit into the *EF\_AMP* Schematic window and click there.

The *EF\_AMP* Schematic window now contains a copy of the *EF\_example* schematic.

**7.** In the *EF\_AMP* Schematic window, if necessary, choose *Window - Fit* to scale and center the circuit in the Schematic window.

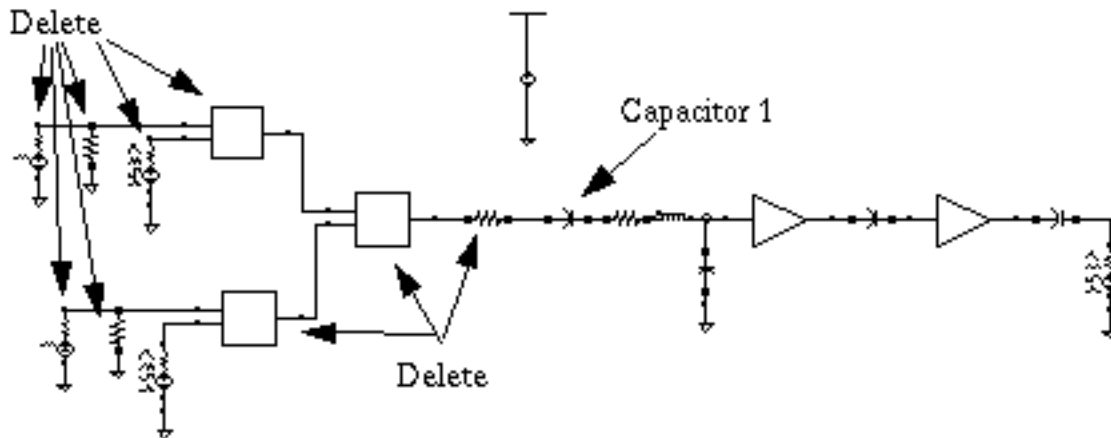
In the *EF\_example* Schematic window, choose *Window - Close*.

The *EF\_example* Schematic window closes.

### Editing the *EF\_AMP* Schematic

In the *EF\_AMP* Schematic window, delete components and their associated connecting wires as shown in Figure 7-20. The final *EF\_AMP* Schematic should look like Figure 7-22 on page 503.

**Figure 7-20 Components and Wires to Delete from the *EF\_LoadPull* Schematic**

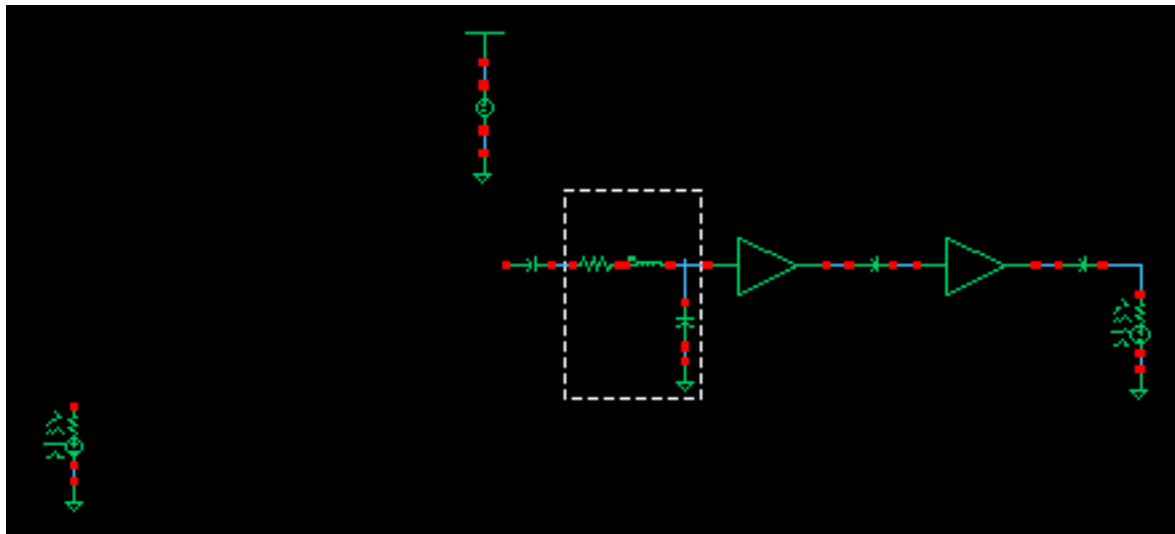


**Note:** If you need assistance with methods for editing the schematic, see the *Virtuoso® Schematic Composer™ User Guide*.

### Delete Components and Wires

1. In the *EF\_AMP* Schematic window, choose *Edit – Delete*.
2. Click a component or wire to delete it.
3. Press the *Esc* key when you are done deleting.
4. After you delete the components and wires, the *EF\_AMP* schematic looks like Figure 7-21.

Figure 7-21 *EF\_AMP* Schematic After Deleting Components

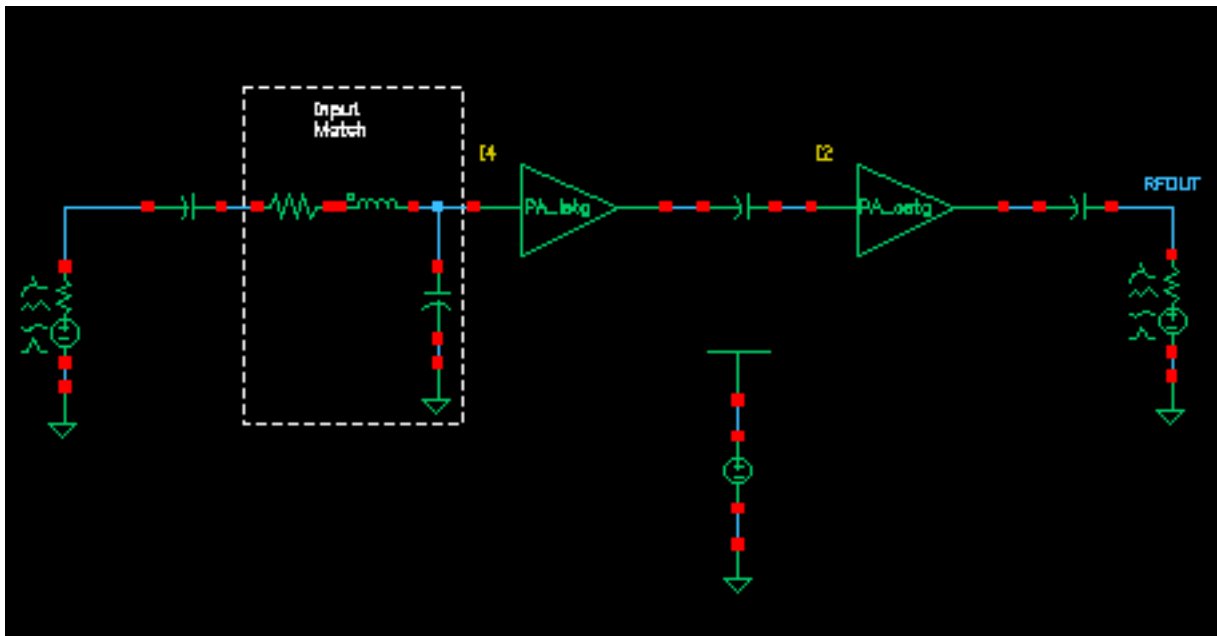


5. In the Schematic window, choose *Edit – Move* to move both *PORT 3* and *VCC* as shown in “The Edited EF\_AMP Schematic” on page 503.
6. To move *PORT 3* closer to capacitor *C1*, click and drag to create a rectangle around *PORT 3* and its ground.  
*PORT 3* and the *Gnd* are highlighted.
7. Click and drag the components to move them close to capacitor *C1* as shown in “The Edited EF\_AMP Schematic” on page 503.
8. To move *VCC* below and between the two device-level amplifier models, click and drag to create a rectangle around *VCC*.  
*VCC* is highlighted in yellow.
9. Click and drag the components to move them below and between the two device-level amplifier models as shown in “The Edited EF\_AMP Schematic” on page 503.

10. Choose *Window - Fit* to center the edited schematic in the window.
11. Wire the Schematic.
12. In the Schematic window, connect *PORT3* to Capacitor *C1*.
  - a. In the Schematic window, choose *Add - Wire (Narrow)* and do the following.
  - b. Click the terminal on *PORT 3* then click the terminal on *C1*.
  - c. Press the Esc key to stop wiring.
13. Choose *Window -- Fit* to center the edited schematic in the window.

The edited schematic looks like the one in Figure 7-22.

Figure 7-22 The Edited EF\_AMP Schematic



### **Edit CDF Properties for PORT3 and RFOUT**

In this section, you edit CDF properties for both *PORT 3* and *RF\_OUT*.

1. In the Schematic window, select *PORT 3*.
2. Choose *Edit – Properties – Objects*.

The Edit Object Properties form appears with information for *PORT 3* displayed.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

3. In the Edit Object Properties form, do the following.

a. Type *fin* for *Frequency 1*.

The completed form looks like this.

CDF Parameter	Value
Resistance	50 Ohms
Reactance	
Port number	1
DC voltage	
Source type	sine
Frequency name 1	fff
Frequency 1	fin Hz
Amplitude 1 (Vpk)	1 v
Amplitude 1 (dBm)	-30
Phase for Sinusoid 1	
Sine DC level	
Delay time	

b. Highlight *Display small signal params*.

c. The small signal parameters section of the form opens up.

d. Type *pac\_dbm* for *PAC Magnitude (dBm)*.

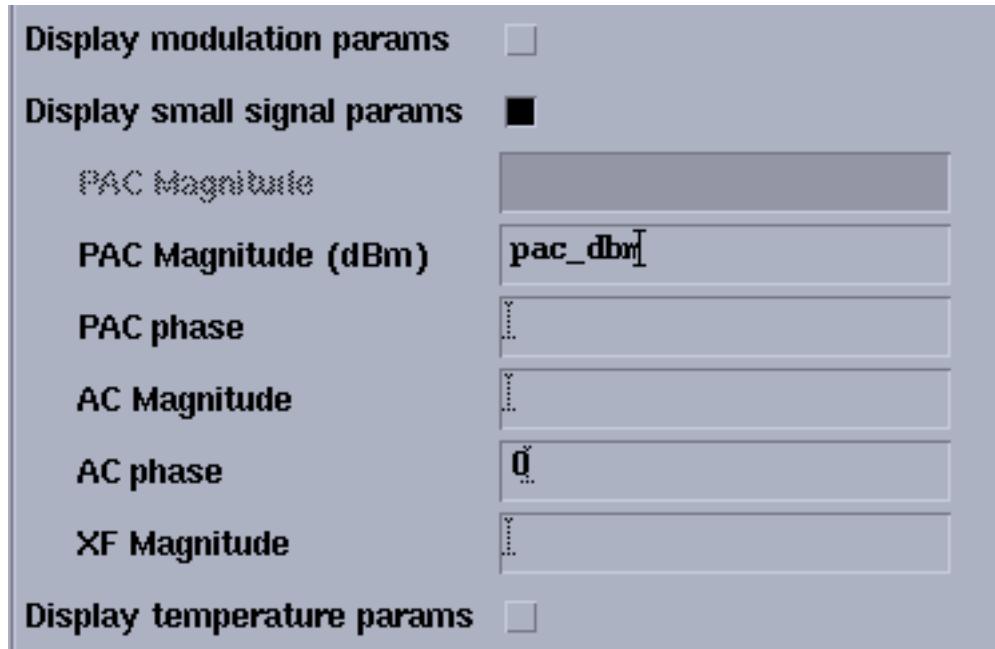


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The completed form looks like this.



The screenshot shows the 'Edit Object Properties' form with the following settings:

- Display modulation params**:
- Display small signal params**:
- PAC Magnitude**: [Empty text box]
- PAC Magnitude (dBm)**: `pac_dbm`
- PAC phase**: [Empty text box]
- AC Magnitude**: [Empty text box]
- AC phase**: `0`
- XF Magnitude**: [Empty text box]
- Display temperature params**:

- e. Click *Apply* in the Edit Object Properties form.
4. In the Schematic window, select *RF\_OUT*.  
The Edit Object Properties form changes to display data for *RF\_OUT*.
5. In the Edit Object Properties form, do the following.
  - a. Highlight *Display small signal params*.  
The small signal parameters section of the form opens up.
  - b. Type `pxfout_mag` for *XF Magnitude*.

The completed form looks like this.

**Display small signal params**

**PAC Magnitude**

**PAC Magnitude (dBm)**

**PAC phase**

**AC Magnitude**

**AC phase**

**XF Magnitude**

**Display temperature params**

c. Click *OK* in the Edit Object Properties form.

6. In the Schematic window, choose *Design - Check and Save*.

## Setting Up the EF\_AMP Circuit

Before you start, perform the setup procedures described in [Chapter 3](#).

Set up and run the necessary PSS, modulated PAC and modulated PXF analyses on the *EF\_AMP* schematic.

Compute transfer functions and gain measurements and display the resulting information with the Direct Plot form.

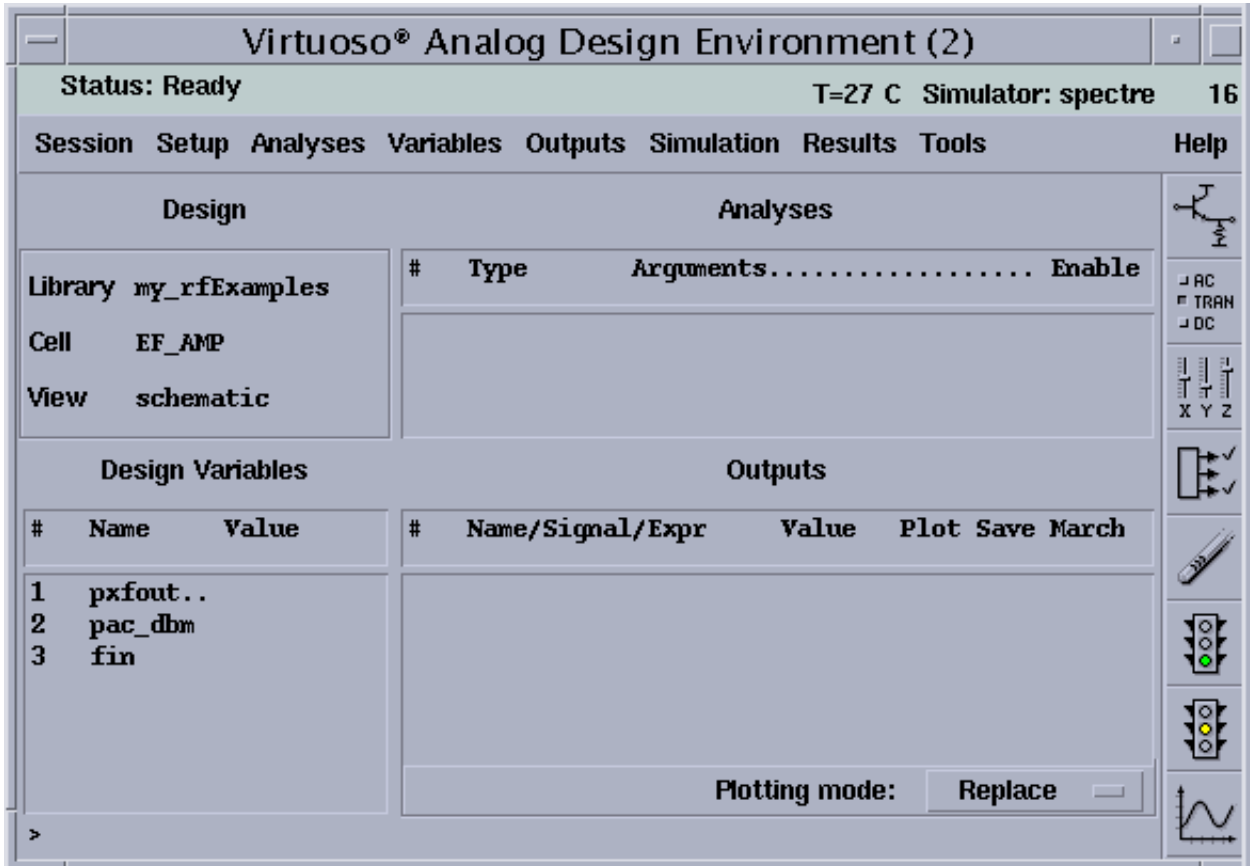
## Opening the Simulation Window for the EF\_AMP Circuit

1. In the *EF\_AMP* Schematic window, choose *Tools – Analog Environment*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

The Simulation window opens.



2. In the Simulation window, choose *Variables – Copy From Cellview*.

The variable names display in the Design Variables area of the Simulation window.

Edit the Variable Values

- Following the directions in “Editing Design Variable Values” on page 187, edit the variables to have the following values.

```

pxfout_mag    1
pac_dbm       0
fin           1G
    
```

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

Check the Design Variables area in the Simulation window to display the variable values.

Design Variables		
#	Name	Value
1	pxfout..	1
2	pac_dbm	0
3	fin	1G

You can also use *Tools – Analog Environment – Simulation* in the CIW to open the Simulation window without opening the design. You can open the design later by choosing *Setup – Design* in the Simulation window and choosing *EF\_example* in the Choosing Design form.

### Setting up the Model Libraries for the EF\_AMP Circuit

1. In the Simulation window, choose *Setup - Model Libraries*.

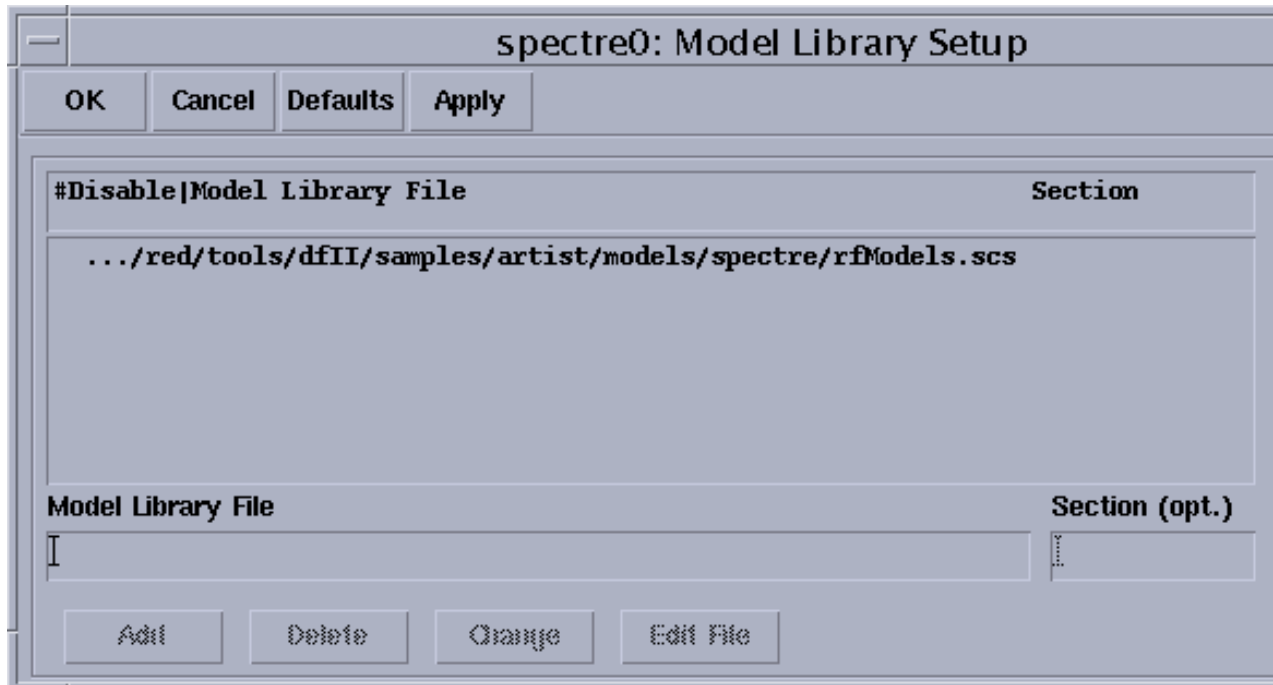
The Model Library Setup form appears.

2. In the *Model Library File* field, type the full path to the model file including the file name. *CDSHOME* is the installation directory for the Cadence software.

`<CDSHOME>/tools/dfII/samples/artist/models/spectre/rfModels.scs`

3. Click *Add*.

The Model Library Setup form looks like the following.

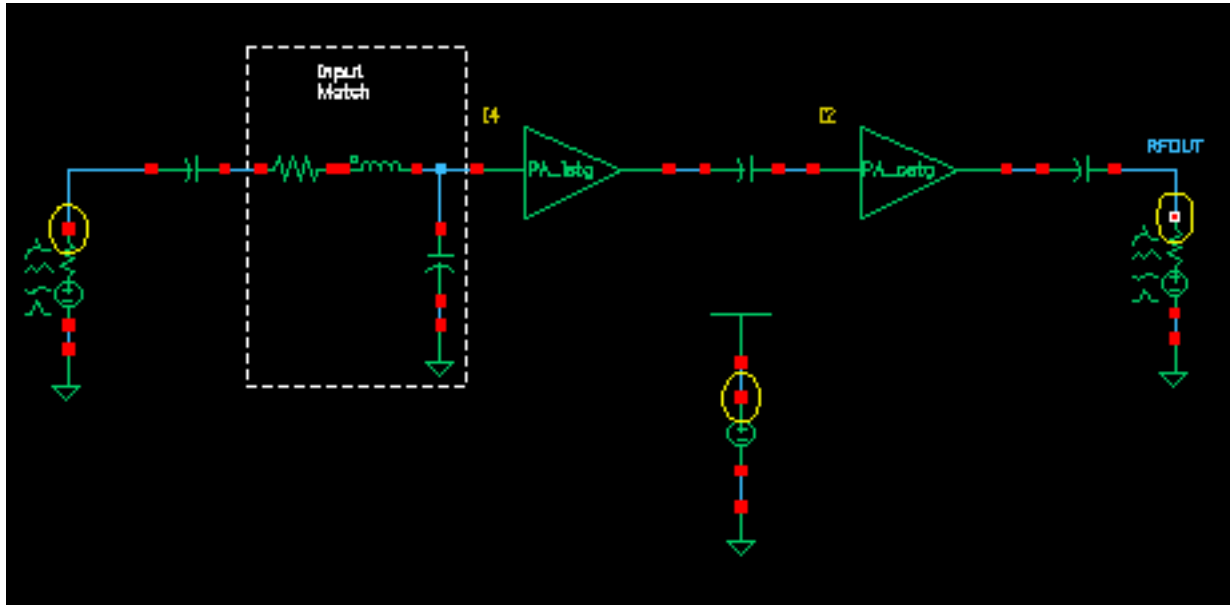


4. Click *OK*.

### Selecting Outputs To Save

1. In the Simulation window, choose *Outputs – To Be Saved – Select on Schematic*.
2. In the Schematic window, click the terminals that are circled in Figure [7-23](#).

Figure 7-23 Outputs to Save in the EF\_AMP Schematic



After you click a terminal, it is circled in the schematic. The selected outputs are also displayed in the Schematic window Outputs area as shown in Figure 7-24.

Figure 7-24 Outputs Area in Simulation Window

Outputs					
#	Name/Signal/Expr	Value	Plot	Save	March
1	PORT3/PLUS		no	yes	no
2	VCC/PLUS		no	yes	no
3	RF_OUT/PLUS		no	yes	no

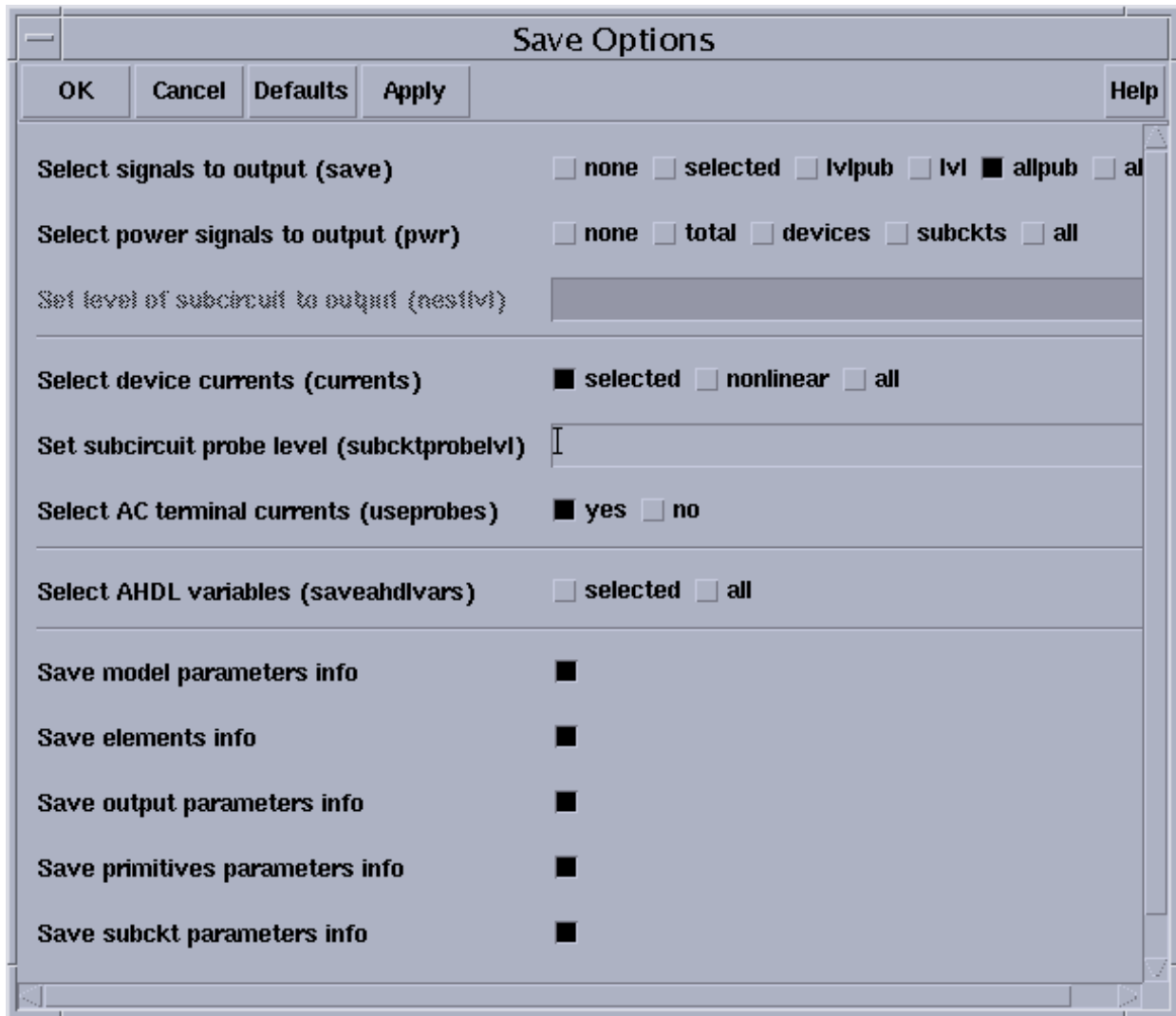
Plotting mode:

- In the Simulation window, choose *Outputs-Save All*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

4. The Save Options form displays.



5. Set the following values

<i>save</i>	<i>allpub</i>
<i>currents</i>	<i>selected</i>
<i>useprobes</i>	<i>yes</i>

6. Click OK.

### **Setting Up and Running the PSS, PAC Modulated, and PXF Modulated Analyses**

- In the Simulation window, choose *Analyses – Choose*.

The Choosing Analyses form appears.

#### ***Setting up the PSS Analysis***

1. In the Choosing Analyses form, highlight *pss*.

The form changes to display options for PSS simulation.

2. Highlight the *Auto Calculate* button to automatically calculate and enter the value 1G in the *Beat Frequency* field.
3. Choose *Number of harmonics* for *Output harmonics* and type 10 in the adjacent field.



# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmitters

The top of the PSS analysis form looks like this.

**Periodic Steady State Analysis**

Engine       Shooting    Flexible Balance

Fundamental Tones					
#	Name	Expr	Value	Signal	SrcId
1	fff	1G	1G	Large	PORT3
2	fff	1G	1G	Large	PORT4

Large

Clear/Add    Delete    Update From Schematic

Beat Frequency    1G    Auto Calculate

Beat Period

Output harmonics

Number of harmonics    10

4. Highlight *moderate* for Accuracy Defaults (*errpreset*).
5. Verify that *Enabled* is highlighted in the PSS Choosing Analyses form.

The bottom of the PSS Choosing Analyses form looks like this.

The screenshot shows a dialog box with the following elements:

- Accuracy Defaults (empreset)**
  - conservative  moderate  liberal
- Additional Time for Stabilization (tstab)** [text input field]
- Save Initial Transient Results (saveinit)**  no  yes
- Oscillator**
- Sweep**
- Enabled**
- Options...** button

6. Click the *Options* button to display the Options form for the PSS analysis.
7. In the Options form, scroll to the *Output Parameters* section, highlight *all* for *save*. The *Output Parameters* section of the Options form looks like the following.

The screenshot shows the **OUTPUT PARAMETERS** section of the Options form. The **save** field has the following options:

- selected
- lvlpub
- lvl
- allpub
- all

8. In the PSS Options form, click *OK*.
9. In the PSS Analysis form, click *Apply*.

### **Setting up the PAC Modulated Analysis**

1. In the Choosing Analyses form, highlight *pac*.
2. The form changes to display options for PAC simulation.
3. For *Sweep*type, select *Relative* in the cyclic field and type 1 in the *Relative Harmonic* field.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

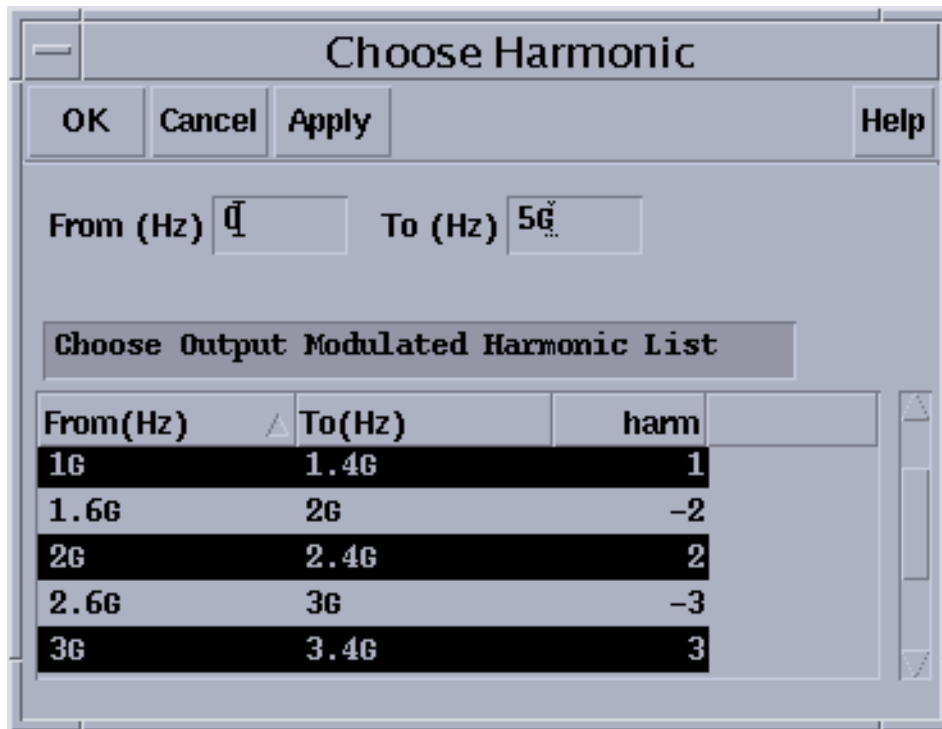
4. For *Frequency Sweep Range*, select *Start-Stop* in the cyclic field. Type 100 in the *Start* field and 400M in the *Stop* field.
5. For *Sweep Type*, select *Logarithmic* in the cyclic field, highlight *Points Per Decade* and type 20 in the field.

The top of the PAC analysis form looks like this.

The screenshot shows the 'Periodic AC Analysis' dialog box. At the top, 'PSS Beat Frequency (Hz)' is set to 16. Below this, there are two rows of controls. The first row has 'Sweeptype' set to 'relative' and 'Relative Harmonic' set to 1. The second row is for 'Frequency Sweep Range (Hz)', with 'Start-Stop' selected, 'Start' set to 100, and 'Stop' set to 400M. The third row is for 'Sweep Type', with 'Logarithmic' selected, 'Points Per Decade' selected (indicated by a filled radio button), and the value set to 20. The 'Number of Steps' option is unselected (indicated by an empty radio button). At the bottom, there is an 'Add Specific Points' checkbox which is unchecked.

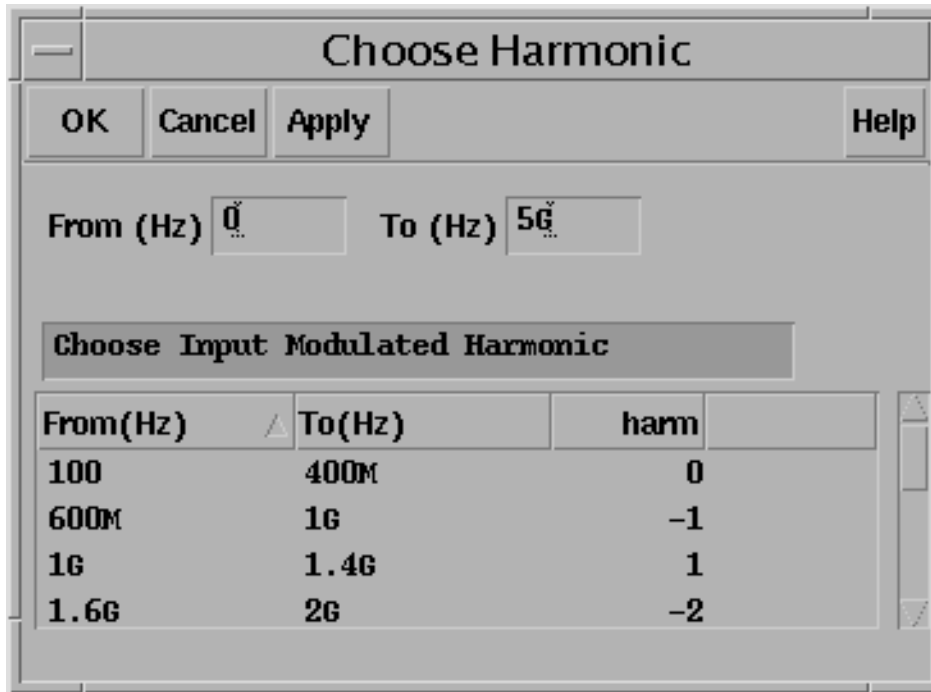
6. For *Sidebands*, select *Maximum sideband* in the cyclic field. Type 3 in the field.
7. In the *Specialized Analyses* cyclic field select *Modulated* to open the modulated analysis section of the PAC Choosing Analyses form.
8. For *Input Type* choose *SSB/AM/PM* in the cyclic field.

9. For *Output Modulated Harmonic List*, press *Choose* to display the Choose Harmonic form.



10. Scroll the list of harmonics and select harmonics with indexes 1, 2 and 3. Click to highlight the first harmonic. Press and hold down the *Control* key while you click to select harmonics 2 and 3. Then click OK.
- 1, 2, and 3 display in the *Output Modulated Harmonic List* field.
11. You can also simply type the values, separated by spaces, in the *Output Modulated Harmonic List* field.

12. For *Input Modulated Harmonic*, press *Choose* to display the Choose Harmonic form.



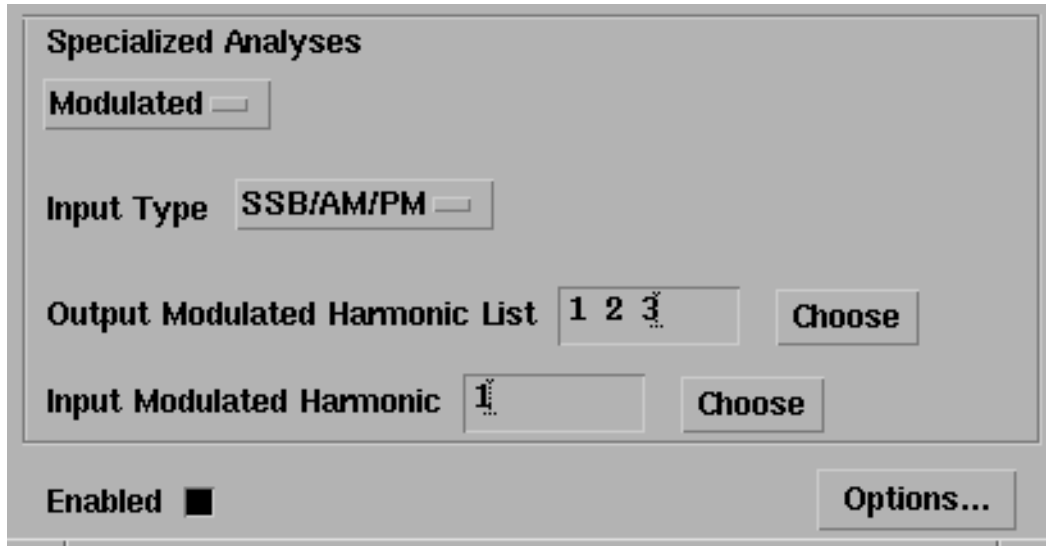
13. Scroll the list of harmonics and select the harmonic with index of 1. Click to highlight the harmonic 1. Then click *OK*.

1 displays in the *Input Modulated Harmonic* field.

You can also simply type the value in the *Input Modulated Harmonic* field.

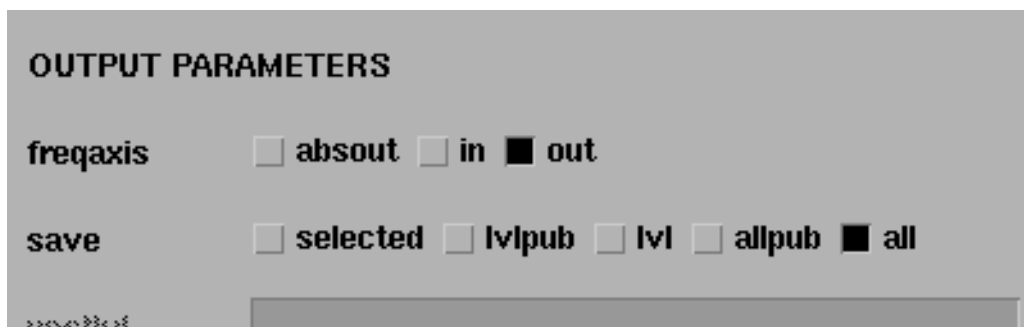
14. Verify that *Enabled* is highlighted in the PAC Choosing Analyses form and click *Apply*.

The bottom of the PAC analysis form looks like this.



15. Click the *Options* button to display the Options form for the PAC analysis.
16. In the *Output Parameters* section of the Options form, highlight, *out* for *freqaxis* and *all* for *save*.

The Output *Parameters* section of the Options form looks like the following.



17. In the PAC Options form, click *OK*.
18. In the PAC Analysis form, click *Apply*.

### **Setting up the PXF Modulated Analysis**

1. In the Choosing Analyses form, highlight *pxf*.

The form changes to display options for PXF analysis.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

2. For *Sweeptype*, select *relative* in the cyclic field and type 1 in the *Relative Harmonic* field.
3. For *Frequency Sweep Range*, select *Start-Stop* in the cyclic field. Type 100 in the *Start* field and 400M in the *Stop* field.
4. For *Sweep Type*, select *Logarithmic* in the cyclic field, highlight *Points Per Decade* and type 20 in the field.

The top of the PXF analysis form looks like this.

The screenshot shows the 'Periodic XF Analysis' dialog box. At the top, 'PSS Beat Frequency (Hz)' is set to '1G'. Below this, 'Sweeptype' is set to 'relative' and 'Relative Harmonic' is set to '1'. The 'Frequency Sweep Range (Hz)' section shows 'Start-Stop' selected, with 'Start' at '100' and 'Stop' at '400M'. Under 'Sweep Type', 'Logarithmic' is selected, and 'Points Per Decade' is chosen with a value of '20'. The 'Add Specific Points' checkbox is unchecked. The 'Sidebands' section shows 'Maximum sideband' selected with a value of '3'.

5. For *Sidebands*, select *Maximum sideband* in the cyclic field and type 3 in the field.
6. For *Output*, highlight *probe*. Click *Select* and follow the prompt at the bottom of the Schematic window

Select output probe instance...

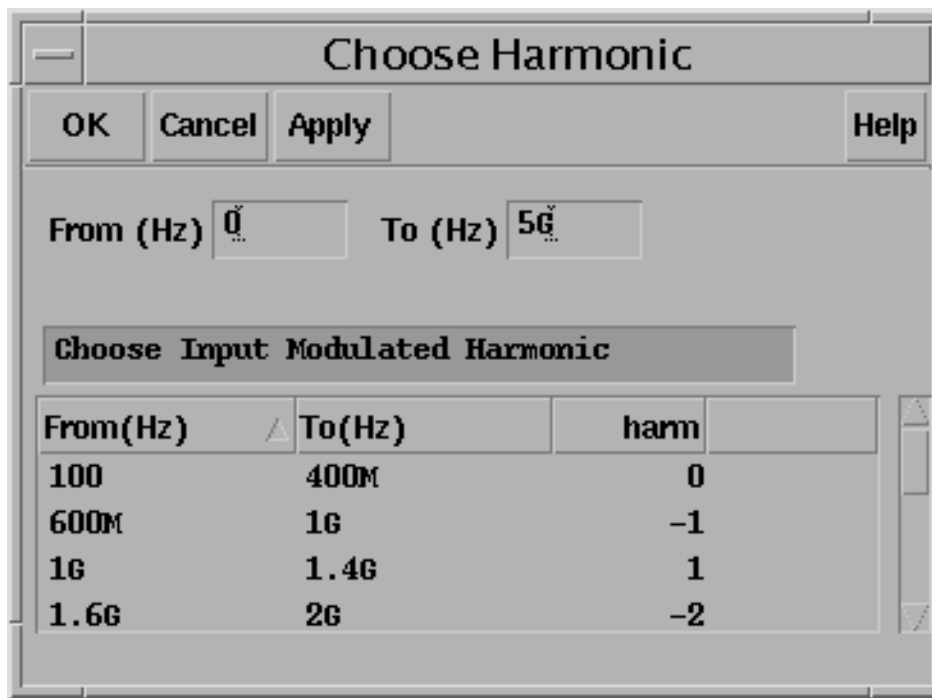
and select the RF port on the right side of the schematic.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

/RF\_OUT displays in the *Output Probe Instance* field.

7. Highlight *Modulated Analysis* to open the modulated analysis section of the PXF Choosing Analyses form.
8. For *Output Type* choose *SSB/AM/PM* in the cyclic field.
9. For *Input Modulated Harmonic List*, press *Choose* to display the Choose Harmonic form.



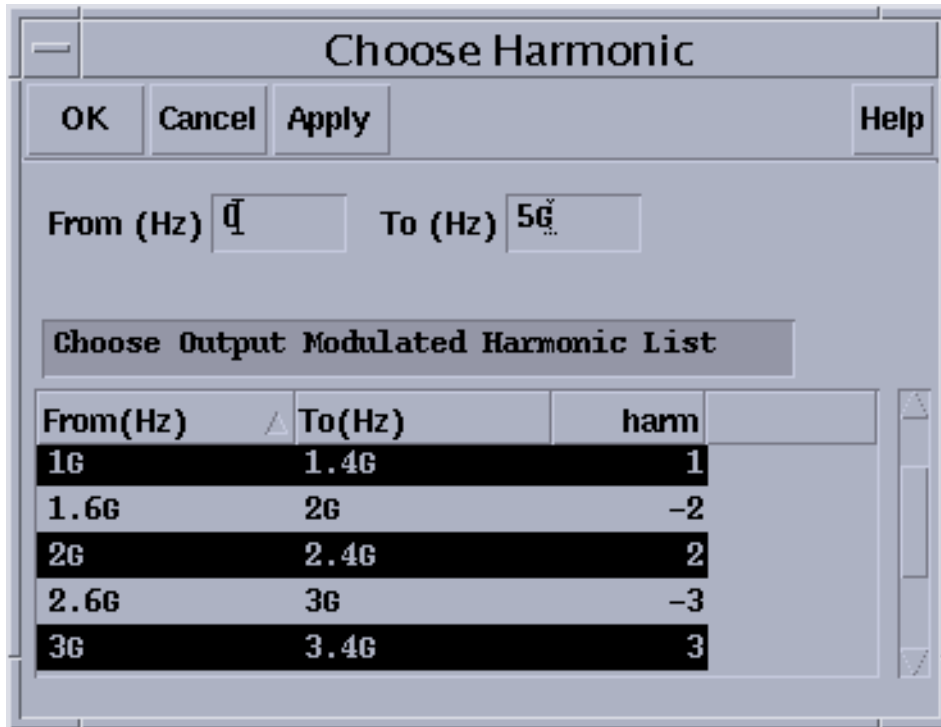
10. Scroll the list of harmonics and select the harmonic with index of 1. Click to highlight the harmonic 1. Then click OK.

1 displays in the *Input Modulated Harmonic List* field.

You can also simply type the value in the *Input Modulated Harmonic List* field.



11. For *Output Modulated Harmonic*, press *Choose* to display the Choose Harmonic form.



12. Scroll the list of harmonics and select the harmonic with index of 1. Click to highlight the harmonic 1. Then click OK.

1 displays in the *Output Modulated Harmonic* field.

You can also simply type the value in the *Input Modulated Harmonic* field.

The bottom of the PXF analysis form looks like this.

The screenshot shows a software interface for configuring a PXF analysis. It is divided into three main sections:

- Output:** Contains two radio buttons: "voltage" (unselected) and "probe" (selected). To the right is a text field labeled "Output Probe Instance" containing the text "/RF\_OUT", followed by a "Select" button.
- Modulated Analysis:** Contains a dropdown menu for "Output Type" set to "SSB/AM/PM". Below this are two rows, each with a text field containing "1" and a "Choose" button. The first row is labeled "Input Modulated Harmonic List" and the second is labeled "Output Modulated Harmonic".
- Enabled:** Contains a checked checkbox labeled "Enabled" and an "Options..." button.

13. Verify that *Enabled* is highlighted in the PXF Choosing Analyses form.
14. Click the *Options* button to display the Options form for the PXF analysis.
15. In the *Output Parameters* section of the Options form *in* for *freqaxis* and *all* for *save*. *sources* is already highlighted for *stimuli* because it is the default.

The *Output Parameters* section of the Options form looks like the following.

The screenshot shows the "OUTPUT PARAMETERS" section of the Options form. It consists of three rows of radio button options:

- stimuli:**  **sources**  **nodes\_and\_terminals**
- freqaxis:**  **absin**  **in**  **out**
- save:**  **selected**  **lvlpub**  **lvl**  **allpub**  **all**

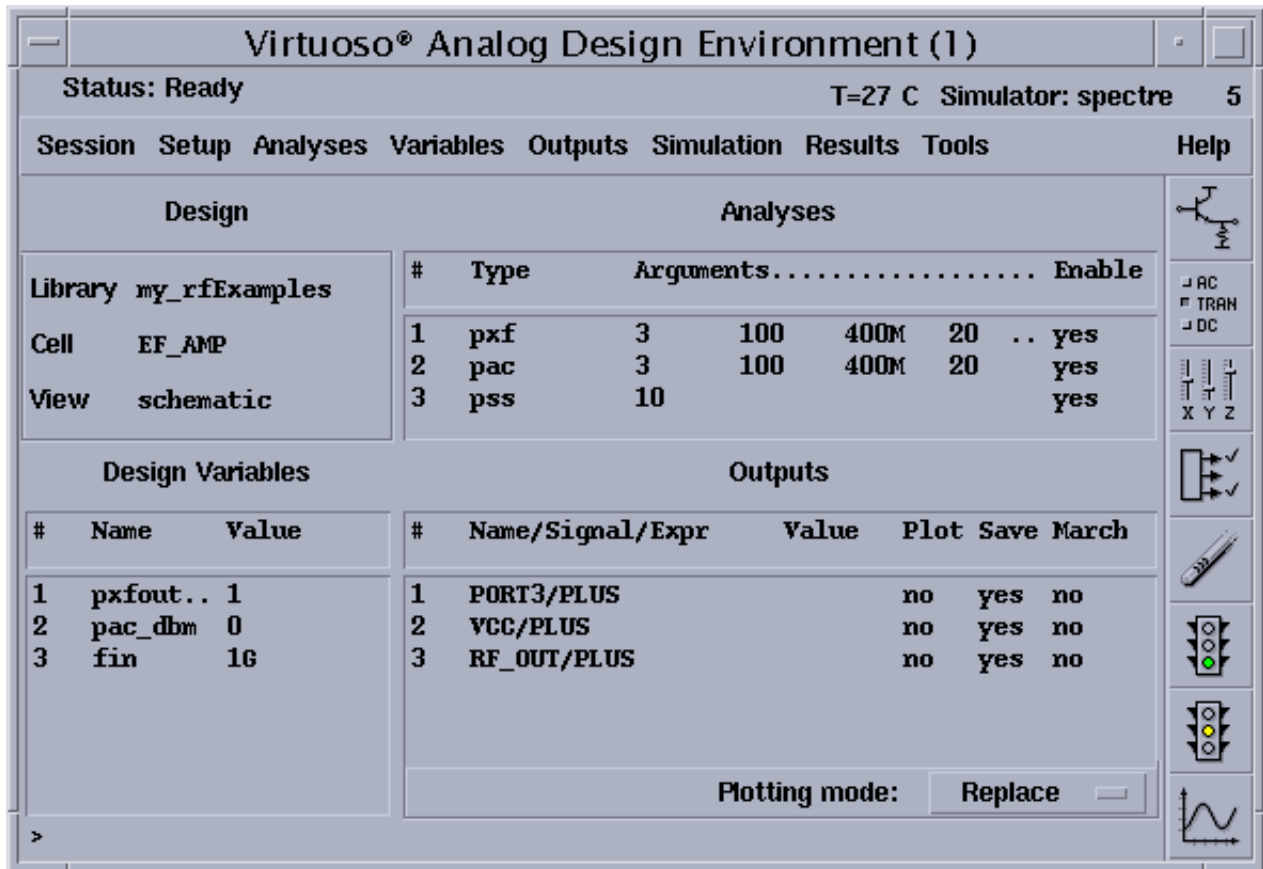
16. Click OK in the PXF Options form.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

17. Click OK in the PXF Choosing Analyses form.

The Choosing Analyses form closes. The Simulation window displays the analysis information you have set up.



## Running the Simulations

1. To run the simulations, choose *Simulation – Netlist and Run* in the Simulation window.  
The output log file appears and displays information about the simulation as it runs.
2. Look in the CIW for a message that says the simulation completed successfully.

## Plotting and Calculating PAC Modulated Results

- Choose *Results – Direct Plot – Main Form* in the Simulation window.  
The Direct Plot form appears.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

In the Direct Plot form, do the following:

1. Highlight *Replace* for *Plot Mode*.
2. Highlight *pac modulated* for *Analysis*.

The form changes to reflect the modulated PAC analysis.

3. In the *Input cyclic* field, select *AM*.
4. In the *Modulated Input Harmonic cyclic* field, select *1G 1*. This is the only available choice.

The *USB* field displays 1G -- 1.4G.

5. In the *Output cyclic* field, select *AM*.
6. In the *Modulated Output Harmonic cyclic* field, select *1G 1*.

The *USB* field displays 1G -- 1.4G.

7. Highlight *Voltage* for *Function*.
8. Highlight *peak* for *Signal Level*.
9. Highlight *dB20* for *Modifier*.

10. The Direct Plot form displays as follows.

The image shows a dialog box titled "Direct Plot Form". At the top, there are buttons for "OK", "Cancel", and "Help". Below these is a "Plotting Mode" section with a "Replace" button. The "Analysis" section contains five radio buttons: "pss", "pac", "pxf", "pac modulated" (which is selected), and "pxf modulated". The "Input" and "Output" sections each have a dropdown menu set to "AM". Below these are "Modulated Input Harmonic" and "Modulated Output Harmonic" sections, both with a dropdown menu set to "1G 1". The "USB" section for both input and output has a text field containing "1G -- 1.4G". The "Function" section has two radio buttons: "Voltage" (selected) and "Current". The "Select" section has a dropdown menu set to "Net". The "Signal Level" section has two radio buttons: "peak" (selected) and "rms". The "Modifier" section has five radio buttons: "Magnitude", "Phase", "dB20" (selected), "Real", and "Imaginary". At the bottom, there is an "Add To Outputs" checkbox (unchecked) and a "Replot" button. A prompt at the very bottom reads "> Select Net on schematic...".

11. Select *Net* in the *Select* cyclic field. Follow the prompt at the bottom of the form

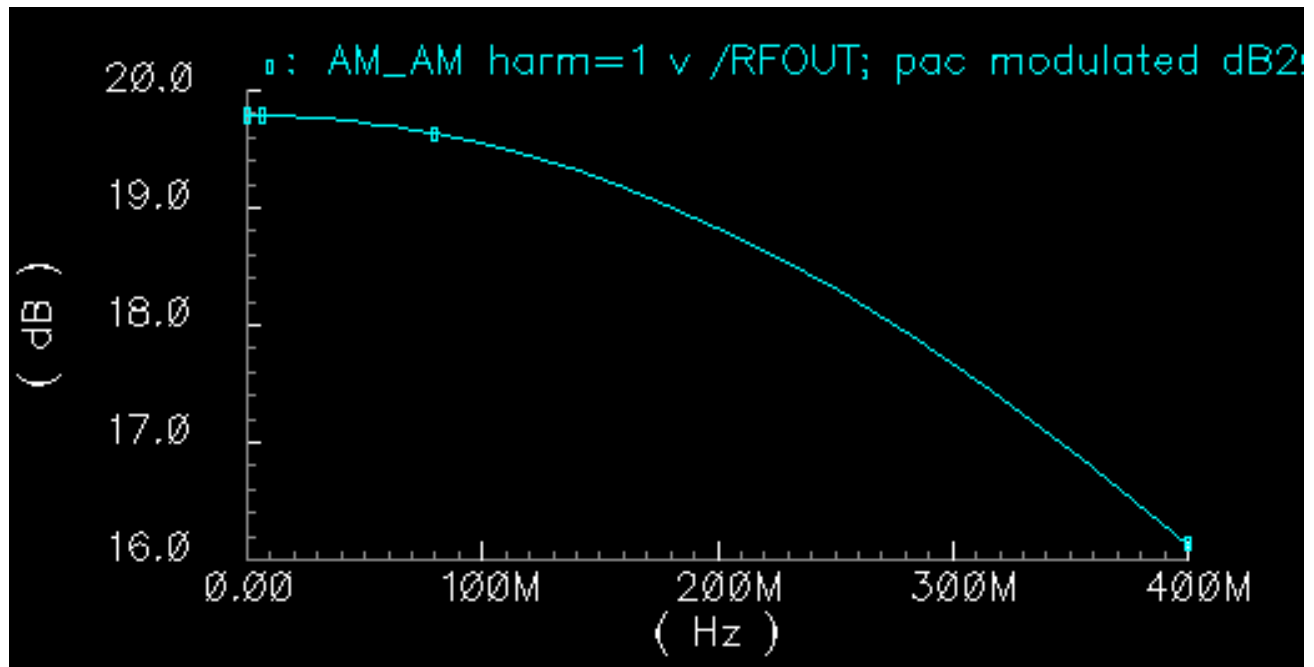
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

Select Net on schematic

Select the net labeled *RFOUT* on the right side of the schematic.

The waveform looks as follows.



To add AM/PM to the plot, make the following changes in the Direct Plot form.

1. Change Plot Mode to Append.
2. Change the *Output* cyclic field to *PM*.
3. In the *Modulated Output Harmonic* cyclic field, select *1G 1*.

The *USB* field displays *1G -- 1.4G*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmitters

4. The Direct Plot form changes as follows.

The image shows a dialog box titled "Direct Plot Form" with the following controls:

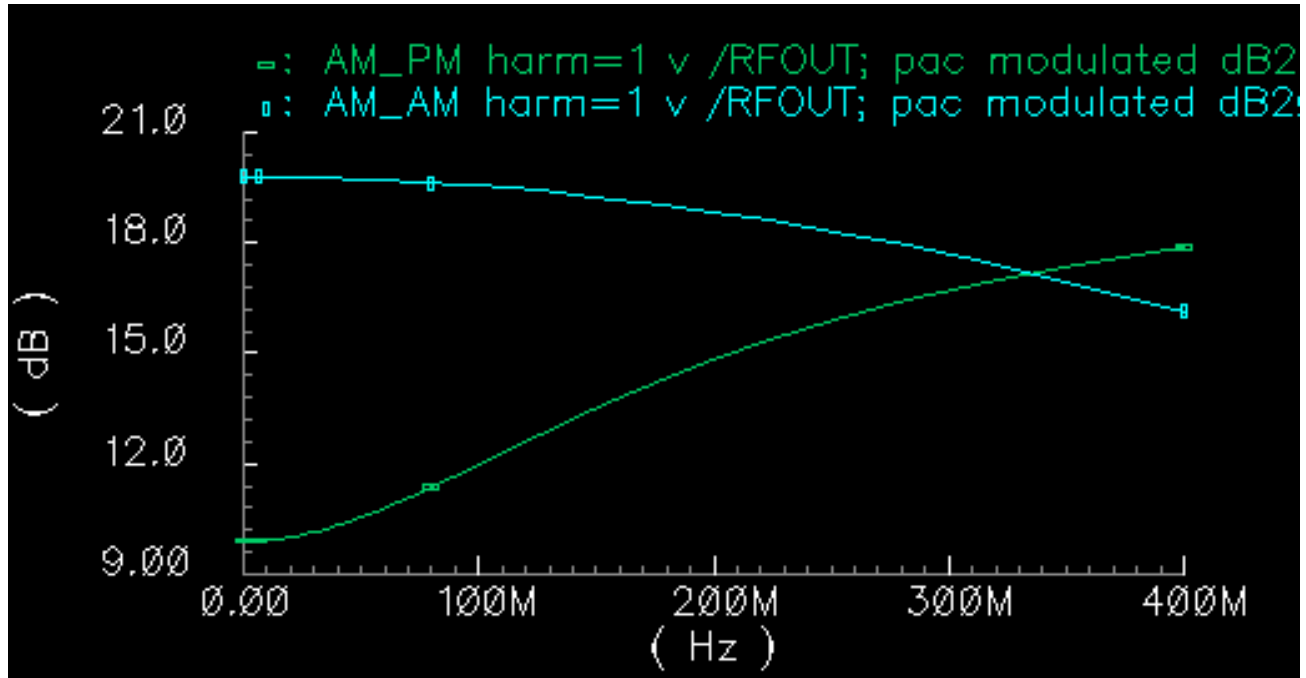
- Buttons: OK, Cancel, Help
- Plotting Mode: Append
- Analysis section:
  - pss
  - pxf
  - pxf modulated
  - pac
  - pac modulated
- Input: AM
- Output: PM
- Modulated Input Harmonic: 1G 1
- Modulated Output Harmonic: 1G 1
- USB: 1G -- 1.4G
- Function section:
  - Voltage
  - Current
- Select: Net
- Signal Level:  peak,  rms
- Modifier section:
  - Magnitude
  - Phase
  - dB20
  - Real
  - Imaginary
- Buttons: Add To Outputs, Replot
- Status bar: > Select Net on schematic...

5. Follow the prompt at the bottom of the form

Select Net on schematic

Select the net labeled *RFOUT* again.

The waveform looks as follows.



### Plotting and Calculating PXF Modulated Results

1. If necessary, choose *Results – Direct Plot – Main Form* in the Simulation window.

The Direct Plot form appears.

2. In the Direct Plot form, highlight *Replace* for *Plot Mode*.
3. Highlight *pxf modulated* for *Analysis*.
4. In the *Input cyclic* field, select *AM*.
5. In the *Modulated Input Harmonic cyclic* field, select *1G 1*. This is the only available choice.

The *USB* field displays *1G -- 1.4G*.

6. In the *Output cyclic* field, select *AM*.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

7. In the *Output Harmonic* cyclic field, select *1G 1*.

The *USB* field displays *1G -- 1.4G*.

8. Highlight *Voltage Gain* for *Function*.

9. Highlight *dB20* for *Modifier*.

10. The Direct Plot form displays as follows.

The image shows a dialog box titled "Direct Plot Form". At the top, there are three buttons: "OK", "Cancel", and "Help". Below the buttons, the "Plot Mode" section has two radio buttons: "Append" (which is selected) and "Replace". The "Analysis" section contains five radio buttons: "pss", "pac", "pxf", "pac modulated", and "pxf modulated" (which is selected). The "Input" and "Output" sections each have a dropdown menu set to "AM". Below these, the "Modulated" sections for "Input Harmonic" and "Output Harmonic" both have a dropdown menu set to "1G 1". The "USB" sections for both "Input" and "Output" have a range of "-2G -- -1.6G". The "Function" section has two radio buttons: "Voltage Gain" (selected) and "Transimpedance". The "Modifier" section has five radio buttons: "Magnitude", "Phase", "dB20" (selected), "Real", and "Imaginary". At the bottom, there is a checkbox for "Add To Outputs" which is unchecked. A prompt at the very bottom reads "> Select Port or Voltage Source on schematic...".

11. Follow the prompt at the bottom of the form

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

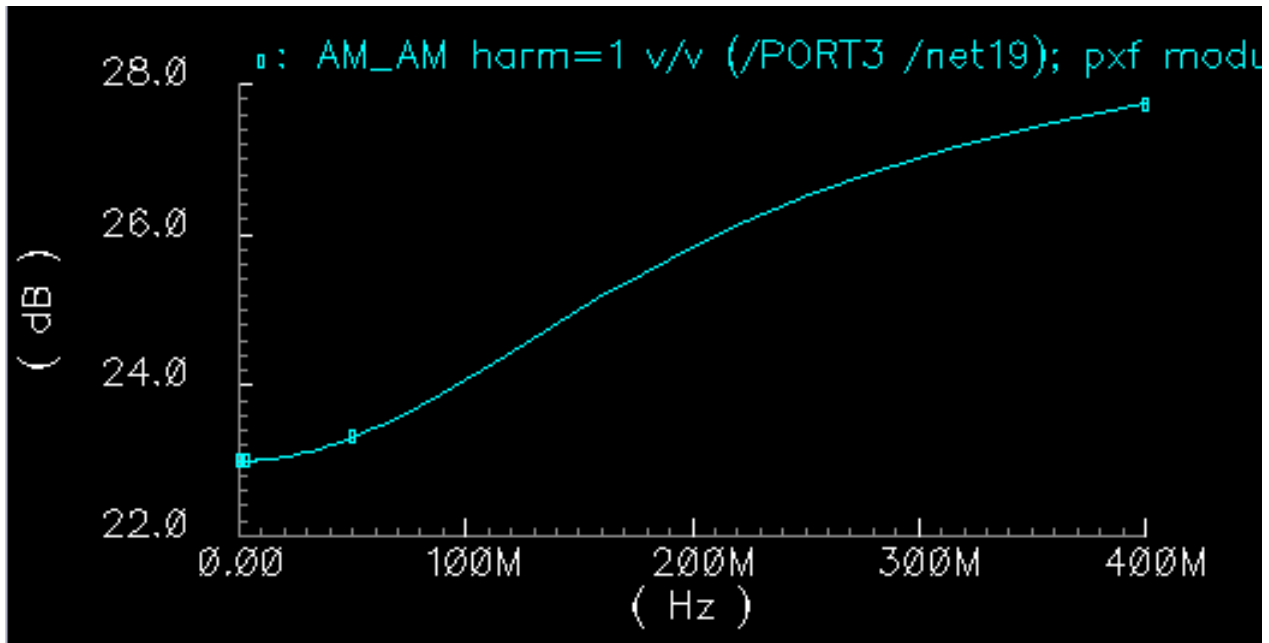
### Modeling Transmitters

---

Select Port or voltage source on schematic

Select the input port labeled *PORT3* on the left side of the schematic.

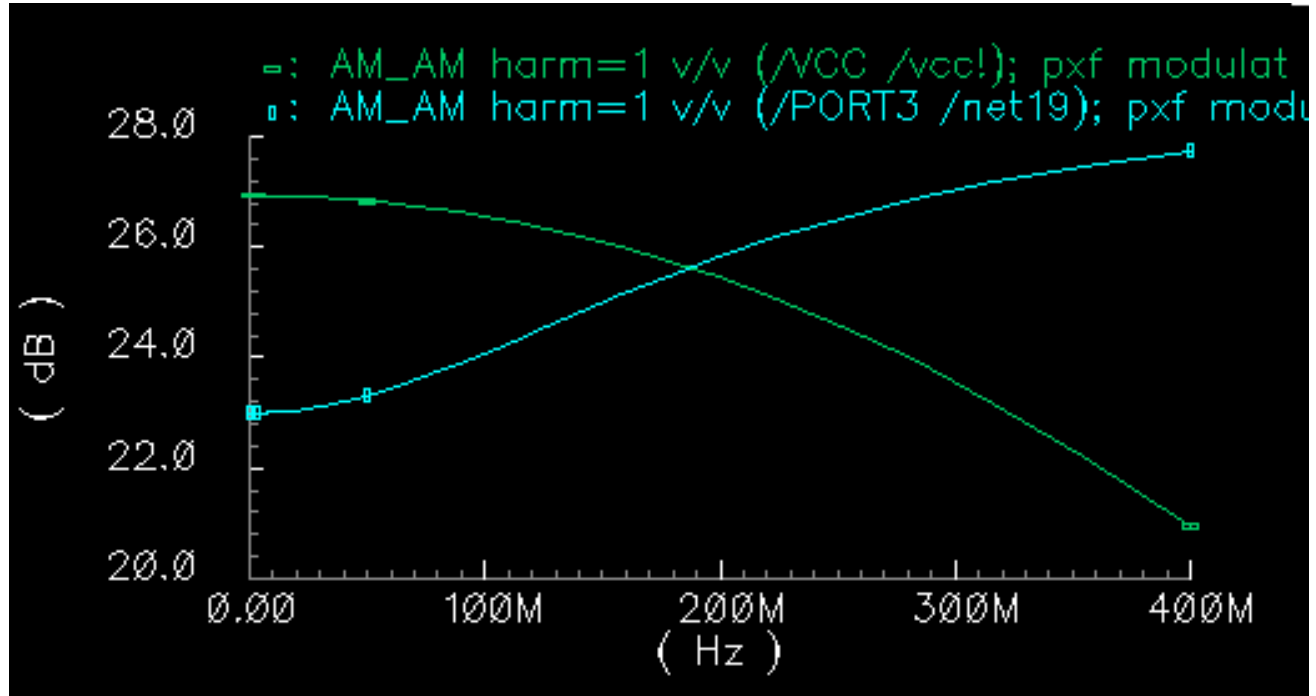
The waveform looks as follows.



To add the VCC voltage source to the plot,

1. Change Plot Mode to Append.
2. Select the voltage source labeled *VCC* in the lower center of the schematic.

The VCC voltage source is added to the plot.



This plot compares two different voltage sources from the schematic.

## Measuring Jitter with PSS and Pnoise Analyses

This section describes jitter measurement.

### Setting Up the EF\_AMP Circuit

Before you start, perform the setup procedures described in [Chapter 3](#).

If you have a copy of the *EF\_AMP* schematic in your *my\_rfExamples* library, simply open it. If you do not have a copy of the *EF\_AMP* schematic, See [“Creating the EF\\_AMP Circuit”](#) on page 498 for information on creating the *EF\_AMP* circuit.

Open the simulation window for the *EF\_AMP* schematic as described in [“Opening the Simulation Window for the EF\\_AMP Circuit”](#) on page 506.

Set up the model libraries as described in [“Setting up the Model Libraries for the EF\\_AMP Circuit”](#) on page 508.

Select Outputs to save as described in “[Selecting Outputs To Save](#)” on page 509.

Set up and run the necessary PSS and modulated Pnoise analyses on the *EF\_AMP* schematic.

Compute transfer functions and gain measurements and display the resulting information with the Direct Plot form.

## **Opening the EF\_AMP Circuit in the Schematic Window**

1. In the CIW, choose *File – Open*.

The Open File form appears.

2. In the Open File form, do the following:

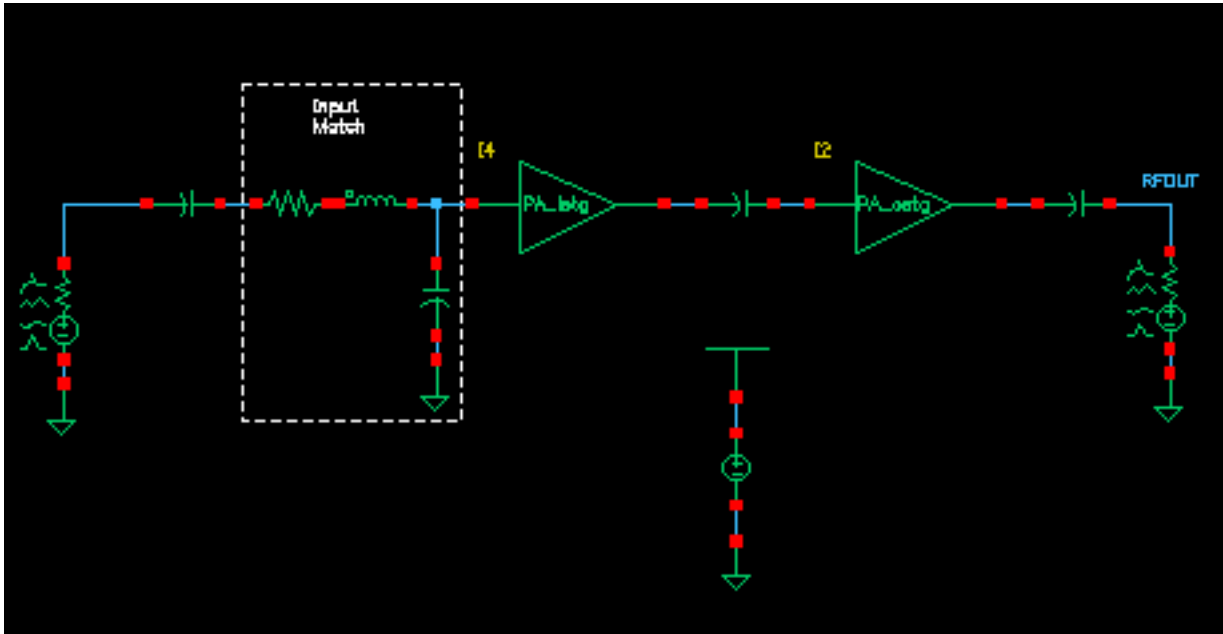
- a. Choose *my\_rfExamples* for *Library Name*.

Select the editable copy of the *rfExamples* library you created following the instructions in [Chapter 3](#).

In the *Cell Name* list box, highlight *EF\_AMP*.

- b. Choose *schematic* for *View Name*.
- c. Highlight *edit* for *Mode*.
- d. Click *OK*.

The Schematic window appears with the *EF\_AMP* schematic.



## Opening the Simulation Window for the EF\_AMP Circuit

1. In the EF\_AMP Schematic window, choose *Tools – Analog Environment*.

The Simulation window opens.

Design			Analyses			
Library	my_rfExamples		#	Type	Arguments.....	Enable
Cell	EF_AMP					
View	schematic					
Design Variables			Outputs			
#	Name	Value	#	Name/Signal/Expr	Value	Plot Save March
1	pxfout..	1				
2	pac_dbm	0				
3	fin	1G				
			Plotting mode: <span style="border: 1px solid black; padding: 2px;">Replace</span> <input type="checkbox"/>			

>

## Setting Up and Running the PSS and Pnoise Analyses

- In the Simulation window, choose *Analyses – Choose*.

The Choosing Analyses form appears.

### Setting up the PSS Analysis

1. In the Choosing Analyses form, highlight *pss*.

The form changes to display options for PSS simulation.

2. Highlight the *Auto Calculate* button to automatically calculate and enter the value 1G in the *Beat Frequency* field.
3. Choose *Number of harmonics* for *Output harmonics* and type 10 in the adjacent field. Output Harmonics are not critical for this analysis.

4. The top of the PSS analysis form looks like this.

**Periodic Steady State Analysis**

Engine  Shooting  Flexible Balance

Fundamental Tones					
#	Name	Expr	Value	Signal	SrcId
1	fff	fin	1G	Large	PORT3

Clear/Add Delete Update From Schematic

Beat Frequency  Beat Period

1G Auto Calculate

5. Highlight *moderate* for Accuracy Defaults (*errpreset*).

6. Verify that *Enabled* is highlighted in the PSS Choosing Analyses form.



The bottom of the PSS Choosing Analyses form looks like this.

Accuracy Defaults (errpreset)  
 conservative  moderate  liberal  
Additional Time for Stabilization (tstab)   
Save Initial Transient Results (saveinit)  no  yes  
Oscillator   
Sweep   
Enabled  Options...

7. Click the *Options* button to display the Options form for the PSS analysis.
8. In the Options form, scroll to the *Output Parameters* section, highlight *all* for save. The *Output Parameters* section of the Options form looks like the following.

OUTPUT PARAMETERS  
save  selected  lvpub  lvl  allpub  all

9. In the PSS Options form, click *OK*.
10. In the PSS Analysis form, click *Apply*.

### Setting up the Pnoise Jitter Analysis

1. In the Choosing Analyses form, highlight *pnoise*.
2. The form changes to display options for pnoise simulation.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

3. For *Sweeptype*, select *Relative* in the cyclic field and type 1 in the *Relative Harmonic* field.
4. For *Frequency Sweep Range*, select *Start-Stop* in the cyclic field. Type 10 in the *Start* field and 500M in the *Stop* field.
5. For *Sweep Type*, select *Logarithmic* in the cyclic field, highlight *Points Per Decade* and type 5 in the field.

The top of the pnoise analysis form looks like this.

The screenshot shows the 'Periodic Noise Analysis' dialog box. It has a title bar 'Periodic Noise Analysis'. Below the title bar, there is a text field 'PSS Beat Frequency (Hz)' with the value '1G'. Below this, there are two rows of controls. The first row has a dropdown menu 'Sweeptype' set to 'relative' and a text field 'Relative Harmonic' with the value '1'. The second row has a dropdown menu 'Frequency Sweep Range (Hz)' set to 'Start-Stop', a text field 'Start' with the value '10', and a text field 'Stop' with the value '500M'. Below these, there is a section for 'Sweep Type' with a dropdown menu set to 'Logarithmic', a radio button selected for 'Points Per Decade' (with a text field '5' next to it), and an unselected radio button for 'Number of Steps'. At the bottom, there is a checkbox 'Add Specific Points' which is unselected.

6. For *Sidebands*, select *Maximum sideband* in the cyclic field. Type 7 in the field.
7. Choose *voltage* in the *Output* cyclic field.

The following message displays in the CIW.

```
Since voltage was selected as the output measurement
technique for the pnoise analysis, spectre will NOT
subtract any load resistance contribution from the Noise
Figure calculation. If this is undesirable, please select
probe as the output measurement technique and select a port.
```

8. Click *Select* for the *Positive Output Node*, and then click the *RFOUT* net in the Schematic window.
9. Leave the *Negative Output Node* field empty.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

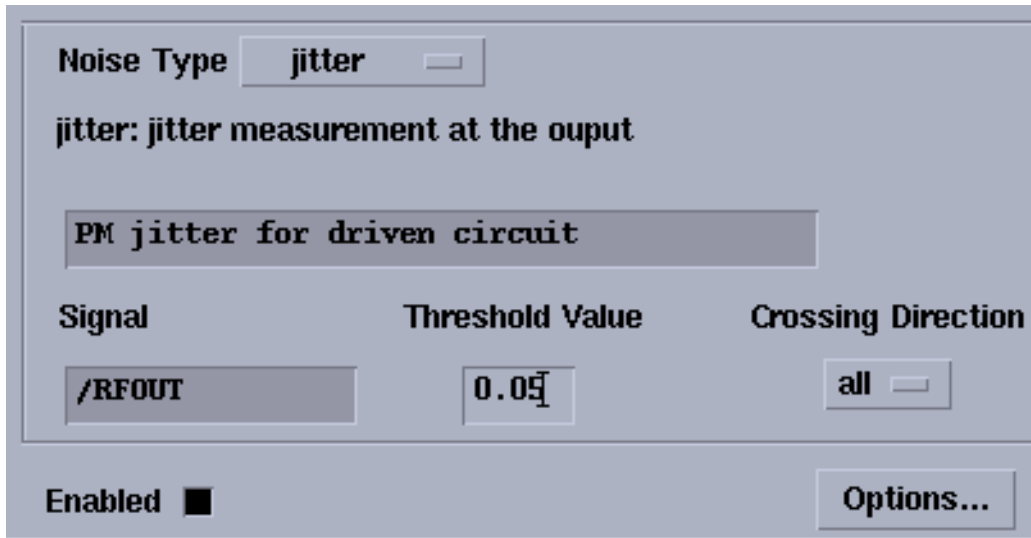
This field defaults to */gnd!* You can set the *Negative Output Node* to a different value by clicking *Select* and then clicking on the output node in the schematic.

10. Choose *port* in the *Input Source* cyclic field.
11. Click *Select* for the *Input Port Source*, and then click *port 3* in the Schematic window.
12. In the *Reference side-band* cyclic field, select *Enter in field* and type 0 in the field.

The image shows a configuration dialog box for the Virtuoso Spectre RF Analysis User Guide. It is divided into four main sections: Sidebands, Output, Input Source, and Reference side-band. Each section contains a cyclic field, a text input field, and a 'Select' button.

Section	Cyclic Field	Text Field	Action
Sidebands	Maximum sideband	/7	
Output	voltage	Positive Output Node: /RFOUT	Select
		Negative Output Node:	Select
Input Source	port	Input Port Source: /PORT3	Select
Reference side-band	Enter in field	0	

13. In the *Noise Type* cyclic field select *Jitter* to open the jitter section of the Pnoise Choosing Analyses form.



The screenshot shows a dialog box titled "Pnoise Choosing Analyses" with the "jitter" option selected in the "Noise Type" field. Below this, the text "jitter: jitter measurement at the output" is displayed. A text box contains "PM jitter for driven circuit". A table with three columns: "Signal", "Threshold Value", and "Crossing Direction" is shown. The "Signal" column has the value "/RFOUT", the "Threshold Value" column has "0.05", and the "Crossing Direction" column has "all". At the bottom left, the "Enabled" checkbox is checked. At the bottom right, there is an "Options..." button.

Signal	Threshold Value	Crossing Direction
/RFOUT	0.05	all

The form indicates that you are measuring PM jitter at the output for a driven circuit and that the output signal is */RFOUT*.

14. Set the *Threshold Value* to 0.05. Select the threshold value based on knowledge of the results of and earlier run of the PSS analysis. Base your selection on knowledge of your circuit.
15. Select *All* in the *Crossing Direction* cyclic field. This captures both up and down crossing points.
- Select *Crossing Direction* equal to *all* cautiously since it directly affects the simulation speed. If the output signal has many transitions, the Pnoise analysis is slowed.
16. Verify that *Enabled* is highlighted in the *Pnoise* Choosing Analyses form and click *Apply*.
17. Click the *Options* button to display the Options form for the *Pnoise* analysis.
18. In the *Output Parameters* section of the Options form, highlight *all* for save.

The Output *Parameters* section of the Options form looks like the following.

**OUTPUT PARAMETERS**

save       selected    lvlpub    lvi    allpub    all

nestlvl     

saveallsidebands    yes    no

19. In the Pnoise Options form, click *OK*.
20. In the Pnoise Analysis form, click *OK*.
21. The Choosing Analyses form closes. The Simulation window displays the analysis information you have set up.

<b>Library</b> my_rfExamples <b>Cell</b> EF_AMP <b>View</b> schematic	<table border="1"> <thead> <tr> <th>#</th> <th>Type</th> <th>Arguments.....</th> <th>End</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>pnoise</td> <td>7 10 500M 5 ..</td> <td>yes</td> </tr> <tr> <td>2</td> <td>pss</td> <td>1G 10</td> <td>yes</td> </tr> </tbody> </table>	#	Type	Arguments.....	End	1	pnoise	7 10 500M 5 ..	yes	2	pss	1G 10	yes																								
#	Type	Arguments.....	End																																		
1	pnoise	7 10 500M 5 ..	yes																																		
2	pss	1G 10	yes																																		
<b>Design Variables</b>	<b>Outputs</b>																																				
<table border="1"> <thead> <tr> <th>#</th> <th>Name</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>pxfout..</td> <td>1</td> </tr> <tr> <td>2</td> <td>pac_dbm</td> <td>0</td> </tr> <tr> <td>3</td> <td>fin</td> <td>1G</td> </tr> </tbody> </table>	#	Name	Value	1	pxfout..	1	2	pac_dbm	0	3	fin	1G	<table border="1"> <thead> <tr> <th>#</th> <th>Name/Signal/Expr</th> <th>Value</th> <th>Plot</th> <th>Save</th> <th>Max</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>PORT3/PLUS</td> <td></td> <td>no</td> <td>yes</td> <td>no</td> </tr> <tr> <td>2</td> <td>VCC/PLUS</td> <td></td> <td>no</td> <td>yes</td> <td>no</td> </tr> <tr> <td>3</td> <td>RF_OUT/PLUS</td> <td></td> <td>no</td> <td>yes</td> <td>no</td> </tr> </tbody> </table>	#	Name/Signal/Expr	Value	Plot	Save	Max	1	PORT3/PLUS		no	yes	no	2	VCC/PLUS		no	yes	no	3	RF_OUT/PLUS		no	yes	no
#	Name	Value																																			
1	pxfout..	1																																			
2	pac_dbm	0																																			
3	fin	1G																																			
#	Name/Signal/Expr	Value	Plot	Save	Max																																
1	PORT3/PLUS		no	yes	no																																
2	VCC/PLUS		no	yes	no																																
3	RF_OUT/PLUS		no	yes	no																																

## Running the Simulations

1. To run the simulations, choose *Simulation – Netlist and Run* in the Simulation window.  
The output log file appears and displays information about the simulation as it runs.

Look in the CIW for a message that says the simulation completed successfully.

## Plotting the Jitter Measurement

- Choose *Results – Direct Plot – Main Form* in the Simulation window.

The Direct Plot form appears.

### Plot the Upward Transition and Event Time

In the Direct Plot form, plot the upward transition.

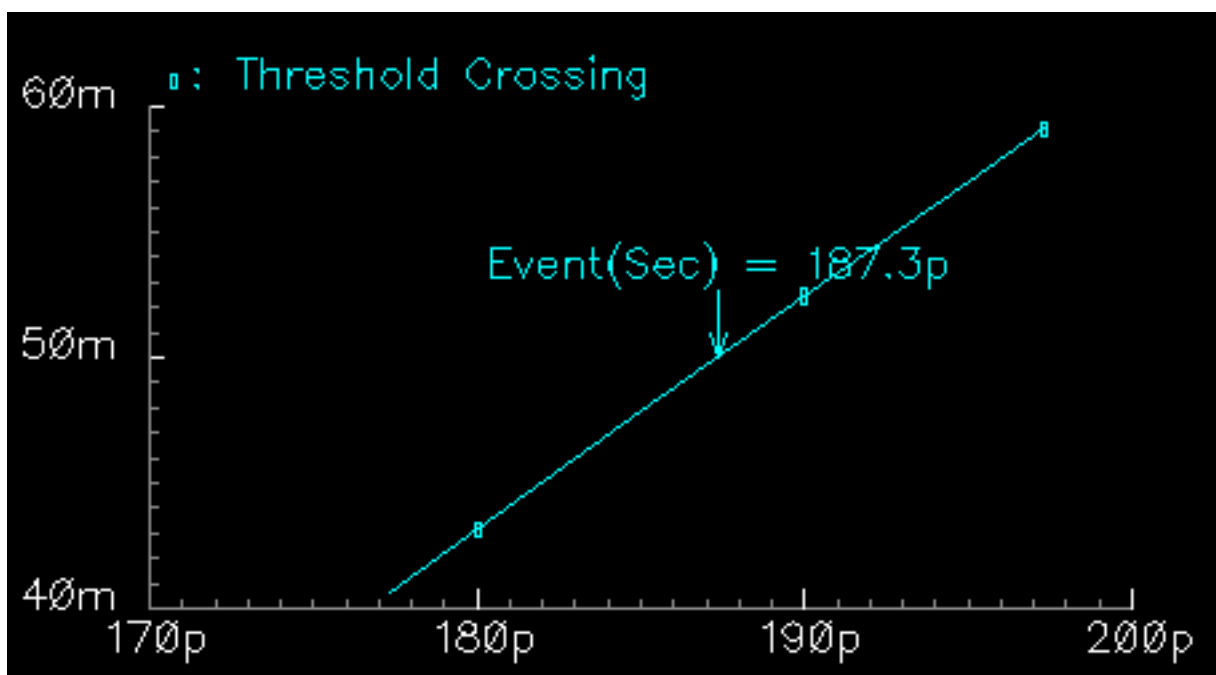
Highlight *Replace* for *Plot Mode*.

1. Highlight *pnoise jitter* for *Analysis*.

The form changes to reflect the pnoise jitter analysis.

2. Highlight *Threshold Xing* for *Function*.
3. In the *Signal Level* cyclic field select *187.3p*.
4. Click *Plot*.

The up transition and its location display.



### Plot the Downward Transition and Event Time

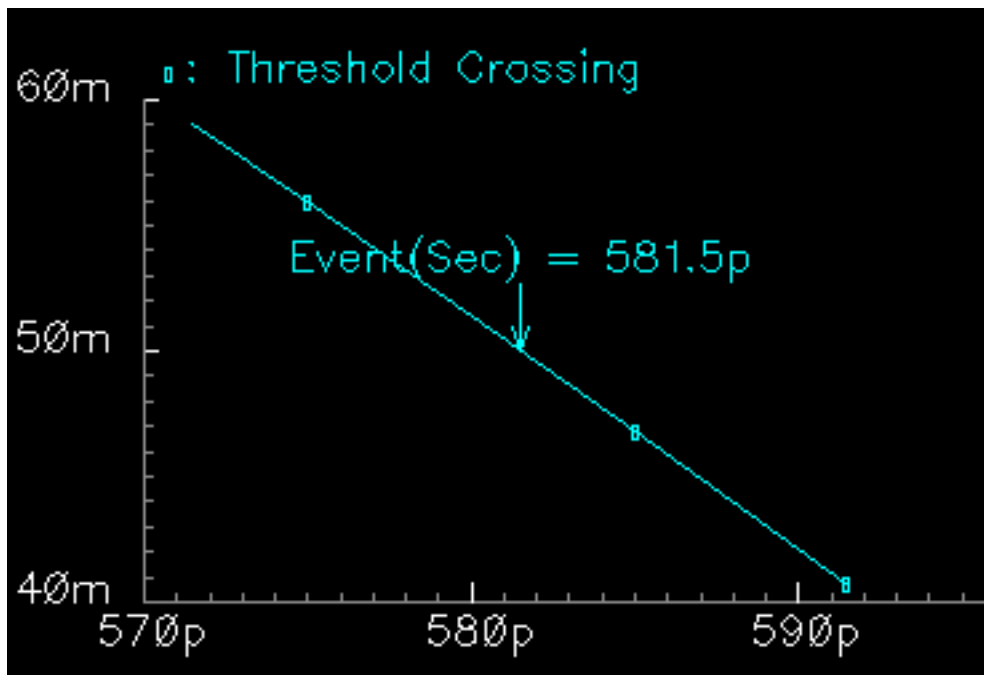
In the Direct Plot form, plot the downward transition.

1. Highlight *Replace* for *Plot Mode*.
2. Highlight *pnoise jitter* for *Analysis*.

The form changes to reflect the pnoise jitter analysis.

3. Highlight *Threshold Xing* for *Function*.
4. In the *Signal Level* cyclic field select *581.5p*.
5. Click *Plot*.

The down transition and its location display.



### Plot Edge to Edge Jitter

In the Direct Plot form, plot the edge to edge jitter.

1. Highlight *Replace* for *Plot Mode*.
2. Highlight *pnoise jitter* for *Analysis*.

3. Highlight *Jee* for *Function*.

The form changes to reflect the Function.

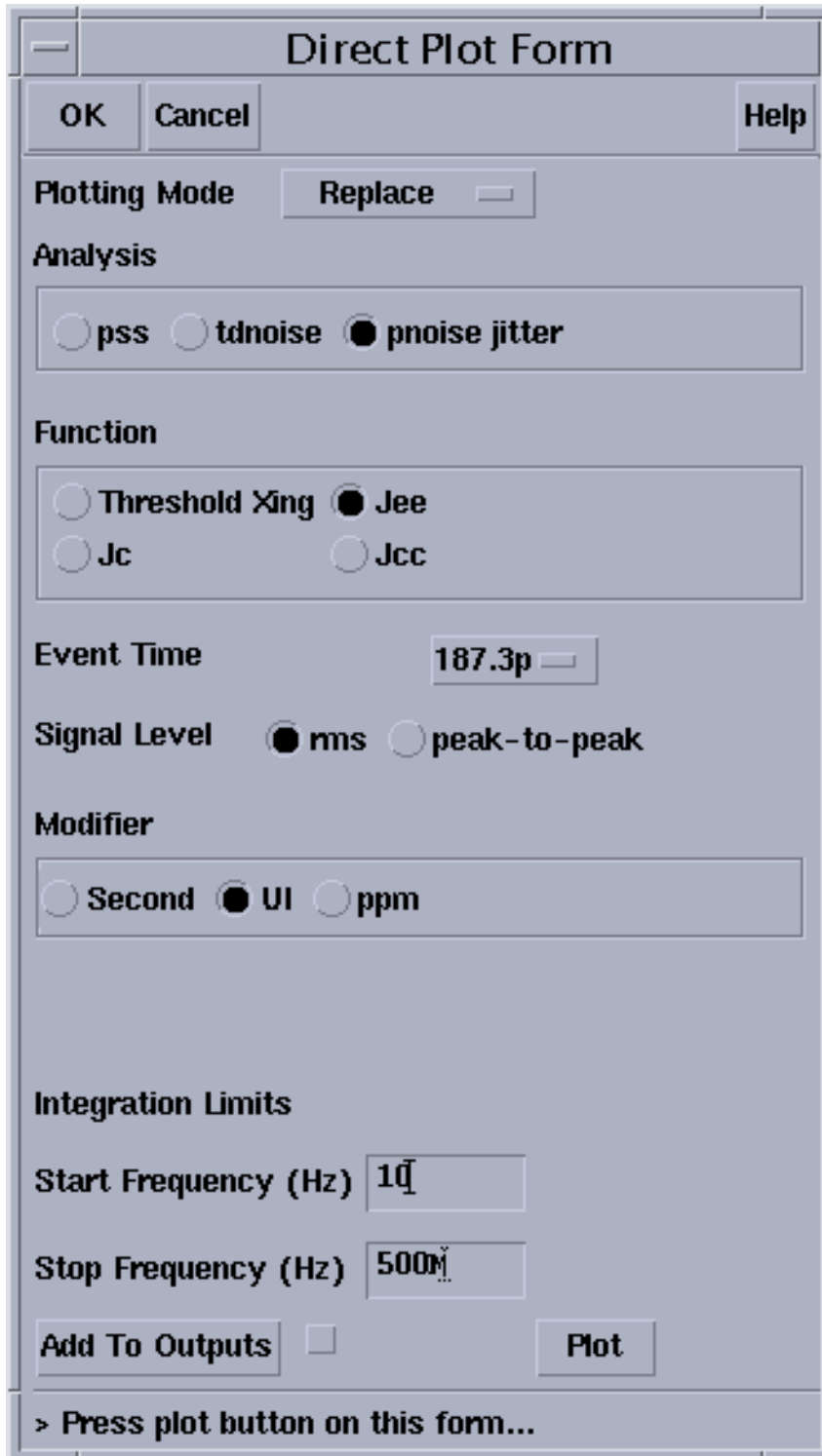
4. In the *Event Time* cyclic field, select *187.3p*.
5. Highlight *rms* for *Signal Level*.
6. Highlight *UI* for *Modifier*.



# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Modeling Transmitters

UI, the Unit Interval, plots time jitter scaled by the period.

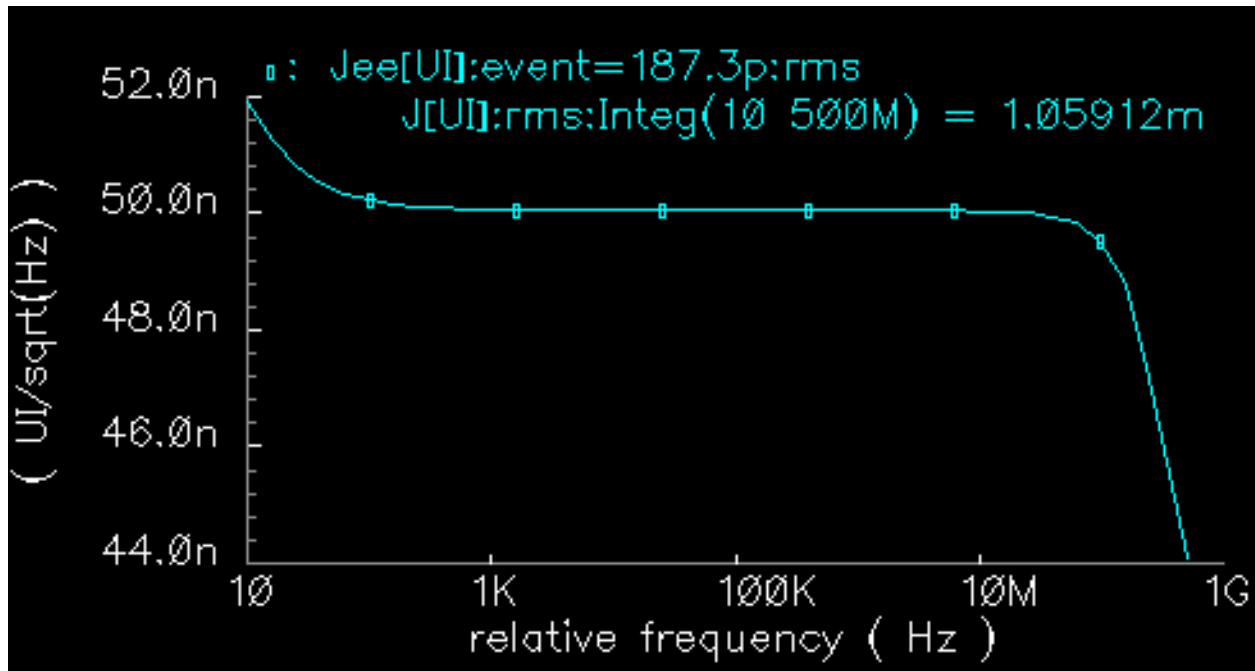


The image shows a dialog box titled "Direct Plot Form" with a standard Windows-style title bar. It contains several sections of controls:

- Buttons:** "OK", "Cancel", and "Help" are located at the top.
- Plotting Mode:** A dropdown menu is set to "Replace".
- Analysis:** Three radio buttons are present: "pss", "tdnoise", and "pnoise jitter", with "pnoise jitter" selected.
- Function:** Four radio buttons are present: "Threshold Xing", "Jee", "Jc", and "Jcc", with "Jee" selected.
- Event Time:** A text input field contains "187.3p".
- Signal Level:** Two radio buttons are present: "rms" (selected) and "peak-to-peak".
- Modifier:** Three radio buttons are present: "Second", "UI" (selected), and "ppm".
- Integration Limits:** Two text input fields: "Start Frequency (Hz)" with "1G" and "Stop Frequency (Hz)" with "500M".
- Buttons:** "Add To Outputs" (with an unchecked checkbox) and "Plot" are at the bottom.
- Footer:** A message "> Press plot button on this form..." is displayed at the very bottom.

7. Click *Plot*.

Edge to edge jitter displays.



Selecting the unit interval, *UI*, selects time jitter scaled by the period. This plots the frequency dependent jitter power spectral density. The plot also indicates the integrated, or total, jitter as a number scaled by the selected modifier, *UI* in this case.

Jitter at the up and down crossings is very close. The primary difference is the amount of noise at the lower frequency range, which does not matter after integration. In addition, the slope at the crossing, which we are using, is slightly different.

The *Integration Limits* display at the bottom of the Direct Plot form. In this instance the *Start Frequency* is 10 Hz and the *Stop Frequency* is 500M Hz.

### Plot Cycle to Cycle Jitter for Both Event Times

In the Direct Plot form, plot cycle to cycle jitter.

1. Highlight *Replace* for *Plot Mode*.
2. Highlight *pnoise jitter* for *Analysis*.
3. Highlight *Jcc* for *Function*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The form changes to reflect the cycle to cycle jitter analysis.

The *Number of cycles [k]* field displays 1.

4. In the *Event Time* cyclic field select 581.5p.
5. Highlight *rms* for *Signal Level*.
6. Highlight *Second* for *Modifier*.

The *Freq. Multiplier* displays as 1 and the *Integration Limits* display as 10 and 500M Hz.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

The Direct Plot form displays as follows.

The image shows a dialog box titled "Direct Plot Form" with a standard Windows-style title bar (minimize, maximize, close buttons). The dialog contains several sections of controls:

- Buttons:** "OK", "Cancel", and "Help" are located at the top.
- Plotting Mode:** A dropdown menu is set to "Replace".
- Analysis:** Three radio buttons: "pss", "tdnoise", and "pnoise jitter". "pnoise jitter" is selected.
- Function:** Four radio buttons: "Threshold Xing", "Jee", "Jc", and "Jcc". "Jcc" is selected.
- Number of Cycles [k]:** A text input field containing "1".
- Event Time:** A text input field containing "581.5p".
- Signal Level:** Two radio buttons: "rms" and "peak-to-peak". "rms" is selected.
- Modifier:** Three radio buttons: "Second", "UI", and "ppm". "Second" is selected.
- Freq. Multiplier:** A text input field containing "1".
- Integration Limits:**
  - Start Frequency (Hz):** A text input field containing "1G".
  - Stop Frequency (Hz):** A text input field containing "500M".
- Buttons:** "Add To Outputs" (with an unchecked checkbox) and "Plot" are at the bottom.
- Footer:** A message "> Press plot button on this form..." is displayed at the very bottom.

7. Click *Plot*.

Cycle to cycle jitter for the 581.5p event time displays.

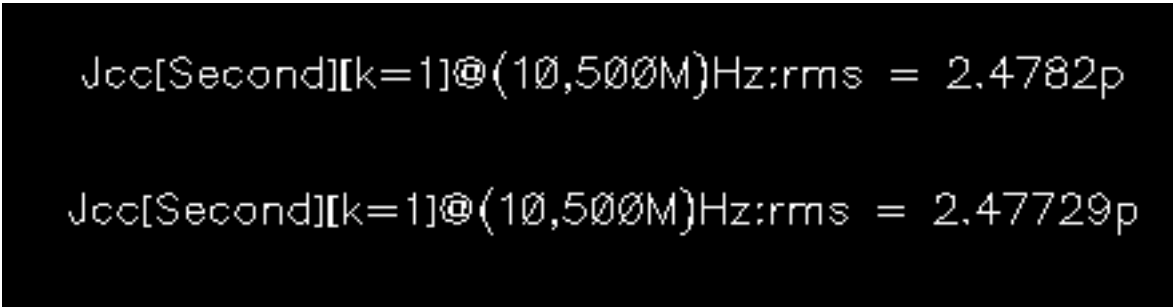


Jcc[Second][k=1]@(10,500M)Hz:rms = 2.4782p

In the Direct Plot form, make the following changes.

1. Highlight *Append* for *Plot Mode*.
2. In the *Event Time* cyclic field select *187.3p*.
3. Click *Plot*.

Cycle to cycle jitter displays for both event times.



Jcc[Second][k=1]@(10,500M)Hz:rms = 2.4782p  
Jcc[Second][k=1]@(10,500M)Hz:rms = 2.47729p

Jitter at the up and down crossings is very close. The primary difference is the amount of noise at the lower frequency range, which does not matter after integration. In addition, the slope at the crossing, which we are using, is slightly different.

The *Integration Limits* display at the bottom of the Direct Plot form. In this instance the *Start Frequency* is 10 Hz and the *Stop Frequency* is 500M Hz.

### Plot Jc for Both Event Times

In the Direct Plot form, plot Jc jitter.

1. Highlight *Replace* for *Plot Mode*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

2. Highlight *pnoise jitter* for *Analysis*.

3. Highlight *Jc* for *Function*.

The form changes to reflect the *Jc* jitter analysis.

The *Number of cycles [k]* field displays 1.

4. In the *Event Time* cyclic field select 187.3p.

5. Highlight *rms* for *Signal Level*.

6. Highlight *UI* for *Modifier*.

The *Freq. Multiplier* displays as 1 and the *Integration Limits* display as 10 and 500M Hz.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Modeling Transmitters

---

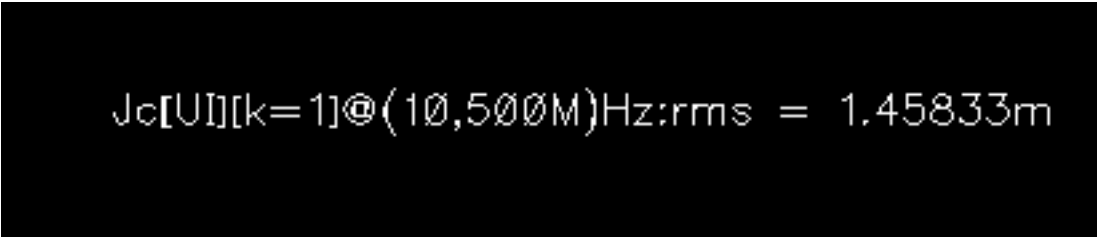
The Direct Plot form displays as follows.

The image shows a dialog box titled "Direct Plot Form" with a standard Windows-style title bar. At the top, there are three buttons: "OK", "Cancel", and "Help". Below the buttons, the dialog is organized into several sections:

- Plotting Mode:** A dropdown menu set to "Replace".
- Analysis:** Three radio buttons: "pss", "tdnoise", and "pnoise jitter". "pnoise jitter" is selected.
- Function:** Four radio buttons: "Threshold Xing", "Jee", "Jc", and "Jcc". "Jc" is selected.
- Number of Cycles [k]:** A text input field containing the value "1".
- Event Time:** A text input field containing the value "187.3p".
- Signal Level:** Two radio buttons: "rms" and "peak-to-peak". "rms" is selected.
- Modifier:** Three radio buttons: "Second", "UI", and "ppm". "UI" is selected.
- Freq. Multiplier:** A text input field containing the value "1".
- Integration Limits:** Two text input fields: "Start Frequency (Hz)" containing "10" and "Stop Frequency (Hz)" containing "500M".
- Buttons:** "Add To Outputs" (with an unchecked checkbox) and "Plot".
- Footer:** A message: "> Press plot button on this form...".

7. Click *Plot*.

Jc displays.



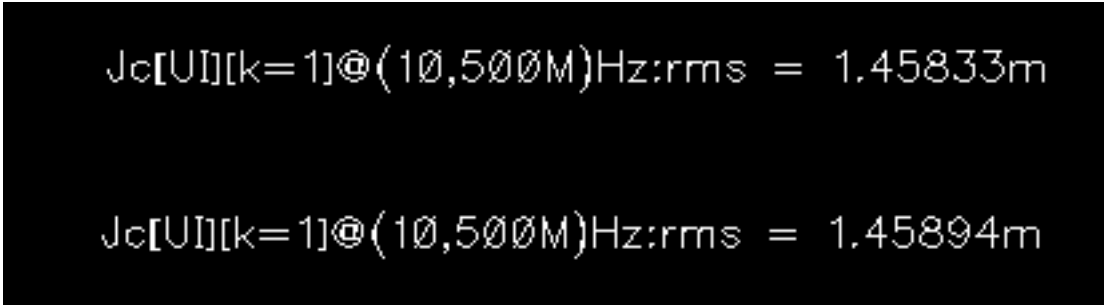
```
Jc[UI][k=1]@(10,500M)Hz:rms = 1.45833m
```

### Plot Jc for the 581.5p Event Time

In the Direct Plot form,

1. Highlight *Append* for *Plot Mode*.
2. Change the *Event Time* cyclic field selection to *581.5p*.
3. Click *Plot*.

Jc display for the second event time is appended to the plot.



```
Jc[UI][k=1]@(10,500M)Hz:rms = 1.45833m
```

```
Jc[UI][k=1]@(10,500M)Hz:rms = 1.45894m
```

Jitter at the up and down crossings is very close. The primary difference is the amount of noise at the lower frequency range, which does not matter after integration. In addition, the slope at the crossing, which we are using, is slightly different.

The *Integration Limits* display at the bottom of the Direct Plot form. In this instance the *Start Frequency* is 10 Hz and the *Stop Frequency* is 500M Hz.



---

# Methods for Top-Down RF System Design

---

## Methods for Top Down RF System Design

This chapter describes a methodology for designing analog RF subsystems that fit into larger DSP systems. In particular, this chapter describes how to use a canonical set of top-down behavioral baseband models for exploring RF architectures in the analog design environment. These models come from the following categories in *rfLib*

- Category *top\_dwnBB* contains models of common RF function blocks.
- The default view of each model is the baseband view (called *veriloga*).
- All models in this category also have a differential passband view (called *veriloga\_PB*).
- The only exceptions are the *BB\_loss* and *VGA\_BB* models. They are meant only for baseband analysis and have no passband view.
- Category *top\_dwnPB* contains single-ended passband versions of the baseband models.
- Category *measurement* contains the instrumentation block and baseband signal generator models used to make RF measurements. These elements are not part of an RF architecture. They simply facilitate RF measurements and diagnostics.
- Category *testbenches* contains the test circuits used in this chapter to define the model specifications in the *rfLib*. Where possible, the models are specified in terms of standard RF measurements. The most precise way to describe a measurement is with a test circuit, set up instructions, and sample measurements. The circuits in the testbenches category serve that purpose

See [Appendix D, “The RF Library”](#) for more information about the *rfLib* and detailed descriptions of the models it contains.

These models provide RF designers with a fast method to map RF system specifications into detailed RF designs. The baseband models facilitate fast evaluation of candidate RF architectures specified with DSP metrics. The passband views of the baseband models

provide a behavioral system testbench for checking detailed designs of individual RF system components.

Baseband models are behavioral models and all behavioral models sacrifice some accuracy for increased simulation speed. Such sacrifices are usually acceptable in architectural studies because many implementation-dependent details do not affect high level decisions. The modeling approach taken in top-down design is to simulate only those effects that drive the decisions at hand.

Baseband modeling in no way replaces passband modeling. Some effects missed by equivalent baseband models can affect high level decisions. However, the application of baseband models early and passband models later minimizes the number of slow simulations needed at the lower levels of design abstraction. Baseband models help you to quickly weed out designs that would surely fail tests simulated with passband models.

The success of a modeling approach to top-down design hinges on knowing how the models fit into the design flow and knowing exactly what each modeling parameter means. This chapter has two goals:

- To describe the top-down design flow, from a modeling perspective, for baseband modeling
- To define, as clearly and concisely as possible, the parameters that specify the models

## Top-Down Design of RF Systems

Ideally, the digital signal processing, or DSP, team specifies an RF subsystem that fits snugly into the DSP system. A *snug fit* means that

- The specified RF subsystem does exactly what it needs to do at the lowest possible cost
- A functional specification exists that describes requirements for the RF subsystem

In a top down design flow like the one shown in [Figure 8-1](#) on page 556, the DSP team writes a functional specification for an RF subsystem that has not yet been designed. The functional specification describes what the RF subsystem should and should not do without describing how to build the RF subsystem.

The functional specification supplied by the DSP team describes the RF subsystem at the highest possible level of abstraction. At this point behavioral models can be specified rather than measured. This early in the design cycle, the functional specification might well be incomplete or inconsistent. A good top-down design flow can detect problems, such as omissions and inconsistencies in the design, early in the design cycle when they are easier

and less expensive to fix. Problems detected later in the design cycle can be much more costly and very difficult to resolve.

Using the functional specification supplied by the DSP team and the behavioral baseband models from *rfLib*, the RF system designers can easily explore RF architectures in the analog design environment. The baseband models facilitate fast evaluation of candidate RF architectures specified with DSP metrics. By switching to the passband views of the baseband models, the RF design team maps DSP measurements to RF measurements. The passband views of the baseband models provide a behavioral system testbench for checking detailed designs for individual system components.

Using the functional specification and exploring and testing with the baseband and passband models, the RF team can efficiently create a detailed design specification that fully describes the RF subsystem. The design specification can include detailed instructions for building the RF subsystem. At this stage of the design cycle, everything that is known about the design is described at the lowest level of abstraction.

You can now extract behavioral models of a detailed design from simulated measurements. The problem remains that detailed designs usually do not exist until the project is complete. To jump directly to a detailed design implies that the design flow is bottom up. Bottom up flows are important in many projects, but not in all.

DSP and RF designers sometimes have trouble communicating through specifications because the two groups deal with different metrics. For example, DSP designers deal with *bit error rates* and *error vector magnitude statistics* whereas RF designers deal with *intercept points* and *noise figures*.

The new models described here are designed to help RF system designers in two ways.

First, the baseband models enable RF system designers to quickly explore the RF architectural space, as specified by the DSP metrics, while letting the RF engineers specify the RF system components with RF metrics. The circuit implementations of the RF system components are easier to design and test when the components are specified with standard RF metrics.

Second, the baseband models can be switched quickly to a passband views where the RF system model can generate end-to-end RF metrics. With end-to-end metrics, the new view can quickly simulate how the detailed design of a particular RF system component affects end-to-end performance.

## **Use Model for Top Down Design**

The following steps outline the RF design process with focus on the early phases of the design as illustrated in [Figure 8-1](#) on page 556.

Figure 8-1 The Top Down Design Flow and Use Model



### Specify the RF Subsystem in Terms of DSP Metrics

Before you begin the RF subsystem top-down design flow, the DSP design team should completely specify the RF subsystem in terms of DSP metrics. This preliminary step distinguishes the end of the DSP design flow from the beginning of the RF top down design flow and formally hands-off the RF subsystem design specifications to the RF design team.

### Explore Candidate Architectures with Baseband Models

The first step in top down RF design is to select a candidate RF architecture. An RF architecture is a set of interconnected RF function blocks that, taken together, describe how a receiver or transmitter operates. You specify each function block in terms of standard RF metrics such as IP3, gain, bandwidth, and noise figure.

The models you use early in the design cycle as you explore candidate RF architectures must run fast. Each simulations can span hundreds of symbols and each symbol can easily span thousands of RF carrier cycles. The space defined by the function block specifications in each

candidate architecture is far too vast to explore with slow, highly precise models. Models used for architectural exploration must quickly reduce the design space down to a size that can be explored with more precision.

The most efficient models for architectural exploration suppress the RF (IF) carrier and are called *baseband* models. In contrast, the *passband* models (introduced in the next step) do not suppress carriers.

You can use the Circuit Optimizer during architectural exploration to help balance the function block specifications for a candidate architecture. For example, you can use the Circuit Optimizer to minimize RMS EVM while ensuring that other measurements stay within acceptable limits.

When you have determined the nominal specifications for each function block, you must put tolerances around them. In the analog world *specifications without tolerances are meaningless*. The tolerance space is usually explored with some mix of experience, feasibility, a variety of analyses, and outright arbitrary decisions.

There are several ways that you can use the baseband models to test candidate tolerances as well as to determine some tolerances analytically.

One way to test a candidate set of tolerances is to run a Monte Carlo analysis on the metric of interest, like RMS, EVM, or signal-to-noise ratio (SNR).

Another approach is to use the Circuit Optimizer *in reverse*, as a de-optimizer, to determine worst case performance.

Yet another approach is to compute each tolerance separately from a parametric plot. When you have determined all but 2 or 3 tolerances, you can use a multidimensional parametric analysis to map out the performance space and easily identify the remaining tolerances.

### **Switch to Passband Models and Create an RF System Testbench**

The second step in top down RF design is to create a passband view of the system model.

The passband system model performs two functions:

- Confirms that the filters perform as expected.
- Creates an end-to-end testbench that you can use to design the individual function blocks.
- For computational efficiency during system passband testing, at any one time, model one or two selected function blocks at the device level. Model all other blocks in the system behaviorally using passband models.

Derive the tolerances by performing the same Monte Carlo analysis or Circuit Optimizer analysis you used to test the function block tolerances in the first step, but this time replace the DSP metrics with end-to-end RF metrics. Once you know how far the end-to-end RF metrics can vary, you can insert a device-level model of a function block into the testbench to see how close it drives the system toward violating a derived end-to-end RF specification.

### **Implement the Function Blocks with Active and Passive Devices**

The last step in the top down design process is to implement the function blocks with device models. Since the function blocks are specified in terms of standard RF metrics, you can easily measure the modeling parameters to make sure they fall within the specified tolerances. You can also insert the measured parameters back into the baseband model of the system to check the DSP metrics, or insert the device-level model directly into the passband testbench to check the derived end-to-end RF specifications.

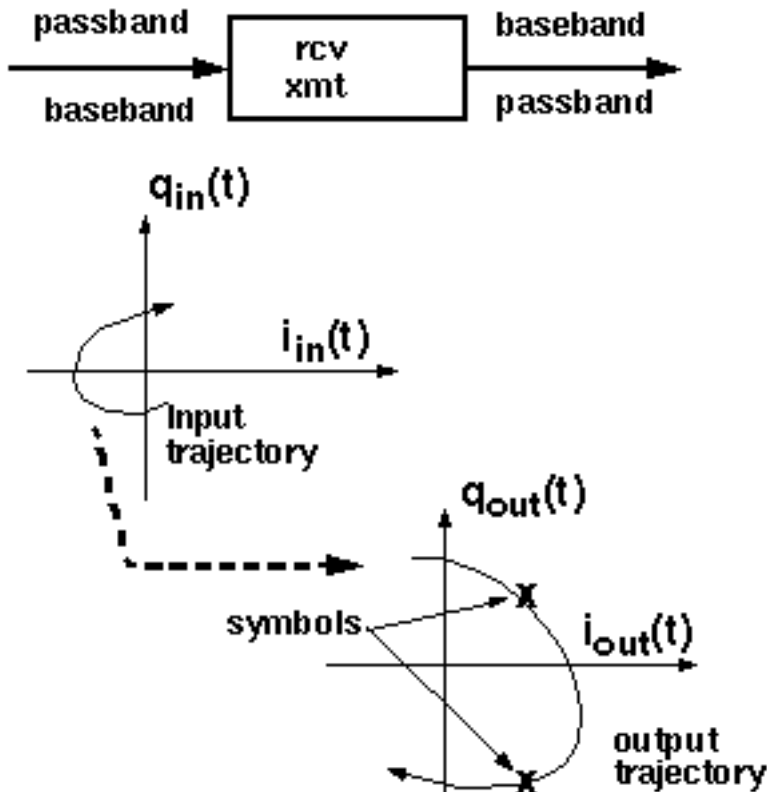
## **Baseband Modeling**

A baseband model for an RF function block simulates what happens to the baseband representation of a signal as it passes through the block. A baseband model maps input baseband signal trajectories into output baseband signal trajectories. If you sample a baseband signal periodically in time and plot the samples in the complex plane, the resulting scatter plot shows the symbol constellation.

Figure 8-2 on page 559 mathematically defines a baseband representation of a passband signal. The  $i$  and  $q$  signals are the real and imaginary parts of a complex signal that rides on the two phases of an RF carrier.

**Figure 8-2 Baseband Representation of a Passband Signal**

**passband signal =  $i(t)\cos(\omega_{rf}t) - q(t)\sin(\omega_{rf}t) = \text{real}$**   
**baseband representation =  $i(t) + j*q(t) = \text{complex}$**



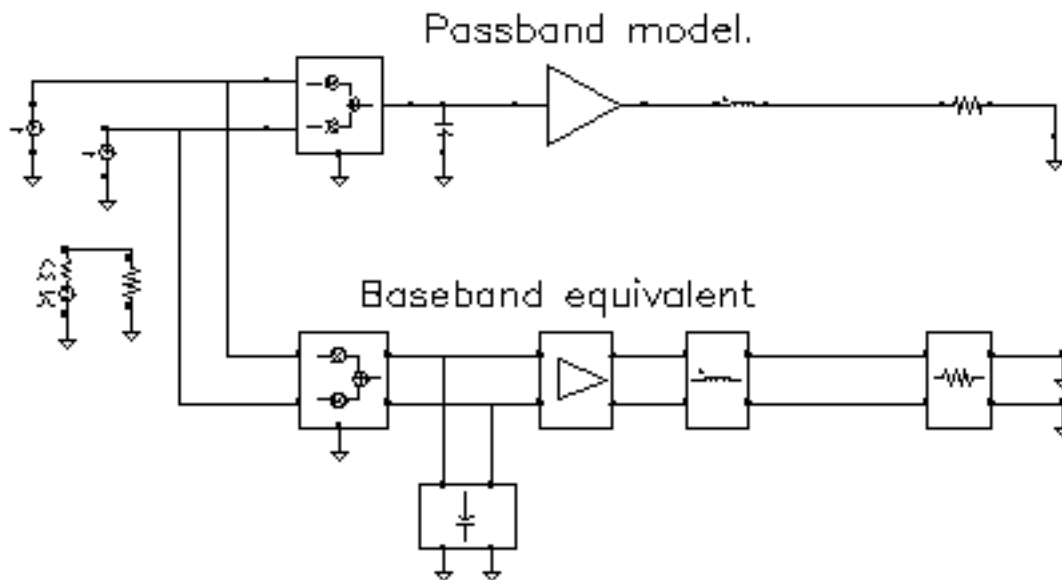
Baseband models simulate only what happens to the carrier fundamental. Consequently, they only account for non-linearities with odd symmetry. Non-linearities with even symmetry produce no output at the carrier fundamental; they affect the carrier fundamental only when cascaded. For example, a second order non-linearity in one block can create a DC offset at it's output. Upon passing through a subsequent block with another second order non-linearity, the DC offset can mix with the carrier to affect the output carrier fundamental. You should model cascaded blocks producing unfiltered even harmonics as a single baseband model rather than as separate baseband models cascaded together. The non-linearities that most often dominate performance have odd symmetry.

## Example Comparing Baseband and Passband Models

The example in this section walks you through an Envelope analysis that illustrates the relationship between baseband and passband models. Following the simulation, you plot the baseband equivalent output signals as computed by the baseband and passband circuits.

The *BB\_test\_bench* schematic shown in [Figure 8-3](#) on page 560 illustrates the difference between passband and baseband modeling. This circuit is located in the *rfExamples* library.

**Figure 8-3 The BB\_test\_bench Schematic**



The *BB\_test\_bench* circuit shows a passband circuit (across the top of the schematic) and its baseband equivalent circuit (across the bottom of the schematic). The same baseband signals drive both circuits but only the passband circuit mixes the baseband signals up to RF. The power amplifier is not matched to either input or output impedances and both impedances are reactive.

Before you start, perform the setup procedures described in [Chapter 3](#).

### Opening the Baseband Test Bench Circuit

1. In the CIW, choose *File – Open*.

The Open File form appears.

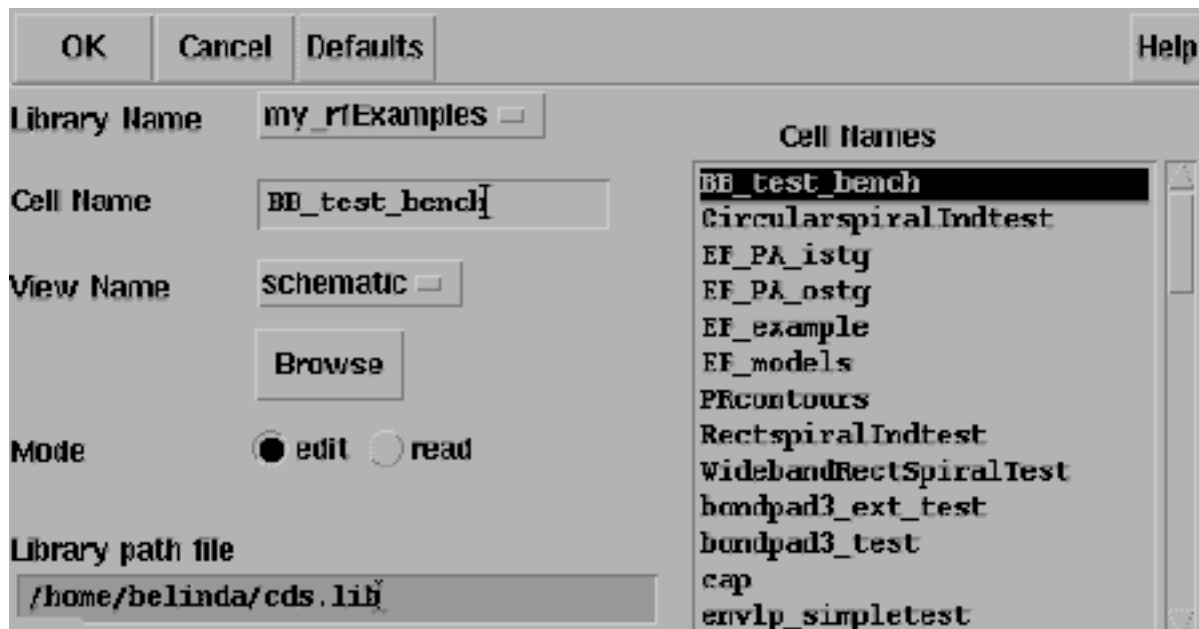


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

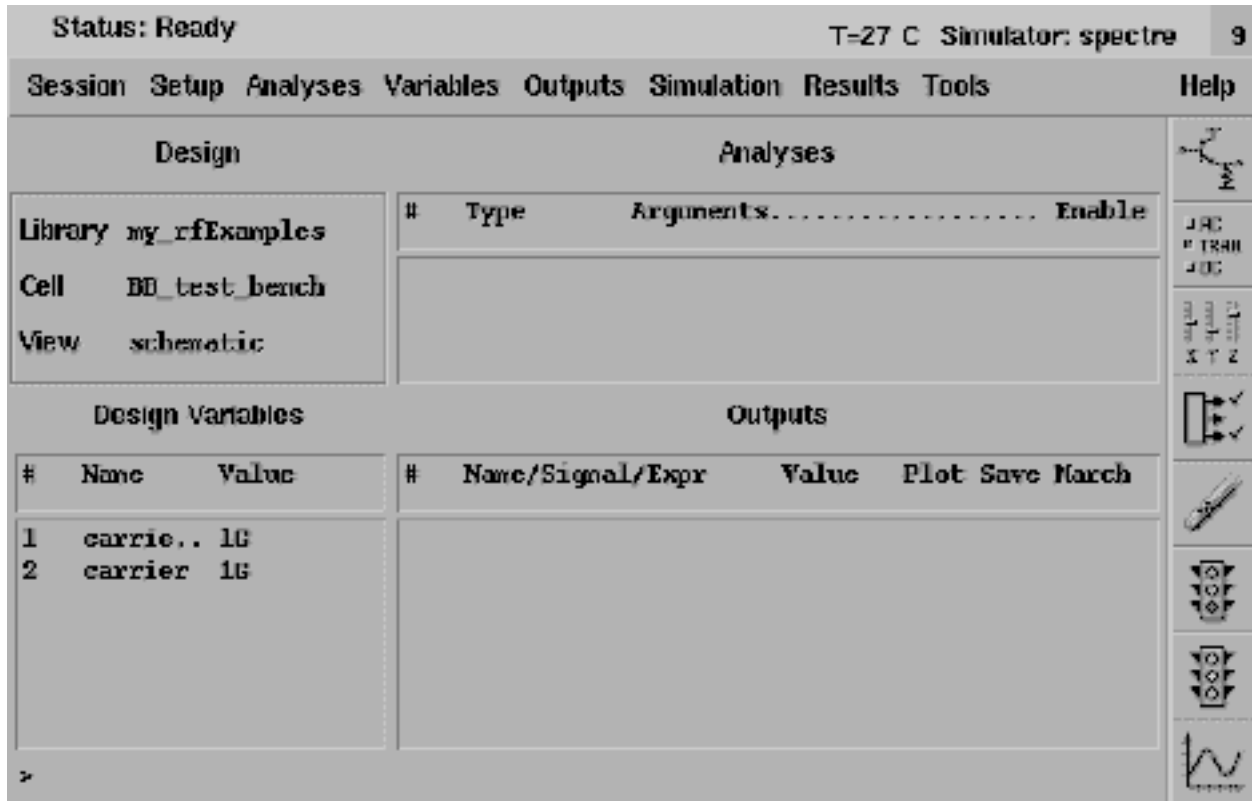
2. In the Open File form,
  - a. Choose *rfExamples* in the *Library Name* cyclic field. (Choose the editable copy of *rfExamples* you created as described in [Chapter 3](#).)
  - b. Choose *BB\_test\_bench* in the *Cell Names* list box. Note that the *View Name* cyclic field displays *Schematic*.
  - c. The completed Open File form appears like the one below.



3. Click *OK*.

The Schematic window for the *BB\_test\_bench* appears.
4. In the Schematic window, choose *Tools – Analog Environment*.

The Simulation window opens.



You can also use *Tools – Analog Environment – Simulation* in the CIW to open the Simulation window without opening the design. You can open the design later by choosing *Setup – Design* in the Simulation window and choosing the *BB\_test\_bench* in the Choosing Design form.

### Choosing Simulator Options

1. Choose *Setup – Simulator/Directory/Host* in the Simulation window.  
 The Simulator/Directory/Host form appears.
2. In the Simulator/Directory/Host form, specify the following:
  - a. Choose *spectre* for the *Simulator*.
  - b. Type the name of the project directory, if necessary.
  - c. Highlight the *local* or the *remote* button to specify the *Host Mode*.

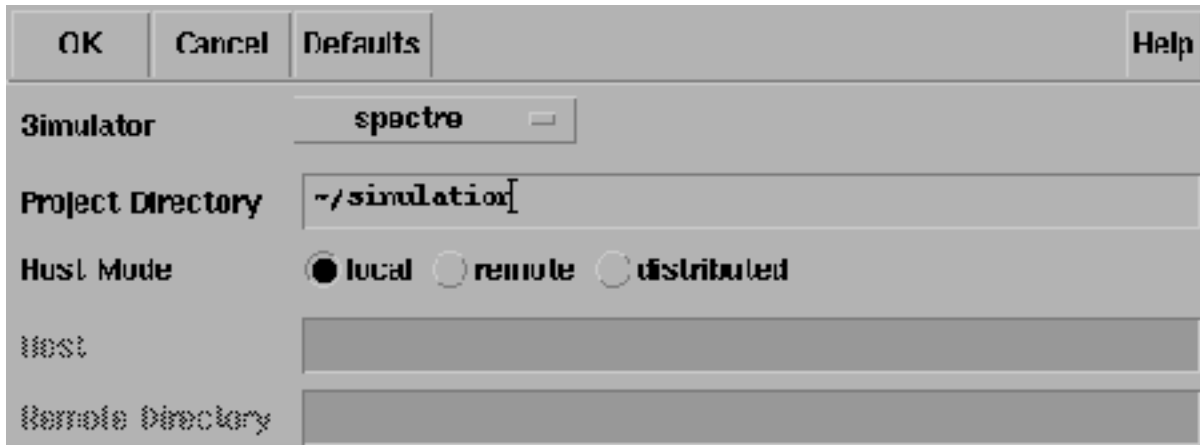
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

For remote simulation, type the name of the host machine and the remote directory in the appropriate fields.

The completed form appears like the one below.



The image shows a dialog box for configuring the Spectre simulator. It has a title bar with buttons for 'OK', 'Cancel', 'Defaults', and 'Help'. The main area contains several fields:

- Simulator:** A dropdown menu showing 'spectre'.
- Project Directory:** A text field containing '~/.simulation['.
- Host Mode:** Three radio buttons labeled 'local', 'remote', and 'distributed'. The 'local' button is selected.
- Host:** An empty text field.
- Remote Directory:** An empty text field.

3. In the Simulator/Directory/Host form, click *OK*.
4. In the Simulation window, choose *Outputs – Save All*.

The Save Options form appears.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

5. In the *Select signals to output (save)* section, be sure *allpub* is highlighted.

The screenshot shows a dialog box titled 'Save Options' with the following settings:

- Select signals to output (save):**  none  selected  lvipub  lvi  allpub  all
- Select power signals to output (pwr):**  none  total  devices  subckts  all
- Set level of subcircuit to output (poslvl):** [Empty text field]
- Select device currents (currents):**  selected  nonlinear  all
- Set subcircuit probe level (subcktprobelvl):** [Empty text field]
- Select AC terminal currents (useprobes):**  yes  no
- Select AHDL variables (saveahdlvars):**  selected  all
- Save model parameters info:**
- Save elements info:**
- Save output parameters info:**

6. In the Save Options form, click *OK*.

## 7. Setting Up Model Libraries

8. In the Simulation window, choose *Setup – Model Libraries*.

The Model Library Setup form appears.

9. In the *Model Library File* field, type the full path to the model file including the file name, `rfModels.scs`.

10. In the Model Library Setup form, click *Add*.

The completed form appears like the one below.

The screenshot shows a dialog box titled "Model Library Setup". At the top, there are buttons for "OK", "Cancel", "Defaults", "Apply", and "Help". The main area is divided into two sections. The first section is titled "Model Library File" and "Section". It contains a text field with the path ".../pink/tools/dfII/samples/artist/models/spectre/rfModels.scs" and an empty "Section" field. The second section is titled "Model Library File" and "Section (opt.)". It contains an empty text field for the file path and an empty "Section (opt.)" field. At the bottom of the dialog, there are buttons for "Add", "Delete", "Change", "Edit File", and "Browse...".

11. In the Model Library Setup form, click *OK*.
12. In the Simulation window, use *Analysis - Disable* to disable any analyses you ran previously. (Check the Simulation window to verify whether or not an analysis is enabled.)

### Setting Up the Envelope Analysis

1. Choose *Analyses – Choose* in the Simulation window.

The Choosing Analyses form appears.

2. In the Choosing Analyses form, click *envlp*.
  - a. Enter *ff* in the *Clock Name* field.
  - b. Enter *10u* in the *Stop Time* field.
  - c. In the *Output Harmonics* cyclic field, select *Number of harmonics*.
  - d. Enter *1* in the *Number of harmonics* field.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

The correctly filled out form appears below.

OK Cancel Defaults Apply Help

Analysis  tran  dc  ac  noise  
 xf  sens  dcmatch  stb  
 sp  envlp  pss  pac  
 pnoise  pxf  psp  qpss  
 qpac  qpnoise  qpxf  qpssp

Envelope Following Analysis

Clock Name

Stop Time

Output Harmonics  
Number of harmonics

Accuracy Defaults (empreset)  
 conservative  moderate  liberal

Enabled

3. In the Choosing Analyses form, click *OK*.

### Running the Simulation

1. In the Simulation window, choose *Simulation – Netlist and Run*.

The output log file appears and displays information about the simulation as it runs.

Look in the CIW for a message that says the simulation completed successfully.

### Plotting the Baseband Equivalent Output Signals

1. In the Simulation window, choose *Results-Direct Plot-Main Form*.

The Direct Plot form appears.

2. In the Direct Plot form, do the following:

- a. Highlight *Replace* for *Plot Mode*.

- b. Highlight *envlp* for *Analysis*.

- c. Highlight *Voltage* for *Function*.

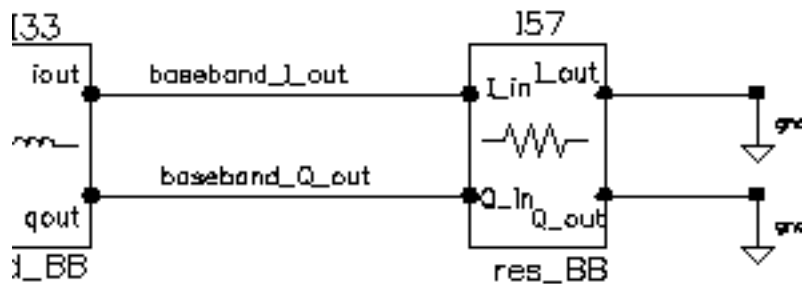
- d. Highlight *time* for *Sweep*.

- e. Notice the *Net* selection in the *Select* cyclic field and the message *Description: Envelope Voltage vs Time*

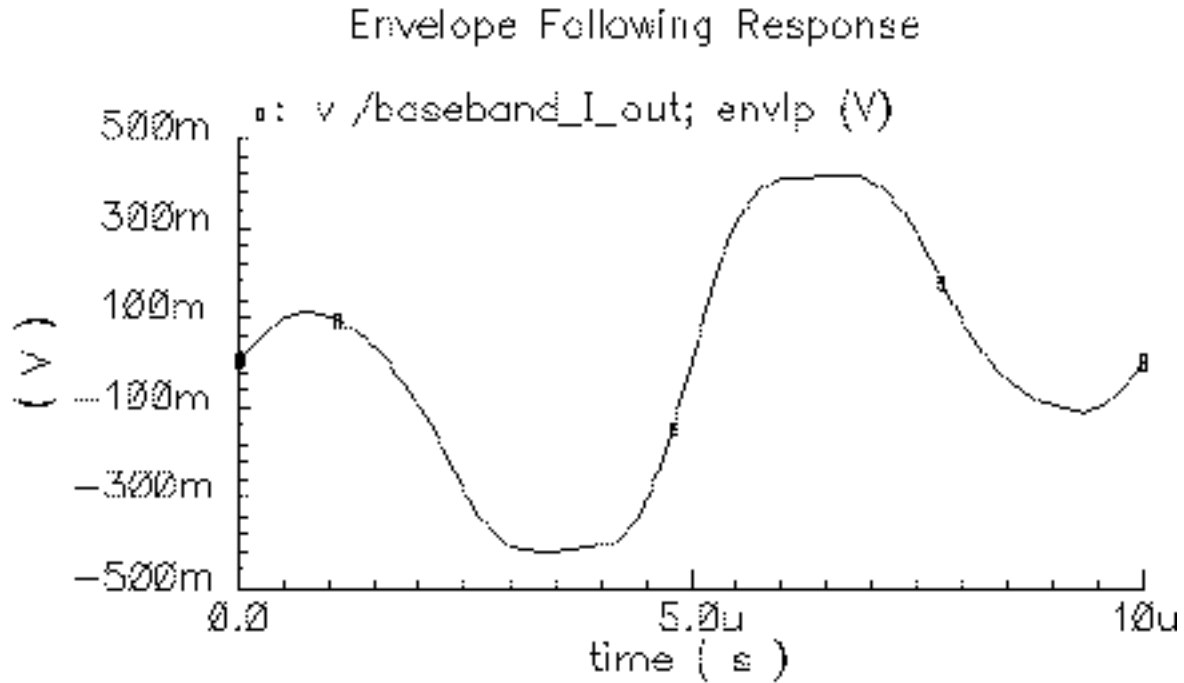
- f. Following the message at the bottom of the Direct Plot form

*Select Net on schematic...*

Click the `baseband_I_out` net.



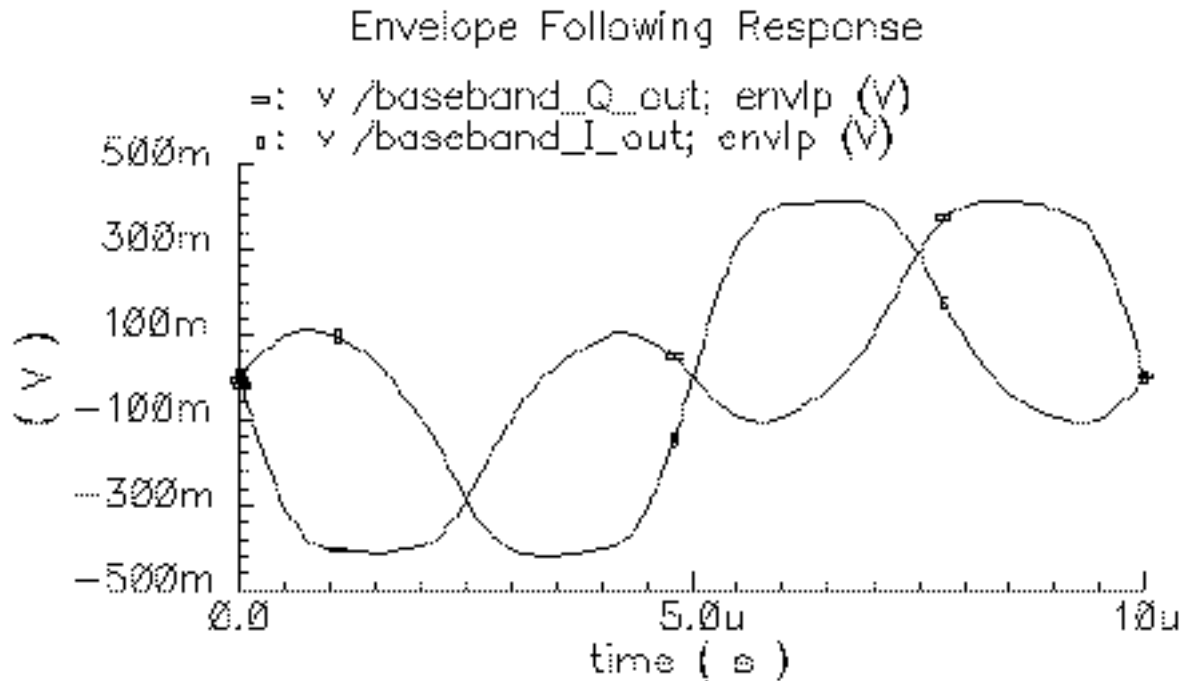
The first trace appears in the Waveform window.



3. In the Direct Plot form, do the following:
  - a. Leave *Voltage* set for *Function* and *time* set for *Sweep*.
  - b. Highlight *Append* for *Plot Mode*.
  - c. Following the message at the bottom of the Direct Plot form  
*Select Net on schematic...*  
Click the `baseband_Q_out` net.



The second trace is added to the Waveform window. Both baseband equivalent output signals for the baseband model are plotted.

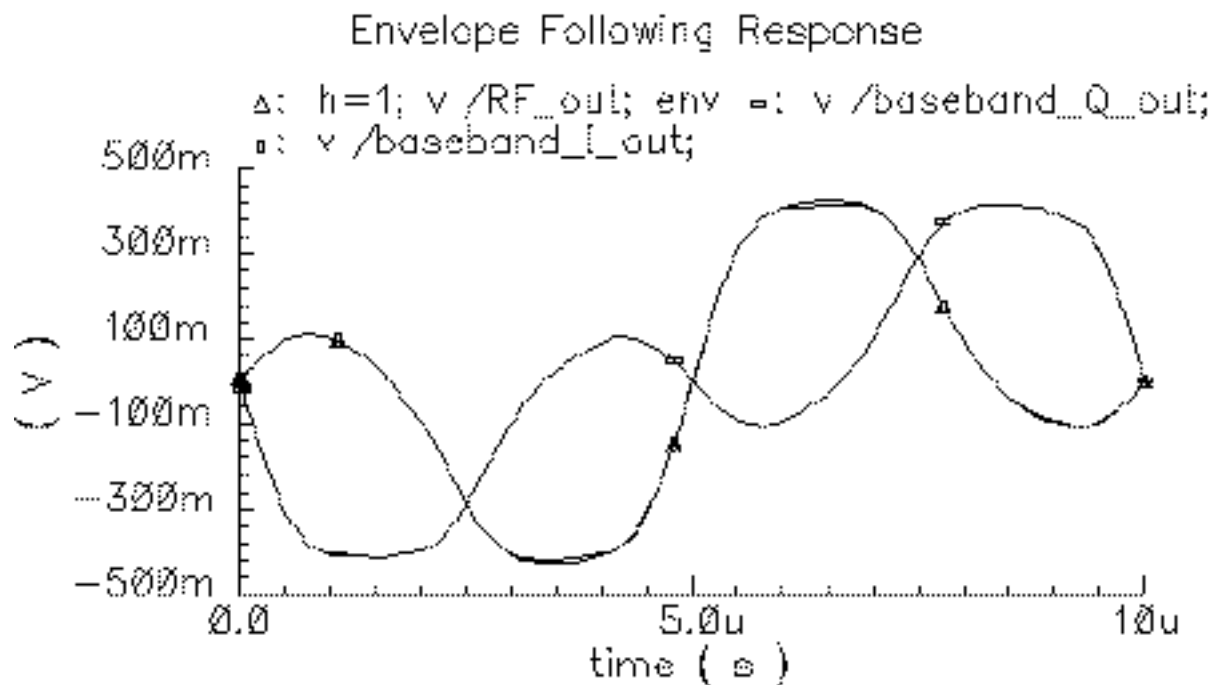


4. In the Direct Plot form, do the following:
  - a. Leave *Append* for *Plot Mode* and *Voltage* for *Function*.
  - b. Highlight *harmonic time* for *Sweep*.
  - c. Highlight *Real* for *Modifier*.
  - d. Following the message at the bottom of the form,  
Select Harmonic Number on this form...  
Select 1 for harmonic number.
  - e. Following the next message at the bottom of the form  
Select Net on schematic...

Click the RF\_out net.

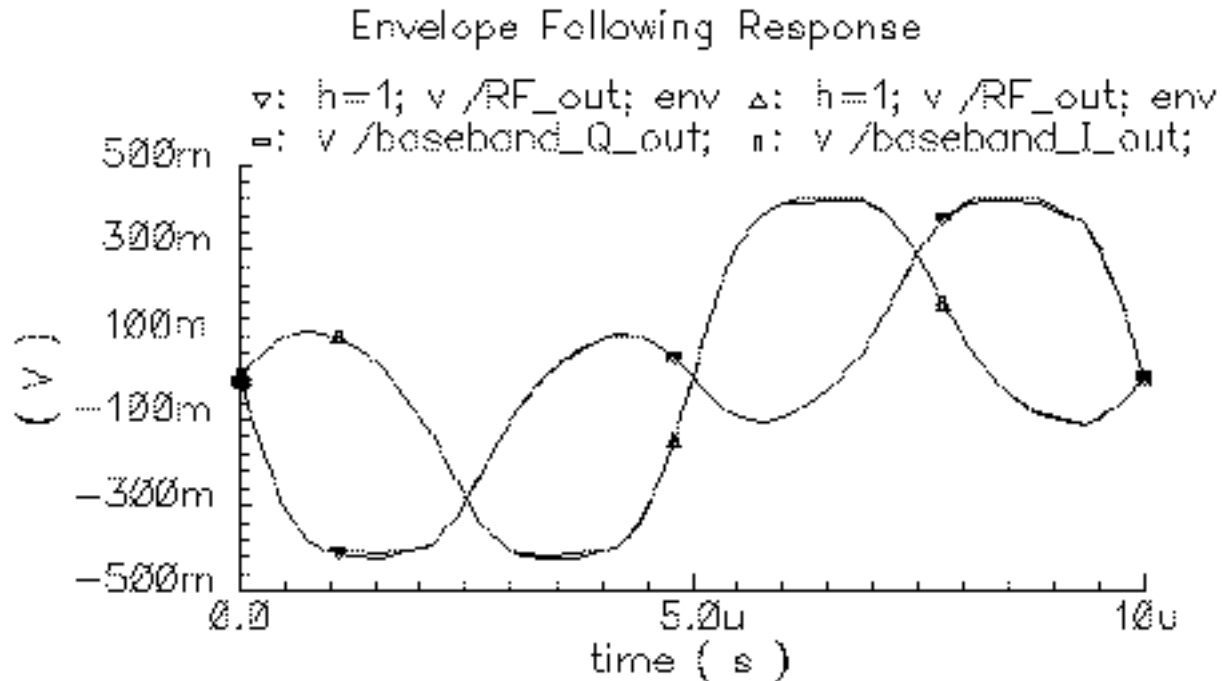


A third trace is added to the Waveform window.



5. In the Direct Plot form, do the following:
  - a. Leave *Append* for *Plot Mode*, *Voltage* for *Function*, *harmonic time* for *Sweep*, and 1 for *Harmonic Number*.
  - b. Highlight *Imaginary* for *Modifier*.
6. Following the message at the bottom of the form  
 Select Net on schematic...  
 Click the RF\_out net again.

A fourth trace is added to the Waveform window. Both baseband equivalent output signals for the passband model are added to the plot.



In the Waveform display window you should now see what at first appears to be two traces. When you look more closely, you should see that each trace is actually two traces, one nearly on top of the other, making a total of four traces.

The plot resulting from this example illustrates how well baseband modeling corresponds to the time-varying fundamental Fourier component computed by Envelope analysis and raises two questions:

- Why use baseband models when Envelope analysis gives the same results?
- Why not use baseband models all the time?

Running a transient analysis with only the baseband models answers the first question. If from the Simulation window you deactivate the passband circuit by setting the *carrier\_pb* variable to zero, disable the Envelope analysis, and set up and run a 10 $\mu$ s transient analysis, you observe the same baseband results, but the transient simulation runs over 100 times faster.

Examining the Envelope results answers the second question. If you look closely at both waveforms you notice that the baseband waveforms clip at a slightly lower level than the Envelope waveforms. This is because hard limiting of the carrier generates higher-than-third-

order harmonics and the behavioral baseband model only simulates third order non-linearities.

## rfLib Library Overview

The *rfLib* include three kinds of models to support baseband modeling:

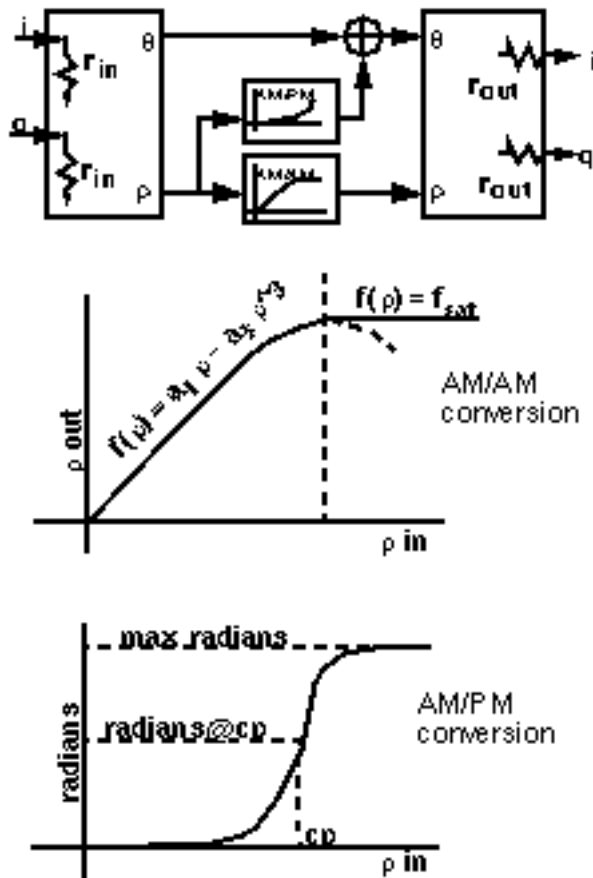
- Instrumentation Models
- Non-Linear Memoryless Models
- Linear Models With Memory

The instrumentation models provide stimuli, diagnostics, and performance metrics relevant to the DSP system.

Both the linear models with memory and the non-linear memoryless models simulate the function blocks in an RF architecture and are specified in terms of common RF metrics. The RF function block models include input referenced white Gaussian noise as specified by noise figure. The *rfLib* includes models for the following RF function blocks—amplifiers, mixers, filters, and phase shifters; where *filters* includes single resistors, capacitors, and inductors.

The non-linear models simulate AM/AM conversion [1] with a third-order polynomial that saturates at the peak of the transfer curve. The polynomial is specified by the gain and either the input-referred IP3 or the output-referred 1 dB compression point. Only the non-linear baseband models simulate AM/PM conversion. AM/PM conversion [1] is an important effect that is hard, if not impossible, to simulate with passband behavioral models. [Figure 8-4](#) on page 573 shows the basic baseband non-linearity.

Figure 8-4 Basic, Baseband Non-Linearity



The linear models are the key to simulating loading effects at baseband. In RF integrated circuits, loading effects are important because it is often hard to integrate impedance matching networks. The baseband models of reactive elements differentiate our approach from the spreadsheet-based approaches to RF system design. The baseband capacitor and inductor models (*cap\_BB* and *ind\_BB* in *top\_downBB*) let you simulate reactive loading effects in the time domain, where non-linearities are more naturally modeled.

The baseband models of reactive elements also play a key role in modeling filters. Most digital communications text books [1,2] explain that you can model a passband transfer function at baseband by simply frequency-shifting the transfer function. What these books do not describe is how to implement the resulting transfer function in a general circuit simulator such as Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF). The shifted transfer function usually lacks complex conjugate symmetry about zero frequency and therefore has a complex impulse response.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

The first consequence of modeling RF function blocks at baseband is that all equivalent baseband models have four terminals instead of two:

- One set of terminals represents the in-phase signals,  $i_{in}(t)$  and  $i_{out}(t)$
- The other set of terminals represents the quadrature signals,  $q_{in}(t)$  and  $q_{out}(t)$

Both sets of terminals are illustrated in [Figure 8-2](#) on page 559.

The mathematics illustrated in [Figure 8-5](#) on page 575 and [Figure 8-6](#) on page 576 summarize the ideas behind a time-varying coordinate transformation that models reactive elements at baseband. The mathematics apply to capacitors as well as inductors.

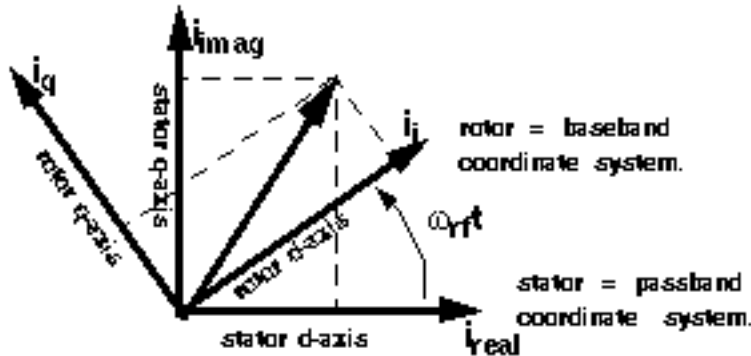
There is a well-documented but little-known electro-mechanical analogy for the derivation of the inductor baseband equivalent model. The four inductor terminals resemble the stator windings of a two-phase rotating machine with shaft speed equal to the RF carrier frequency. Modulation is mathematically analogous to the flux linking a stator winding due to currents in orthogonal rotor windings. The flux depends on the shaft angle just as a modulated signal depends on the carrier phase. Transforming the vectorial equation for  $v=Ldi/dt$  to the rotor reference frame suppresses the RF carrier and introduces a *speed voltage* [3,4,5,6,7,8,9], or back electro-motive force (back EMF), that couples the differential equations.

An expression for the real current (i.e. the passband current) appears in [Figure 8-6](#) on page 576. The real current is modeled as the projection of a two-dimensional rotating vector onto a stationary axis, the *real* axis. The vector rotates with an angular velocity equal to the RF carrier frequency.

Figure 8-5 Passband Current for an Inductor

$$i_{\text{real}} = i_i(t)\cos(\omega_{\text{rf}}t) - i_q(t)\sin(\omega_{\text{rf}}t) = \text{Real}[(i_i + j i_q) e^{j\omega_{\text{rf}}t}]$$

$$i_{\text{imag}} = i_i(t)\sin(\omega_{\text{rf}}t) + i_q(t)\cos(\omega_{\text{rf}}t) = \text{Imag}[(i_i + j i_q) e^{j\omega_{\text{rf}}t}]$$



$$i = (i_i + j i_q) e^{j\omega_{\text{rf}}t}$$

$$v = (v_i + j v_q) e^{j\omega_{\text{rf}}t}$$

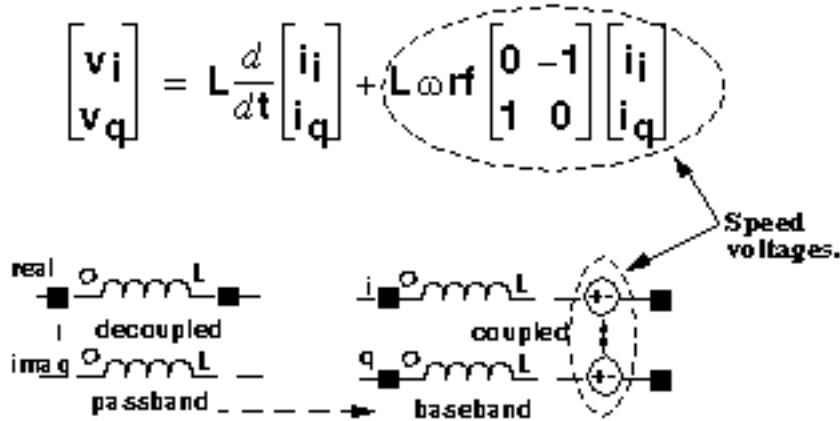
$$v = L di/dt$$

~~$$(v_i + j v_q) e^{j\omega_{\text{rf}}t} = L [d(i_i + j i_q)/dt + j \omega_{\text{rf}} (i_i + j i_q)] e^{j\omega_{\text{rf}}t}$$~~

The rotating vector also has a projection onto another stationary axis orthogonal to the real axis. In the baseband literature, the orthogonal projection is the Hilbert transform of the real signal. The constitutive relationship of the inductor,  $v = L di/dt$ , is expressed in terms of coordinates in a reference frame that rotates with the vector.

Figure 8-6 on page 576 shows the constitutive inductor relationship between voltage and current in the rotating reference frame. Note that the trigonometric terms, the terms that slow simulation speed, are gone and the two projections are now coupled through *speed voltages*. The term speed voltage comes from the fact that the voltages depend on the angular speed of the rotating reference frame. In motor theory, that speed is the shaft speed. Speed voltage is similar to the back EMF in a motor. Because of speed voltages, baseband models of filters and reactive elements must have their carrier frequency specified. The carrier frequency is the frequency for which the baseband signals are referenced. For example, the carrier frequency for an RF filter would be the RF frequency while the carrier frequency for an IF filter would be the IF frequency.

**Figure 8-6 Relationship Between Voltage and Current for an Inductor**



The baseband counterparts of the passband filter models are built up from inductors and capacitors modeled in the rotating reference frame.

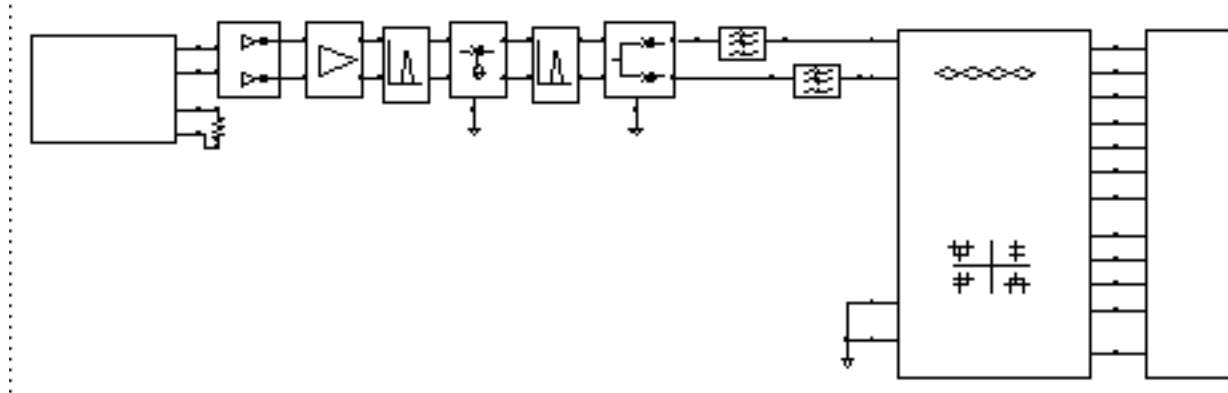
In the complex expression for  $v=Ldi/dt$ , if you replace  $d/dt$  with  $j\omega$ , you find that the impedance of the inductor changes from  $jL\omega$  to  $jL(\omega+\omega_{rf})$ . The same holds for capacitors, which means a filter transfer function,  $H(\omega)$ , has a baseband equivalent equal to  $H(\omega+\omega_{rf})$ . This is simply the original passband transfer function shifted to the left by an amount equal to the carrier frequency. Our time domain baseband models are consistent with the text book frequency domain explanation of baseband modeling.

## Use Model and Design Example

This section describes how to use the baseband models during the architectural design phase. The following examples show you how to



- Construct a baseband model for a simple receiver



- Use the Circuit Optimizer to balance specifications among the function blocks
- Create a passband testbench for the receiver

The design goals were chosen arbitrarily. The example is meant simply to illustrate how to use the library and is a derivative of the design found in [10]. If you find that some parameters are not specified, leave them as default values. You construct the receiver from left to right, from input to output.

## Opening a New Schematic Window

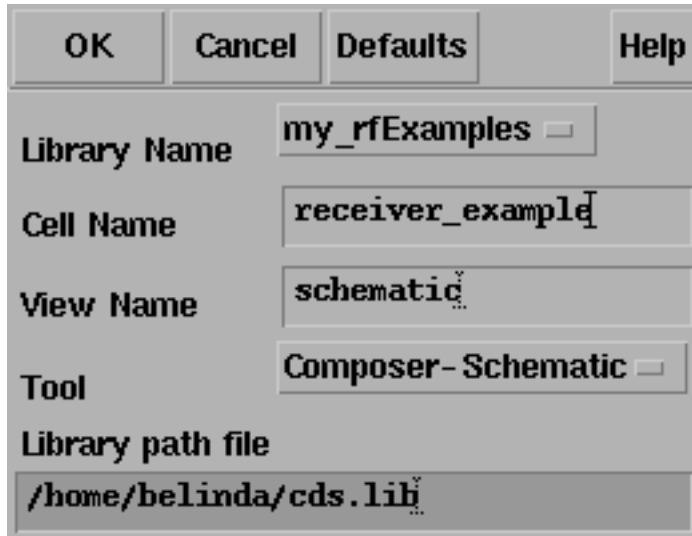
1. In the CIW, choose *File – New – Cellview*.

The Create New File form appears.

2. In the Create New File form,

- a. Choose *my\_rfExamples* in the *Library Name* cyclic field. (Choose the editable copy of *rfExamples* you created as described in [Chapter 3](#).)
- b. Enter *receiver\_example* in the *Cell Name* field.
- c. Select *Composer-Schematic* in the *Tool* cyclic field. *Schematic* appears in the *View Name* field.

- d. The completed form appears like the one below.



The image shows a dialog box with the following fields and buttons:

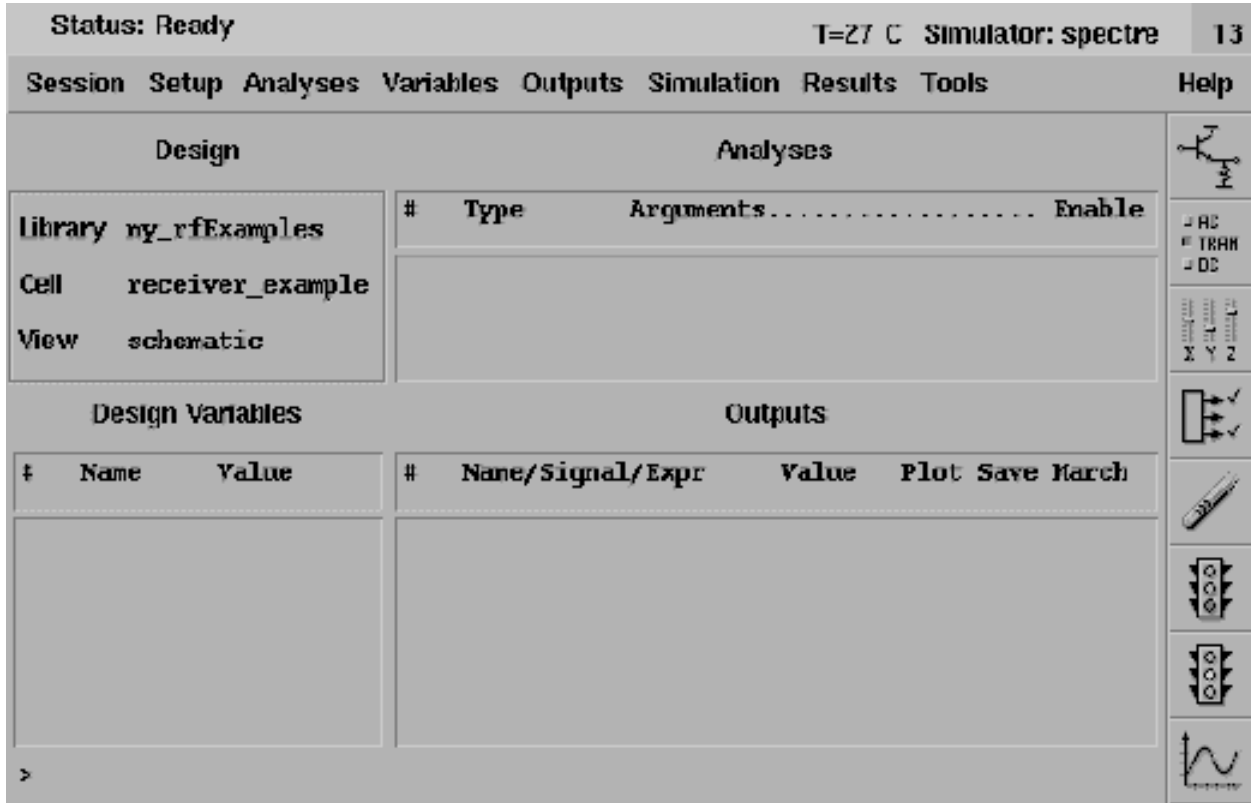
- Buttons: OK, Cancel, Defaults, Help
- Library Name: my\_rfExamples
- Cell Name: receiver\_example
- View Name: schematic
- Tool: Composer-Schematic
- Library path file: /home/belinda/cds.lib

3. Click *OK*.
4. An empty Schematic window for the *receiver\_example* appears.

## Opening the Analog Environment

1. In the Schematic window, choose *Tools – Analog Environment*.

The Simulation window opens.



The *Library*, *Cell*, and *View* names appear in the *Design* section of the Simulation window.

Set the simulator options from the Simulator window as described in [“Choosing Simulator Options”](#) on page 562.

Set up the model libraries from the Simulator window as described in [“Setting Up Model Libraries”](#) on page 564.

## Constructing the Baseband Model for the Receiver

Construct the receiver in the Schematic window by adding blocks from left to right, from input to output, as listed in [Table 8-1](#).

Except for the resistor, ground, and port models (which come from the *analogLib*), all blocks come from the *rfLib*. Unless otherwise instructed, leave the port resistances at their default value of 50 Ohms.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

**Table 8-1 Blocks Used to Create the Receiver**

Block Name and Reference	Element Name	Library and Category
CDMA signal source — See <a href="#">Adding the CDMA Signal Source</a>	CDMA_reverse_xmit	From the <i>measurement</i> category in <i>rfLib</i> .
Resistor—attach to CDMA signal source	res	From <i>analogLib</i>
Driver — See <a href="#">Adding the Driver</a>	BB_driver	From the <i>measurement</i> category in <i>rfLib</i> .
Low noise amplifier — See <a href="#">Adding the Low Noise Amplifier</a>	LNA_BB	From the <i>top_dwnBB</i> category in <i>rfLib</i> .
Butterworth bandpass filter — See <a href="#">Adding a Butterworth Band Pass Filter</a>	BB_butterworth_bp	From the <i>top_dwnBB</i> category in <i>rfLib</i>
RF-to-IF mixer — See <a href="#">Adding an RF-to-IF Mixer</a>	dwn_cnvrt	From the <i>top_dwnBB</i> category in <i>rfLib</i>
Butterworth bandpass filter — See <a href="#">Adding Another Butterworth Bandpass Filter</a>	BB_butterworth_bp	From the <i>top_dwnBB</i> category in <i>rfLib</i>
IQ demodulator — See <a href="#">Adding an IQ Demodulator</a>	IQ_demod_BB	From the <i>top_dwnBB</i> category in <i>rfLib</i>
Butterworth lowpass filters (create two) — See <a href="#">Adding Two Butterworth Lowpass Filters</a>	butterworth_lp	From the <i>top_dwnPB</i> category in <i>rfLib</i>
Instrumentation model — See <a href="#">Adding an Instrumentation Block</a>	offset_comms_instr	From the <i>measurement</i> category in <i>rfLib</i> .
Terminator — See <a href="#">Adding an Instrumentation Terminator</a>	instr_term	From the <i>measurement</i> category in <i>rfLib</i>
Grounds—attach to RF-to-IF mixer, IQ demodulator, and Instrumentation model	gnd	From <i>analogLib</i>

### Adding the CDMA Signal Source

Add the first receiver block, a CDMA signal source (CDMA\_reverse\_xmit), to the schematic.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

In the Schematic window, choose *Add – Instance*.

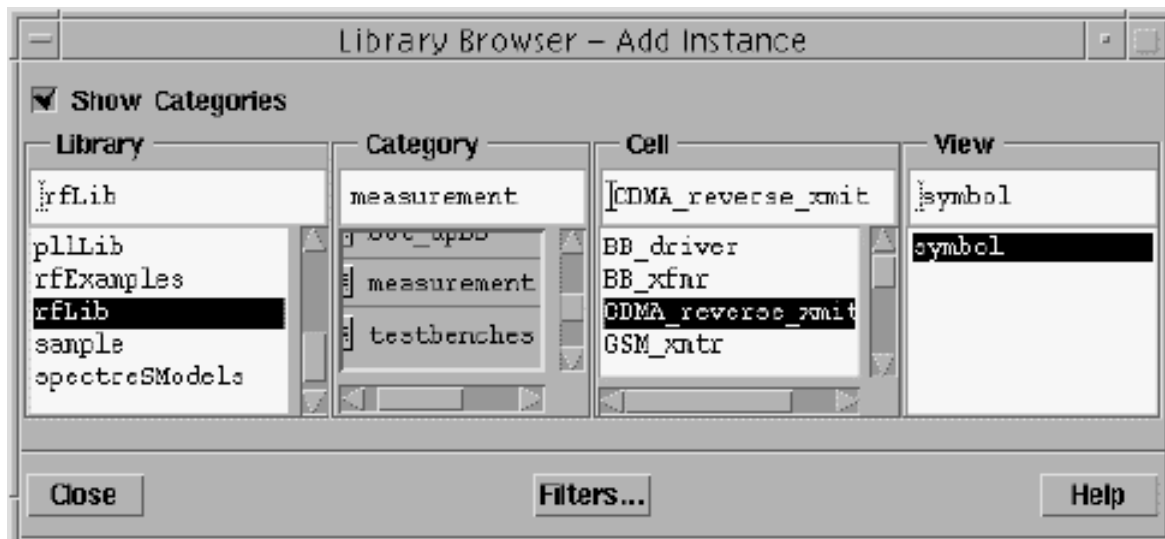
The Add Instance form appears. It may be empty or it may display information for a previously added element. The default for the *View* field is `symbol`.

In the Add Instance form, click *Browse*.

The Library Browser - Add Instance form appears.

In the Library Browser - Add Instance form,

- a. If necessary, click *Show Categories* to display the *Category* column so you can view the elements (or cells) in the *rfLib* by category.
- b. In the *Libraries* column, click *rfLib* to display categories of elements in *rfLib*.
- c. The *Everything* category is displayed by default and all cells in *rfLib* are listed in the *Cells* column. (In the Add Instance form, *rfLib* displays in the *Library* field.)
- d. In the *Category* column, click *measurement* to list only the cells in the *measurement* category.
- e. In the *Cell* column, click *CDMA\_reverse\_xmit*.
- f. In the Library Browser, cell *CDMA\_reverse\_xmit* and its default view *symbol* are both selected.



In the Add Instance form,

- `rfLib` appears in the *Library* field

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

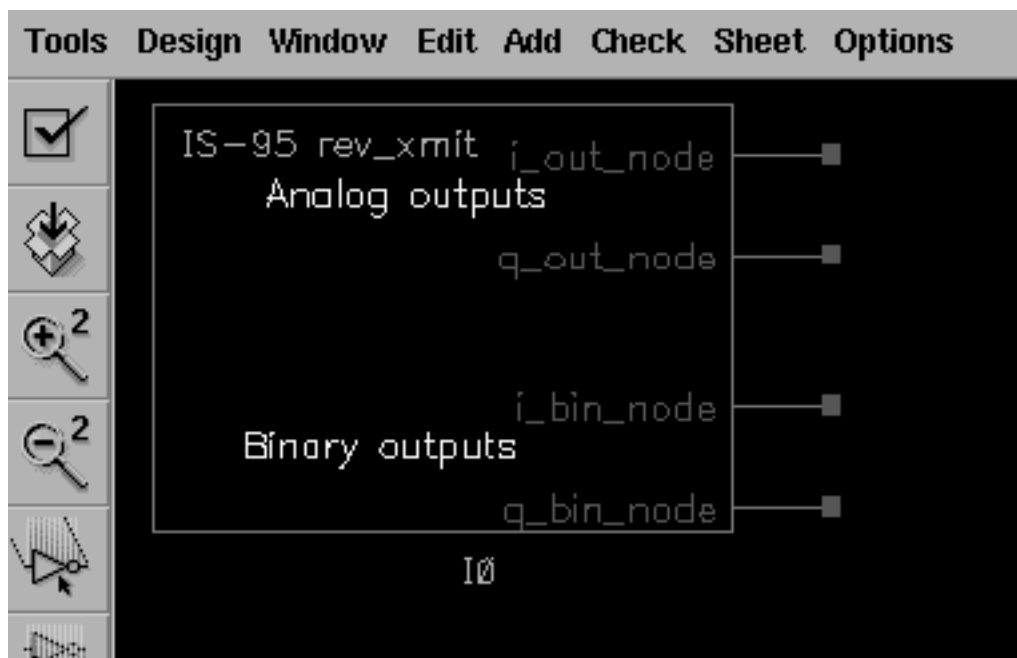
### Methods for Top-Down RF System Design

- ❑ `CDMA_reverse_xmit` displays in the *Cell* field
- ❑ `symbol` displays in the *View* field

The CDF parameters for the element and their default values appear at the bottom of the form.

CDF Parameter of view	Use Tools Filter <input type="checkbox"/>
seed	21
amplitude	1
t-rise_fall,a symbol fraction	1

1. To place a `CDMA_reverse_xmit` block in the schematic, move the cursor over the Schematic window. The outline for the `CDMA_reverse_xmit` symbol is attached to the cursor.
2. Move the cursor near the top left corner of the schematic and click to place the `CDMA_reverse_xmit` block. This block models a CDMA signal source.
3. Click *Esc* to remove the symbol from the cursor.



### **Add a Resistor to the CDMA Signal Source**

Since this example does not use the binary output nodes (`i_bin_node` and `q_bin_node`) on the CDMA signal source, connect a resistor between these nodes to avoid unused pin warnings.

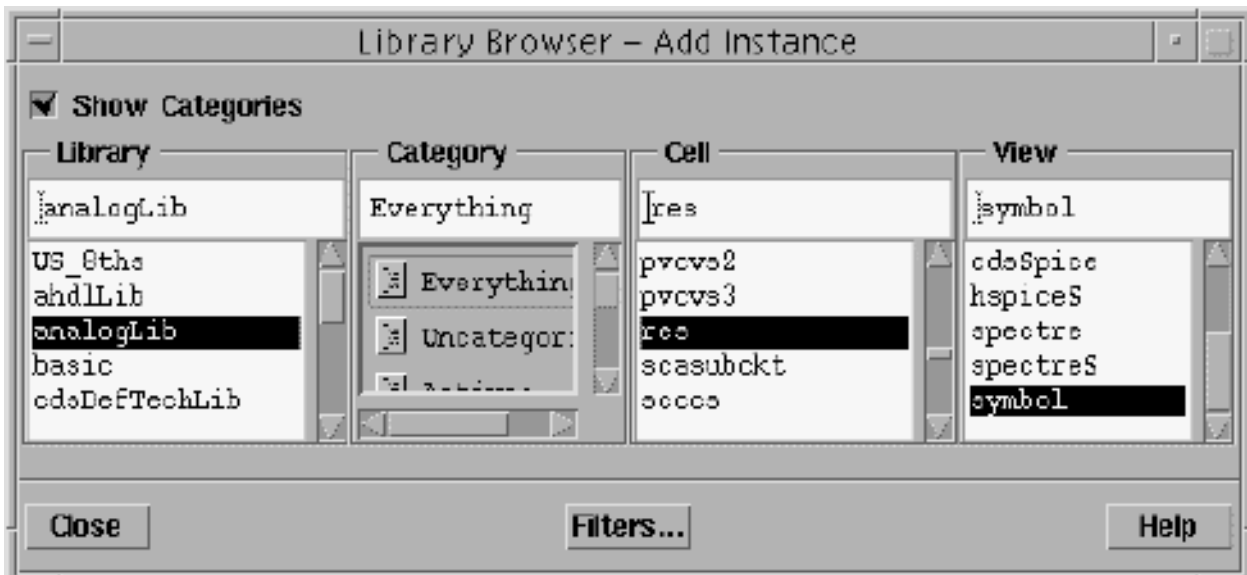
In the *Libraries* column of the Library Browser - Add Instance form, click *analogLib* to display elements in *analogLib*.

If *Show Categories* is selected, the *Everything* category is displayed by default and all cells in *analogLib* are listed in the *Cells* column.

Scroll through the list of cells in *analogLib* to locate the resistor cell, *res*.

Click *res* in the *Cell* column.

The cell *res* and its default view *symbol* are both selected.



In the Add Instance form,

- `analogLib` displays in the *Library* field
- `res` displays in the *Cell* field
- and `symbol` displays in the *View* field

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

The CDF parameters for the element and their default values appear at the bottom of the form.

CDF Parameter	Value
Resistance	1K Ohms
Temperature coefficient 1	
Temperature coefficient 2	
Model name	
Length	
Width	
Resistance Form	
Multiplier	
Scale factor	
Temp rise from ambient	
Generate noise?	<input type="checkbox"/>

1. Move the cursor over the Schematic window.
2. Click to place the top resistor terminal in line with the top binary output node (i\_bin\_out) on the lower right side of the *CDMA\_reverse\_xmit* block
3. Click *Esc* to remove the symbol from the cursor.
4. Wiring the Resistor to the CDMA Signal Source

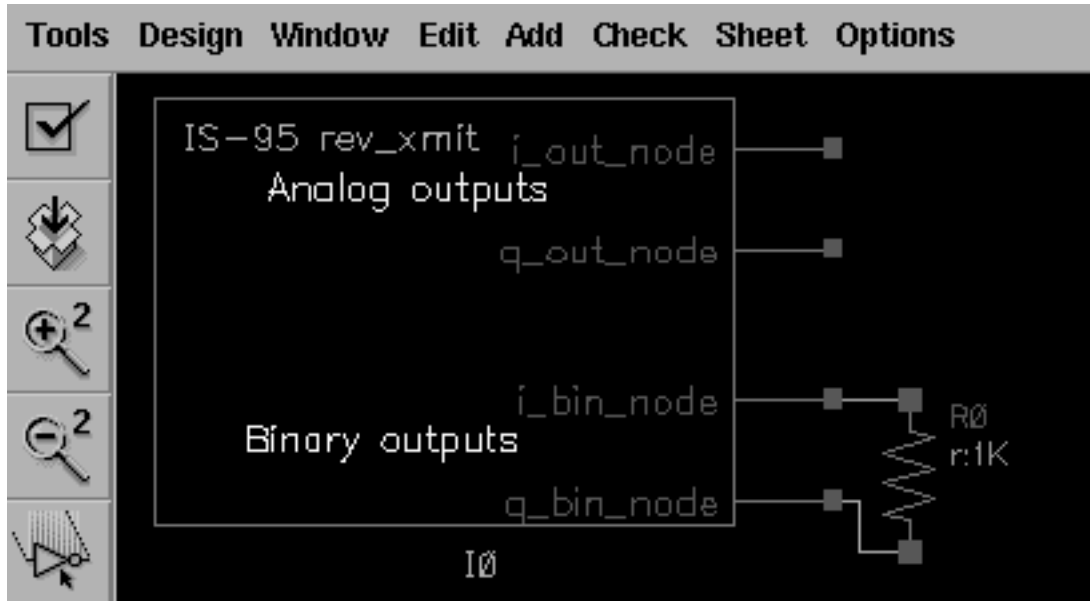
Wire the resistor to the binary outputs, i\_bin\_node and q\_bin\_node , of the CDMA signal source.

1. To wire the resistor to the *CDMA\_reverse\_xmit* block, in the Schematic window choose *Add - Wire (narrow)*.
2. Click i\_bin\_node on the *CDMA\_reverse\_xmit* block then move the cursor and click the top node of the resistor.



3. Click `q_bin_node` on the `CDMA_reverse_xmit` block then move the cursor and click the bottom node of the resistor.
4. Click *Esc* to stop wiring.

The CDMA signal source and resistor wired together appear as follows.



### Adding the Driver

Add a driver block to the right of the CDMA signal source block.

In the Schematic window, choose *Add – Instance* to display the Add Instance form.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

1. In the Library Browser - Add Instance form, make the following selections.

Library	Category	Cell	View
<code>rfLib</code>	<code>measurement</code>	<code>BB_driver</code>	<code>symbol</code>

At the top of the Add Instance form,


- `rfLib` displays in the *Library* field,

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

- ❑ `BB_driver` displays in the *Cell* field and
- ❑ `symbol` displays in the *View* field.
- ❑ The CDF parameters and their default values appear at the bottom of the Add Instance form.



CDF Parameter of view  Use Tools Filter

Output resistance

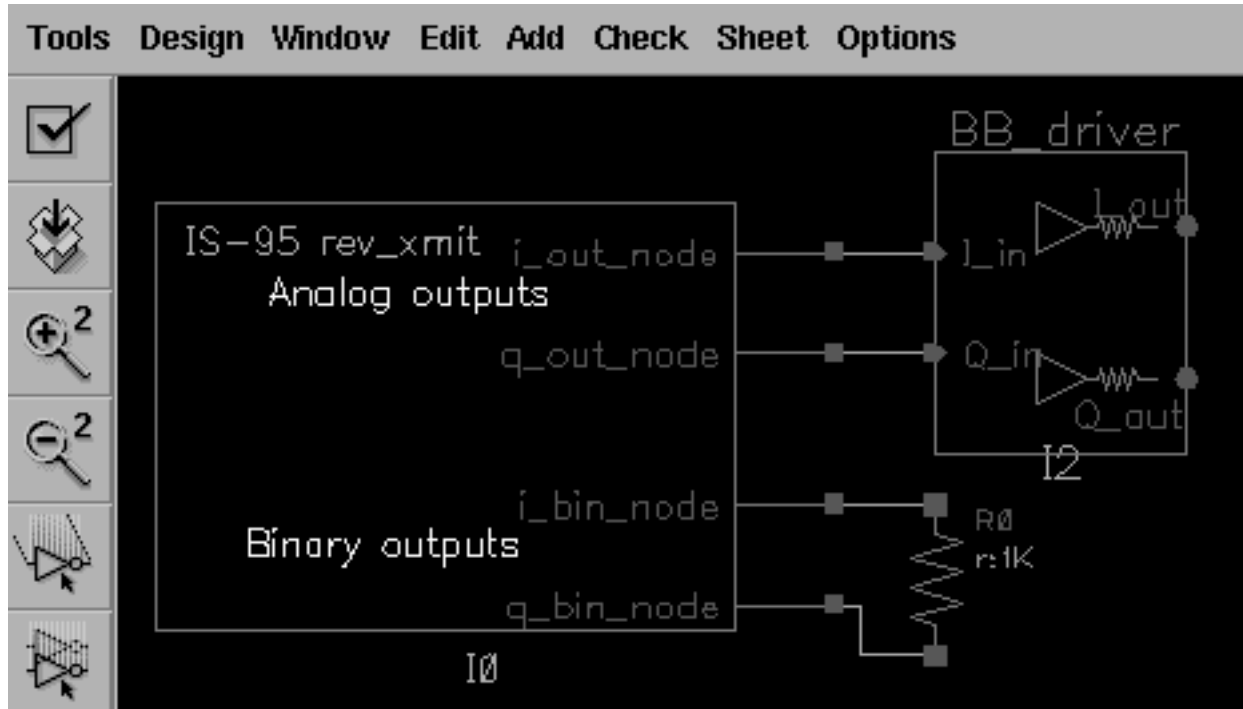
dBm-out @ 1v peak in.

2. Move the cursor over the Schematic window.
3. Click to place the *BB\_driver* to the right of the *CDMA\_reverse\_xmit* block. Align the input pins of the driver with the analog output pins of the CDMA signal source.
4. Click *Esc* to remove the symbol from the cursor.

#### ***Wiring the Signal Source to the Driver***

1. To wire the *BB\_driver* block to the *CDMA\_reverse\_xmit* block, in the Schematic window choose *Add — Wire (narrow)*.
2. Click `i_out_node` on the *CDMA\_reverse\_xmit* block then click `I_in` on the *BB\_driver* block.
3. Click `q_out_node` on the *CDMA\_reverse\_xmit* block then click `Q_in` on the *BB\_driver* block.
4. Click *Esc* to stop wiring.

5. The schematic now appears as follows.



### **Modifying Parameter Values for the Driver**

Edit the value of the *BB\_driver* CDF parameter *dBm-out@1v peak in driver* as follows.

1. Choose *Edit – Properties – Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the values of CDF (component description format) properties for the driver and modify the schematic for this simulation.

2. In the Schematic window, click the *BB\_driver* block.
3. The Edit Object Properties form changes to display information for the *BB\_driver* block
4. Change the *dBm-out@1v peak in* parameter value as follows.

Parameter Name	Value
<i>dBm-out@1v peak in</i>	-16

The driver converts 1 peak volt from the CDMA signal source to  $-16$  dBm referenced to the output resistance of the driver.

### **Adding the Low Noise Amplifier**

Add a low noise amplifier to the right of the driver.

In the Schematic window, choose *Add—Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form, make the following selections. (If necessary, click *Show Categories* at the top of the Library Browser, to display the *Category* column.)

<b>Library</b>	<b>Category</b>	<b>Cell</b>	<b>View</b>
rfLib	top_dwnBB	LNA_BB	symbol

In the Library Browser, cell *LNA\_BB* and its default view *symbol* are both selected.

At the top of the Add Instance form,

- `rfLib` displays in the *Library* field
- `LNA_BB` displays in the *Cell* field
- `symbol` displays in the *View* field

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

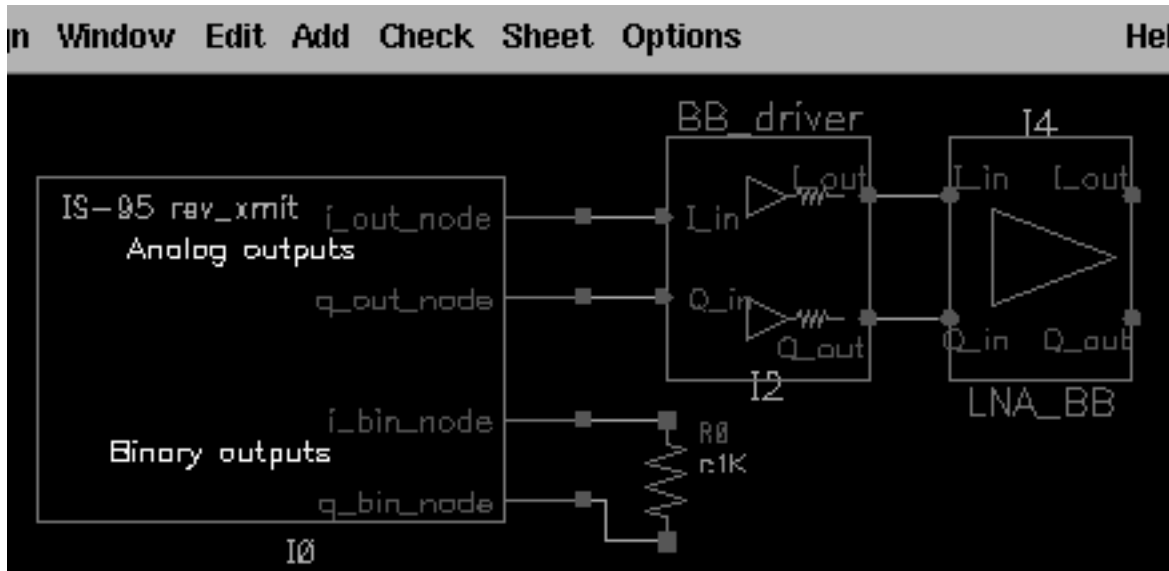
---

The CDF parameters for the element and their default values display at the bottom of the LNA\_BB Add Instance form.

CDF Parameter of view	Use Tools Filter <input type="checkbox"/>
Available pwr gain[dB]	40
input resistance	50
output resistance	50
input referred IP3[dBm]	-30
noise figure [dB]	0
am/pm sharpness	2
cmp[dBm]	-30
radians  @ cmp	.7
radians  @ big input	2
{1,0,-1} for {cw,none,ccw}	0

1. Move the cursor over the Schematic window. The outline for the LNA symbol is attached to the cursor. Align the input pins of the LNA with the output pins of the driver.
2. Click to place the *LNA\_BB* block to the right of the *BB\_driver* block.
3. Click *Esc* to remove the symbol from the cursor.
4. Wiring the Driver to the LNA
5. To wire the *LNA\_BB* block to the *BB\_driver* block, in the Schematic window choose *Add - Wire (narrow)*.
6. Click *I\_out* on the *BB\_driver* block then click *I\_in* on the *LNA\_BB* block.
7. Click *Q\_out* on the *BB\_driver* block then click *Q\_in* on the *LNA\_BB* block.
8. Click *Esc* to stop wiring.

9. The schematic now appears as follows.



### **Modifying Parameter Values for the LNA**

1. Edit the CDF parameter values for the LNA.
2. Choose *Edit—Properties—Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for the LNA and modify the schematic for this simulation.

- a. In the Schematic window, click the *LNA*.
- b. The Edit Object Properties form changes to display information for the LNA.
- c. Change the CDF parameter values for the LNA as follows.

Parameter Name	Value
<i>Available pwr gain [dB]</i>	lna_gain
<i>Input resistance</i>	50
<i>Output resistance</i>	300
<i>Input referred IP3 [dBm]</i>	lna_ip3
<i>Noise figure [dB]</i>	10

Parameter Name	Value
<i>am/pm sharpness</i>	2
<i>cmp [dBm]</i>	lna_ip3
<i> radians  @ cmp</i>	.05
<i> radians  @ big input</i>	.7
<i>{1, 0, -1} for {cw, none, ccw}</i>	1

### Adding a Butterworth Band Pass Filter

Add a Butterworth band pass filter to the right of the low noise amplifier.

In the Schematic window, choose *Add—Instance* to display the Add Instance form.

Information for the LNA is still displayed in the form.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form, make the following selection.

Library	Category	Cell	View
<i>rfLib</i>	<i>top_dwnBB</i>	<i>BB_butterworth_bp</i>	<i>symbol</i>

In the Library Browser, cell *BB\_butterworth\_bp* and its default view are both selected.

At the top of the Add Instance form,

- rfLib* displays in the *Library* field
- BB\_butterworth\_bp* displays in the *Cell* field
- symbol* displays in the *View* field.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

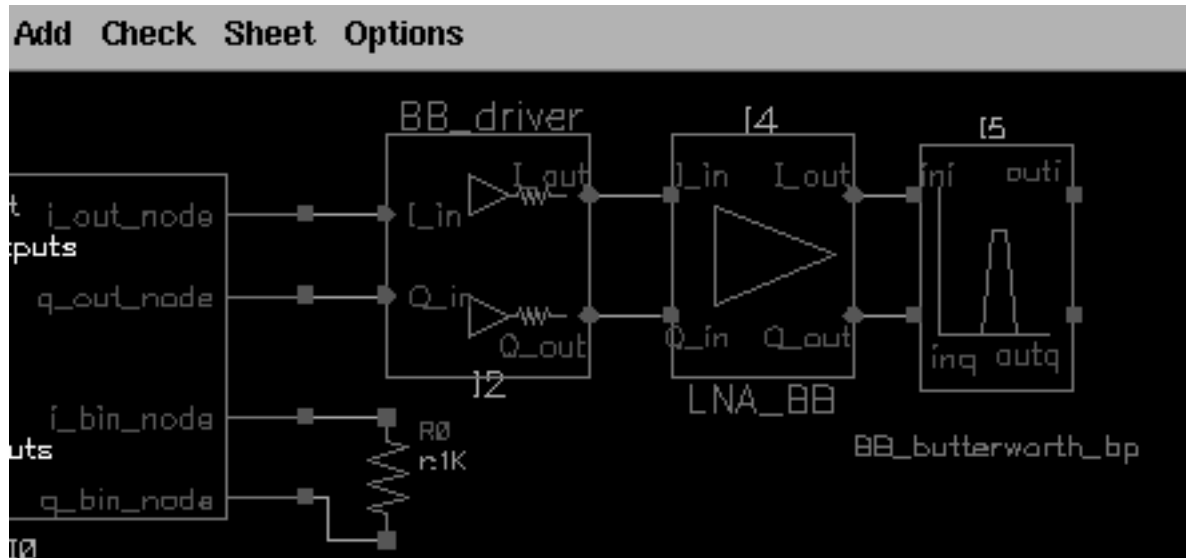
- The CDF parameters for the Butterworth band pass filter and their default values display at the bottom of the Add Instance form.

Parameter	Value
Filter Order	3
Input impedance	50
Output impedance	50
Center frequency (Hz)	1e9
Relative bandwidth	0.1
Insertion loss (dB)	0
carrier frequency	

1. Move the cursor over the Schematic window. The outline for the filter symbol is attached to the cursor. Align the input pins of the filter with the output pins of the LNA.
2. Click to place the *BB\_butterworth\_bp* block to the right of the *LNA\_BB* block.
3. Click *Esc* to remove the symbol from the cursor.
4. Wiring the LNA to the Filter
5. To wire the *BB\_butterworth\_bp* block to the *LNA\_BB* block, in the Schematic window choose *Add—Wire (narrow)*.
6. Click *I\_out* on the *LNA\_BB* block then click *ini* on the *BB\_butterworth\_bp* block.
7. Click *Q\_out* on the *LNA\_BB* block then click *inq* on the *BB\_butterworth\_bp* block.
8. Click *Esc* to stop wiring.



9. The schematic now appears as follows.



### **Modifying Parameter Values for the Band Pass Filter**

Edit the CDF parameter values for the Butterworth band pass filter.

Choose *Edit—Properties—Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for the filter and modify the schematic for this simulation.

1. In the Schematic window, click the filter.
2. The Edit Object Properties form changes to display information for the filter.
3. Edit the parameter values to match those in Table 8-2.
4. Click *OK*.

**Table 8-2 CDF Parameter Values for the Butterworth Filter**

Parameter Name	Value
<i>Filter order</i>	3
<i>Input impedance</i>	50
<i>Output impedance</i>	50

**Table 8-2 CDF Parameter Values for the Butterworth Filter**

Parameter Name	Value
<i>Center frequency (Hz)</i>	<i>frf</i>
<i>Relative bandwidth</i>	<i>rf_rbw</i>
<i>Insertion loss (dB)</i>	3
<i>Carrier frequency</i>	<i>frf</i>

Specify the *Carrier frequency* parameter value for the baseband equivalent model of the Butterworth band pass filter, just as you do for any reactive element. As shown in [Figure 8-6](#) on page 576, the carrier frequency is used to compute speed voltages. Since filters are built up from inductors and capacitors which have speed voltages, you must specify the carrier frequency for filters.

When a filter follows an RF-to-IF mixer, its *Carrier frequency* parameter value is the IF frequency.

The *Carrier frequency* is the frequency value to which the baseband signals are referenced.

The *Center frequency* is the frequency for which a filter is designed. The *Center frequency* parameter value for a bandpass filter does not have to equal the *Carrier frequency* parameter value.

### Adding an RF-to-IF Mixer

Add an RF-to-IF mixer (*dwn\_cnvr*) block to the right of the bandpass filter block.

In the Schematic window, choose *Add—Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form, make the selections indicated in [Table 8-3](#).

**Table 8-3 Library Browser selections for the RF to IF Mixer**

Library	Category	Cell	View
<i>rfLib</i>	<i>top_dwnBB</i>	<i>dwn_cnvr</i>	<i>symbol</i>

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

In the Library Browser, cell *dwn\_cnvr* (the RF-to-IF mixer) and its default view *symbol* are both selected.

At the top of the Add Instance form,

- `rflib` displays in the *Library* field
- `dwn_cnvr` displays in the *Cell* field
- `symbol` displays in the *View* field.

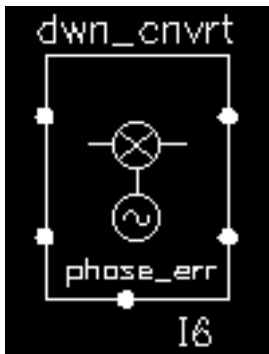
The CDF parameters for the down converter and their default values display at the bottom of the form.

CDF Parameter of view	Use Tools Filter <input type="checkbox"/>
available power gain[dB]	40
input resistance	50
output resistance	50
input referred IP3[dBm]	-30
noise figure [dB]	0
RF frequency	1e9
LO frequency	0.9e9
AM/PM input point[dBm]	-30
phase shift at cmp[rad]	.7
phase shift at infinity	2
sharpness factor	2
{1,0,-1} = {cw,none,ccw}	0

1. Move the cursor over the Schematic window. The outline for the RF-to-IF Mixer symbol is attached to the cursor. Align the input pins of the mixer with the output pins of the filter.
2. Click to place the *dwn\_cnvr* block to the right of the *BB\_butterworth\_bp* block.

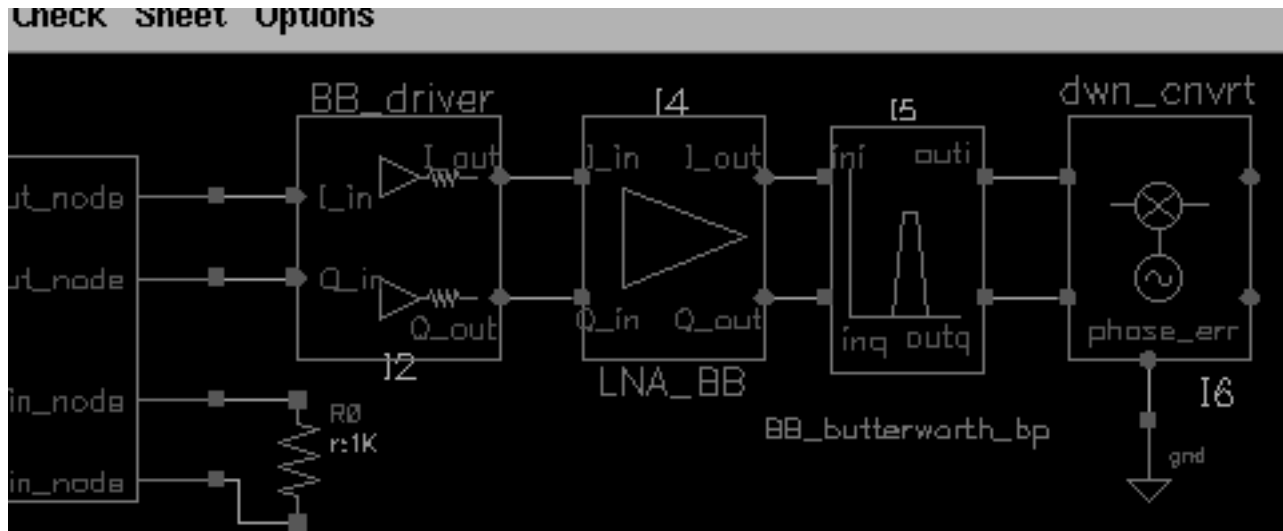
3. Click *Esc* to remove the symbol from the cursor.
4. Grounding the phase\_err Pin on the Mixer

It is necessary to ground the phase error (*phase\_err*) pin on the bottom of the RF-to-IF mixer.



1. In the Schematic window, choose *Add – Instance* to display the Add Instance form.  
The Add Instance form appears. It may be empty or it may display information for a previously added element.  
In the Add Instance form, type
  - `analogLib` in the *Library* field
  - `gnd` in the *Cell* field
  - `symbol` in the *View* field.
  - Move the cursor over the Schematic window.
  - Click to place the ground terminal in line with the phase error node (`phase_err`) on the bottom of the `dwn_cnvt` block.
  - Click *Esc* to remove the symbol from the cursor.
  - Wiring the Filter and Ground to the Mixer
2. To wire the `dwn_cnvt` block to the `BB_butterworth_bp` block and the ground, in the Schematic window choose *Add - Wire (narrow)*.
3. Click `out_i` on the `BB_butterworth_bp` block then click `I_in` on the `dwn_cnvt` block.
4. Click `out_q` on the `BB_butterworth_bp` block then click `Q_in` on the `dwn_cnvt` block.
5. Click the port on the `gnd` block then click `phase_err` on the `dwn_cnvt` block.
6. Click *Esc* to stop wiring.

7. The schematic now appears as follows.



### Modifying Parameter Values for the RF-to-IF Mixer

1. Edit the CDF parameter values for the RF-to-IF mixer (*dwn\_cnvr*) as listed in [Table 8-4](#) on page 597.

a. Choose *Edit – Properties – Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for *dwn\_cnvr* and modify the schematic for this simulation.

b. In the Schematic window, click *dwn\_cnvr*.

c. The Edit Object Properties form changes to display information for *dwn\_cnvr*.

d. Change the parameter values to match those in [Table 8-4](#).

e. Click OK.

**Table 8-4 CDF Parameter Values for the RF-to-IF Mixer**

Parameter Name	Value
available power gain[dB]	if_mx_gain
Input resistance	50
output resistance	50

**Table 8-4 CDF Parameter Values for the RF-to-IF Mixer**

Parameter Name	Value
input referred ip3[dBm]	if_mx_ip
noise figure [dB]	10
RF frequency	frf
LO frequency	flo1
AM/PM input point[dBm]	-30
phase shift at cmp[rad]	.7
phase shift at infinity	2
sharpness factor	2
{1,0,-1} = {cw,none,ccw}	0

### Adding Another Butterworth Bandpass Filter

Add another Butterworth band pass filter block to the right of the RF-to-IF Mixer block.

In the Schematic window, choose *Add – Instance* to display the Add Instance form.

Information for the mixer is still displayed in the form and the outline of the mixer is still attached to the cursor.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form, make the following selection.

Library	Category	Cell	View
<i>rfLib</i>	<i>top_dwnBB</i>	<i>BB_butterworth_bp</i>	<i>symbol</i>

In the Library Browser, cell *BB\_butterworth\_bp* and its default view are both selected.

At the top of the Add Instance form,

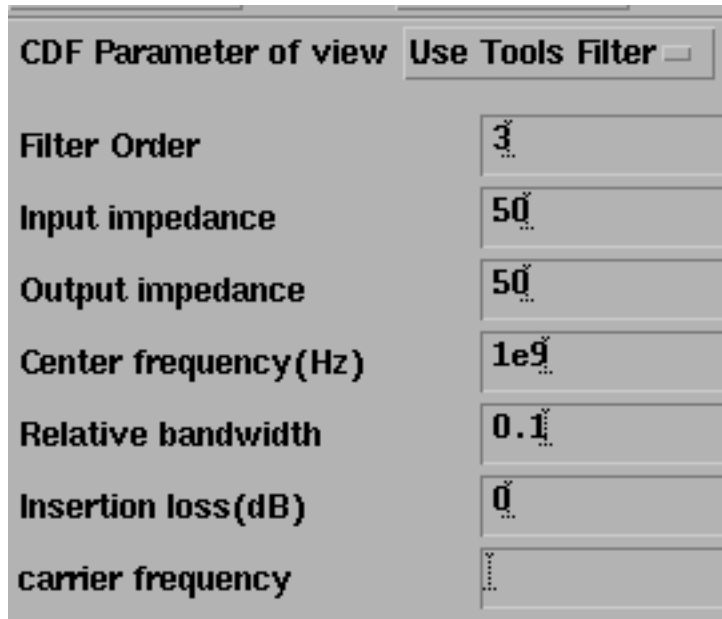
- `rfLib` displays in the *Library* field
- `BB_butterworth_bp` displays in the *Cell* field
- `symbol` displays in the *View* field.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

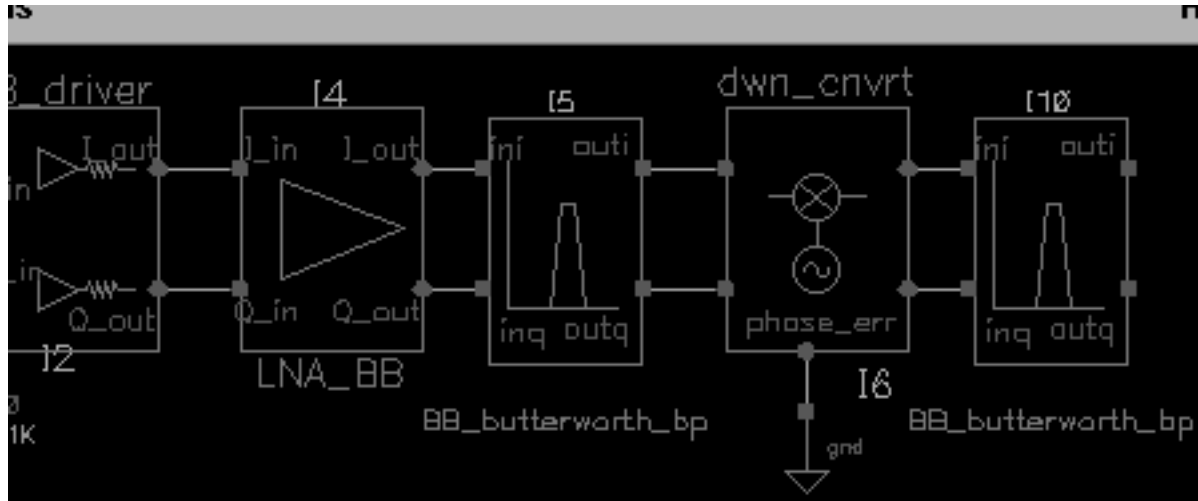
- The CDF parameters for the Butterworth band pass filter and their default values display at the bottom of the Add Instance form.



Parameter	Value
Filter Order	3
Input impedance	50
Output impedance	50
Center frequency (Hz)	1e9
Relative bandwidth	0.1
Insertion loss (dB)	0
carrier frequency	

1. Move the cursor over the Schematic window. The outline for the filter symbol is attached to the cursor. Align the input pins of the filter with the output pins of the LNA.
2. Click to place the *BB\_butterworth\_bp* block to the right of the *LNA\_BB* block.
3. Click *Esc* to remove the symbol from the cursor.
4. Wiring the Mixer to the Filter
5. To wire the *BB\_butterworth\_bp* block to the *dwn\_cnvrt* block, in the Schematic window choose *Add - Wire (narrow)*.
6. Click *I\_out* on the *dwn\_cnvrt* block then click *ini* on the *BB\_butterworth\_bp* block.
7. Click *Q\_out* on the *dwn\_cnvrt* block then click *inq* on the *BB\_butterworth\_bp* block.
8. Click *Esc* to stop wiring.

9. The schematic now appears as follows.



**Modifying Parameter Values for the Band Pass Filter**

1. Edit the CDF parameter values for the Butterworth band pass filter as listed in [Table 8-5](#) on page 600.

a. Choose *Edit – Properties – Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for the filter and modify the schematic for this simulation.

b. In the Schematic window, click the filter.

c. The Edit Object Properties form changes to display information for the filter.

d. Change the parameter values to match those in [Table 8-5](#).

e. Click OK.

**Table 8-5 CDF Parameter Values for the Second Butterworth Filter**

Parameter Name	Value
Filter Order	3
Input impedance	50
Output impedance	50
Center frequency (Hz)	-frf+f1o1



**Table 8-5 CDF Parameter Values for the Second Butterworth Filter**

Parameter Name	Value
Relative bandwidth	if_rbw
Insertion loss (dB)	1
Carrier frequency	-frf+flol

As for the first band pass filter, specify the carrier frequency for the baseband equivalent model of the Butterworth band pass filter, just as you do for any reactive element. As shown in [Figure 8-6](#) on page 576, the carrier frequency is used to compute speed voltages. Since filters are built up from inductors and capacitors which have speed voltages, you must specify the carrier frequency for filters.

When a filter follows an RF-to-IF mixer, its carrier frequency is the IF frequency. The carrier frequency is the frequency to which the baseband signals are referenced. The center frequency of the bandpass filter does not have to equal the carrier frequency. The center frequency is the frequency for which a filter is designed.

### ***Adding an IQ Demodulator***

Add an IQ Demodulator (*IQ\_demod\_BB*) block to the right of the bandpass filter block.

In the Schematic window, choose *Add – Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form,

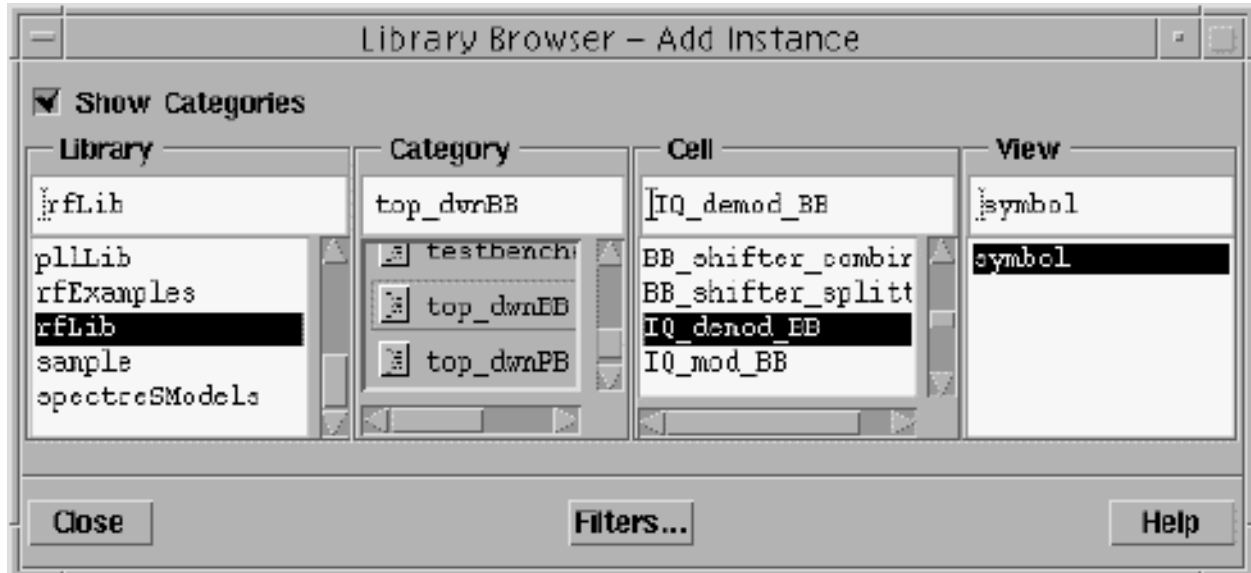
- a. If necessary, click *Show Categories* to display the *Category* column.
- b. Click *rfLib* to display elements in *rfLib*. The *Everything* category is displayed by default and all cells in *rfLib* are listed in the *Cells* column.
- c. In the *Category* column, click *top\_dwnBB* to display cells in the *top\_dwnBB* category.
- d. In the *Cell* column, click *IQ\_demod\_BB*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

- e. In the Library Browser, cell *IQ\_demod\_BB* (the IQ demodulator) and its default view *symbol* are both selected.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

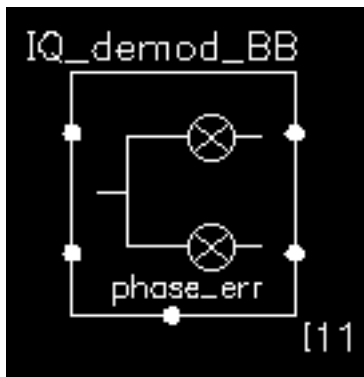
---

At the top of the Add Instance form, `rfLib` displays in the *Library* field, `IQ_demod_BB` displays in the *Cell* field and `symbol` displays in the *View* field. The CDF parameters for the IQ demodulator and their default values display at the bottom of the form.

CDF Parameter of view		Use Tools Filter <input type="checkbox"/>
available I-mixer gain[dB]		40
available Q-mixer gain[dB]		40
input resistance		50
output resistance		50
I- [dBm] input referred IP3		-30
Q- [dBm] input referred IP3		-30
noise figure [dB]		0
quadrature error		0
I-sharpness factor		2
Q-sharpness factor		2
I_cmp		-30
Q_cmp		-30
I-radians@I_cmp		.7
Q-radians@Q_cmp		.7
I-radians@big I-input		2
Q-radians@big Q-input		2
I {1,0,-1} for {cw,none,ccw}		0
Q {1,0,-1} for {cw, none,ccw}		0

1. Move the cursor over the Schematic window. The outline for the IQ demodulator symbol is attached to the cursor. Align the input pins of the IQ demodulator with the output pins of the filter and click to place the *IQ\_demod\_BB* block to the right of the second *BB\_butterworth\_bp* block. Click *Esc* to remove the symbol from the cursor.
2. Grounding the phase\_err Pin on the IQ Demodulator

It is necessary to ground the phase error (*phase\_err*) pin on the bottom of the IQ demodulator.



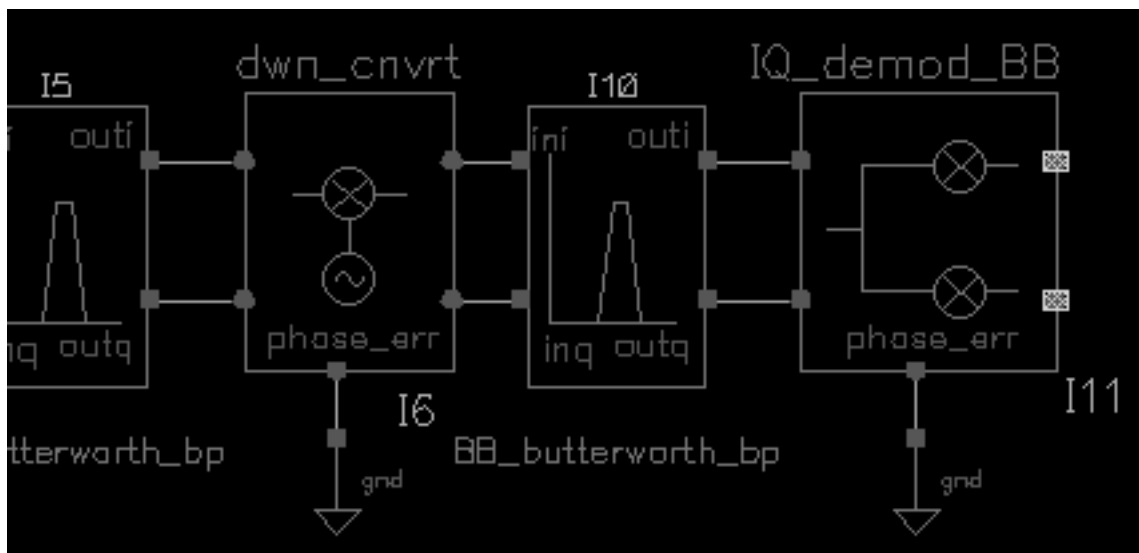
1. In the Schematic window, choose *Add – Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

In the Add Instance form, type

- `analogLib` in the *Library* field
  - `gnd` in the *Cell* field
  - `symbol` in the *View* field.
  - Move the cursor over the Schematic window.
  - Click to place the ground terminal in line with the phase error node (*phase\_err*) on the bottom of the *IQ\_demod\_BB* block.
  - Click *Esc* to remove the symbol from the cursor.
  - Wiring the Filter and Ground to the IQ Demodulator
2. To wire the *IQ\_demod\_BB* block to the *BB\_butterworth\_bp* block and the ground, in the Schematic window choose *Add - Wire (narrow)*.

3. Click `out_i` on the `BB_butterworth_bp` block then click `I_in` on the `IQ_demod_BB` block.
4. Click `out_q` on the `BB_butterworth_bp` block then click `Q_in` on the `IQ_demod_BB` block.
5. Click the port on the `gnd` block then click `phase_err` on the `IQ_demod_BB` block.
6. Click `Esc` to stop wiring.
7. The schematic now appears as follows.



### **Modifying Parameter Values for the IQ Demodulator**

1. Edit the CDF parameter values for the IQ Demodulator (`IQ_demod_BB`) as listed in [Table 8-6](#) on page 606.
  - a. Choose *Edit – Properties – Objects* in the Schematic window.
 

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for `IQ_demod_BB` and modify the schematic for this simulation.
  - b. In the Schematic window, click `IQ_demod_BB`.
  - c. The Edit Object Properties form changes to display information for `IQ_demod_BB`.
  - d. Change the parameter values to match those in [Table 8-6](#).

e. Click OK.

**Table 8-6 CDF Parameter Values for the IQ Demodulator**

Parameter Name	Value
available I-mixer gain[dB]	0
available Q-mixer gain[dB]	0
Input resistance	50
output resistance	50
I-[dBm] input referred IP3	40
Q-[dBm] input referred IP3	40
noise figure [dB]	2
quadrature error	0
I-sharpness factor	2
Q-sharpness factor	2
I_cmp	-30
Q_cmp	-30
I-radians@I_cmp	.7
Q-radians@Q_cmp	.7
I-radians@big I-input	2
Q-radians@big Q-input	2
I {1,0,-1} for {cw,none,ccw}	0
Q {1,0,-1} for {cw,none,ccw}	0

### Adding Two Butterworth Lowpass Filters

1. Add two Butterworth low pass filters to the right of the IQ demodulator block.
2. Align one filter with the demodulator's `i_out` pin.
3. Align the other filter with its `q_out` pin.
4. In the Schematic window, choose *Add – Instance* to display the Add Instance form.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

5. In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.
6. In the Library Browser - Add Instance form, make the following selections.

Library	Category	Cell	View
<i>rfLib</i>	<i>top_dwnPB</i>	<i>butterworth_lp</i>	<i>symbol</i>

In the Library Browser, cell *butterworth\_lp* and it's default view are both selected.

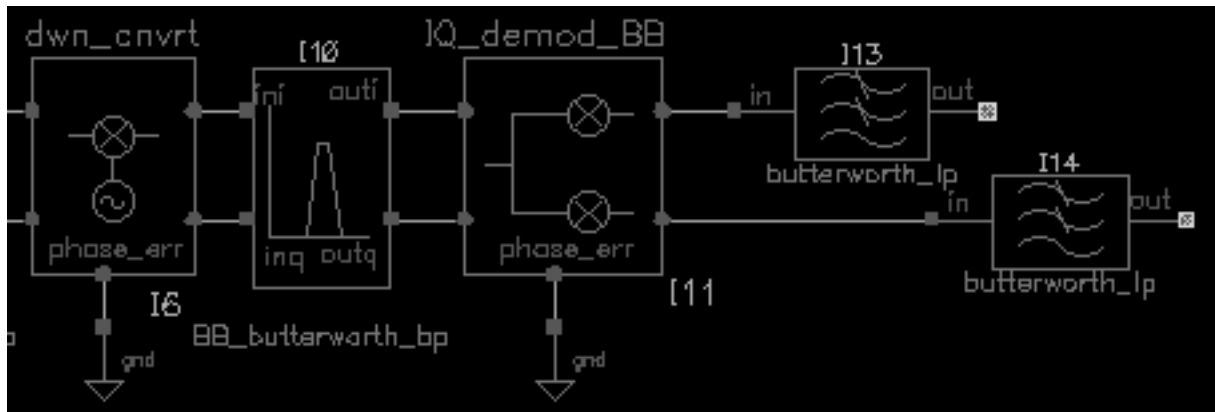
At the top of the Add Instance form,

- `rfLib` displays in the *Library* field
- `butterworth_lp` displays in the *Cell* field
- `symbol` displays in the *View* field.
- The CDF parameters for the Butterworth low pass filter and their default values display at the bottom of the Add Instance form.

<b>Filter Order</b>	3
<b>Input impedance</b>	50
<b>Output impedance</b>	50
<b>Corner frequency(Hz)</b>	1e9
<b>Insertion loss(dB)</b>	0

7. Move the cursor over the Schematic window. The outline for the butterworth low pass filter symbol is attached to the cursor.
8. Align the *in* pin of the first butterworth low pass filter with the *I\_out* pin (the top pin) of the IQ demodulator and click to place the filter close to the demodulator. Align the *in* pin of the second butterworth low pass filter with the *Q\_out* pin (the bottom pin) of the IQ demodulator. You have to place it further from the demodulator to align it with the *Q\_out* pin.
9. Click *Esc* to remove the symbol from the cursor.
10. Wiring the IQ Demodulator to the Filters

11. To wire the *IQ\_demod\_BB* block to the *butterworth\_lp* blocks, in the Schematic window choose *Add - Wire (narrow)*.
12. Click *I\_out* on the *IQ\_demod\_BB* block then click *in* on the first *butterworth\_lp* block.
13. Click *Q\_out* on the *IQ\_demod\_BB* block then click *in* on the second *butterworth\_lp* block.
14. Click *Esc* to stop wiring.
15. The schematic now appears as follows.



### **Modifying Parameter Values for Both Low Pass Filters**

1. Edit the CDF parameter values for the Butterworth low pass filters as listed in [Table 8-7](#) on page 609.
  - a. Choose *Edit – Properties – Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for each filter and modify the schematic for this simulation.
  - b. In the Schematic window, click the first low pass filter.
  - c. The Edit Object Properties form changes to display information for the filter.



- d. Change the parameter values to match those in Table 8-7.

**Table 8-7 CDF Parameter Values for the Butterworth Low Pass Filters**

Parameter Name	Value
Filter Order	3
Input impedance	50
Output impedance	50
Corner frequency (Hz)	10M
Insertion loss (dB)	0

- e. Click Apply.
- f. In the Schematic window, click the second low pass filter.
- g. The Edit Object Properties form displays the information you entered for the filter as shown in Table 8-7.
- h. Click OK.
- i. Adding an Instrumentation Block

Add an instrumentation block (*offset\_comms\_instr*) block to the right of the low pass filter blocks.

1. In the Schematic window, choose *Add – Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form, make the following selections. (If necessary, click Show Categories at the top of the Library Browser, to display the Category column.)

Library	Category	Cell	View
<i>rfLib</i>	<i>measurement</i>	<i>offset_comms_instr</i>	<i>symbol</i>

In the Library Browser, cell *offset\_comms\_instr* and its default view *symbol* are both selected.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

At the top of the Add Instance form,

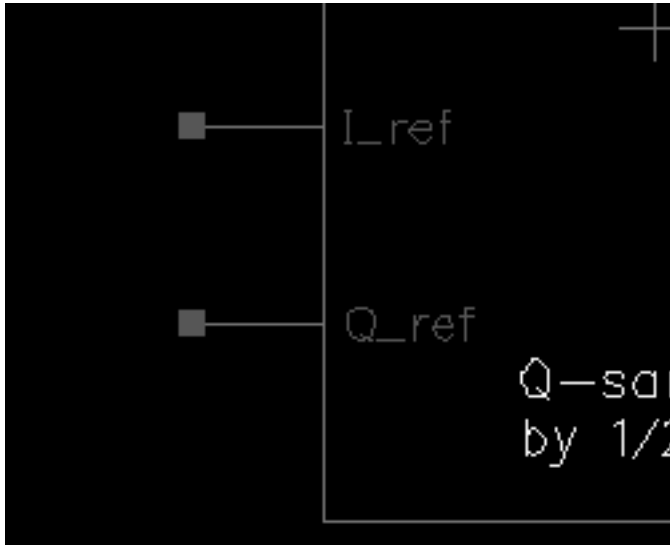
- ❑ `rfLib` displays in the *Library*
- ❑ `offset_comms_instr` displays in the *Cell* field
- ❑ `symbol` displays in the *View* field

The CDF parameters for the `offset_comms_instr` and their default values display at the bottom of the form.

CDF Parameter of view	Use Tools Filter <input type="checkbox"/>
symbols per second	1228800
I-sampling delay (secs)	
number of symbols	2
max eye-diaq volts	1
min eye volts	-1
number of hstgm bins	100
I-noise (volts^2)	0
Q-noise (volts^2)	0
statistics start time	0
input resistance	10e6

2. Move the cursor over the Schematic window. The outline for the `offset_comms_instr` block symbol is attached to the cursor. Align the `I_in` and `Q_in` pins of the `offset_comms_instr` block with the `out` pins of the butterworth low pass filters and click to place the `offset_comms_instr` block to the right of the low pass filter blocks.
3. Click `Esc` to remove the symbol from the cursor.
4. Grounding the Reference Pins on the Instrumentation Block

It is necessary to ground the reference pins (*I\_ref* and *Q\_ref*) pin near the lower left corner of the instrumentation block.



### ***Grounding the phase\_err Pin on the Mixer***

1. In the Schematic window, choose *Add – Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

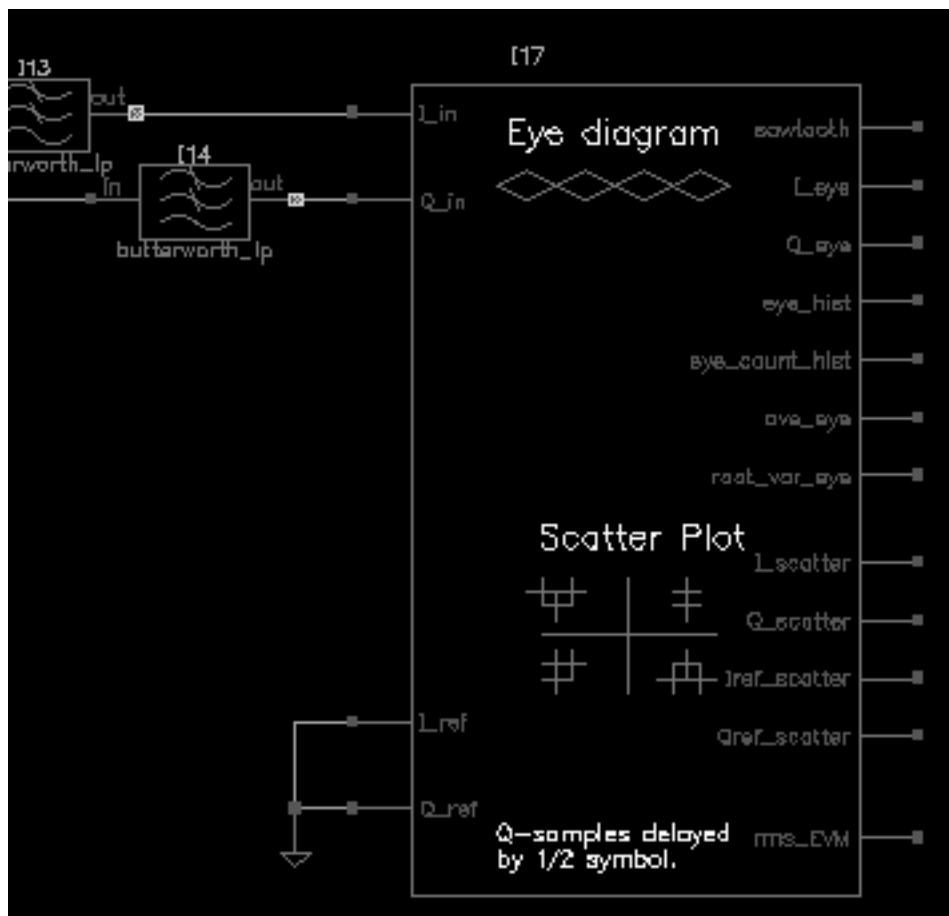
In the Add Instance form, type

- `analogLib` in the *Library* field
  - `gnd` in the *Cell* field
  - `symbol` in the *View* field.
  - Move the cursor over the Schematic window.
  - Click to place the ground terminal in line with the `Q_ref` node on the bottom of the `offset_comms_instr` block.
  - Click *Esc* to remove the symbol from the cursor.
  - Wiring the Filter and Ground to the Instrumentation Block
2. To wire the low pass filters to the `offset_comms_instr` block and the ground, in the Schematic window choose *Add - Wire (narrow)*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

3. Click `out` on the upper `butterworth_lp` block (the low pass filter connected to the `I_out` node on the IQ demodulator) then click `I_in` on the `offset_comms_instr` block.
4. Click `out` on the lower `butterworth_lp` block (the low pass filter connected to the `Q_out` node on the IQ demodulator) then click `Q_in` on the `offset_comms_instr` block.
5. Click the port on the `gnd` block then click the `Q_ref` node on the `offset_comms_instr` block.
6. Click the port on the `gnd` block then click the `I_ref` node on the `offset_comms_instr` block.
7. Click `Esc` to stop wiring.
8. The schematic now appears as follows.



### ***Modifying Parameter Values for the Instrumentation Block***

1. Edit the CDF parameter values for the *offset\_comms\_instr* as listed in [Table 8-8](#) on page 613.
  - a. Choose *Edit – Properties – Objects* in the Schematic window.

The Edit Object Properties form appears. You use this form to change the list of CDF (component description format) properties for *offset\_comms\_instr* and modify the schematic for this simulation.
  - b. In the Schematic window, click the *offset\_comms\_instr* block.
  - c. The Edit Object Properties form changes to display information for *offset\_comms\_instr* block.
  - d. Change the parameter values to match those in [Table 8-8](#).
  - e. Click OK.

**Table 8-8 CDF Parameter Values for the Instrumentation Block**

<b>Parameter Name</b>	<b>Value</b>
symbols per second	1228800
I-sampling delay (secs)	134n
number of symbols	2
max eye-diag volts	1
min eye volts	-1
number of hstgm bins	100
I-noise (volts <sup>2</sup> )	0
Q-noise (volts <sup>2</sup> )	0
statistics start time	30u
input resistance	50

### **Adding an Instrumentation Terminator**

Add an instrumentation termination (*instr\_term*) block to the right of the instrumentation block. The *instr\_term* block terminates the outputs on the instrumentation block and prevents unused pin warnings.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

In the Schematic window, choose *Add – Instance* to display the Add Instance form.

The Add Instance form appears. It may be empty or it may display information for a previously added element.

In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

In the Library Browser - Add Instance form, make the following selections. (If necessary, click Show Categories at the top of the Library Browser, to display the Category column.)

Library	Category	Cell	View
<i>rfLib</i>	<i>measurement</i>	<i>instr_term</i>	<i>symbol</i>

In the Library Browser, cell *instr\_term* and it's default view *symbol* are both selected.

At the top of the Add Instance form,

- `rfLib` displays in the *Library*
- `instr_term` displays in the *Cell* field
- `symbol` displays in the *View* field

There are no CDF parameters for the *instr\_term* cell.

Move the cursor over the Schematic window. The outline for the *instr\_term* symbol is attached to the cursor. Move the *instr\_term* block to the right of the instrumentation block and align the input pins of the instrumentation termination block with the output pins of the instrumentation block.

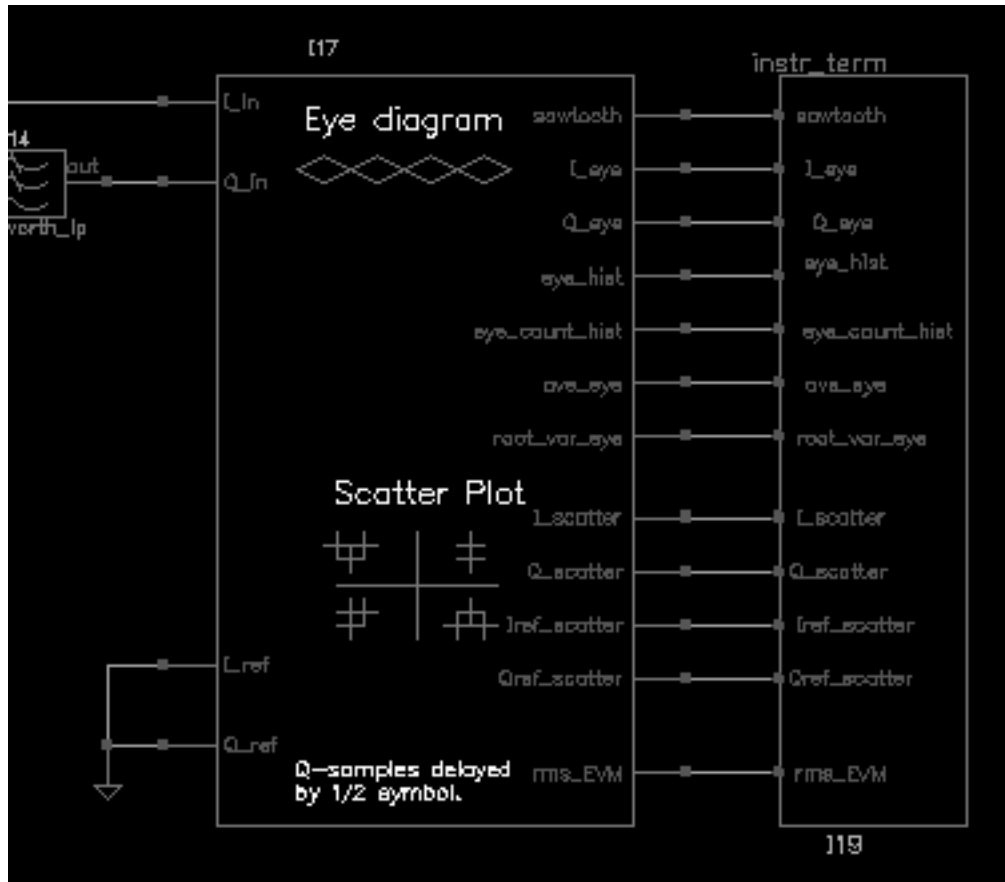
Click to place the *instr\_term* block.

Click *Esc* to remove the symbol from the cursor.

### Wiring the Termination Block to the Instrumentation Block

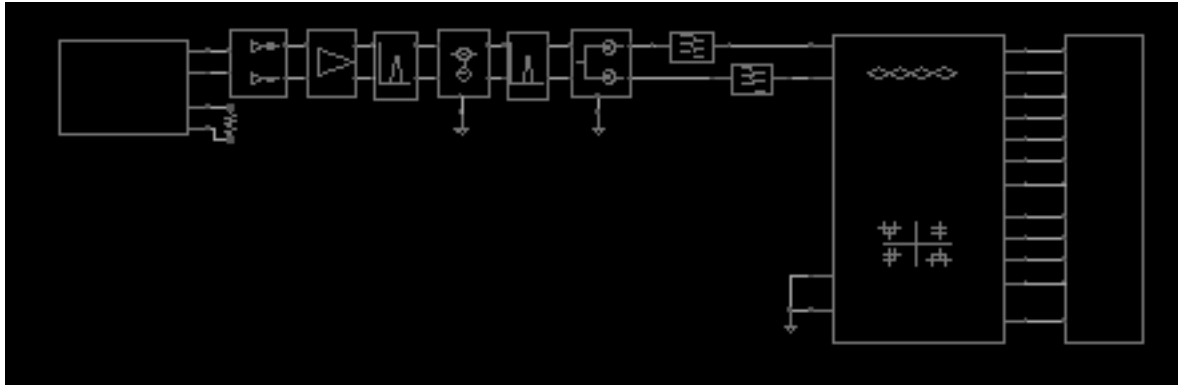
1. To wire the instrumentation (*offset\_comms\_instr*) block to the Instrumentation terminator (*instr\_term*) block, in the Schematic window choose *Add - Wire (narrow)*.
2. Wire the aligned pins straight across.
3. Click *Esc* to stop wiring.

The schematic should look as follows.



4. In the Schematic window, choose *Design — Design Check and Save*.
5. The completed schematic is verified and saved.
6. The schematic for the complete receiver model should look like the one in [Figure 8-7](#) on page 616.

**Figure 8-7 Completed receiver model**



### Setting Variable Values for the Receiver Schematic

Copy the variables you entered as CDF parameters for the individual blocks from the receiver schematic to the Simulation window. Then edit each variable to give it the value specified in [Table 8-9](#) on page 617.

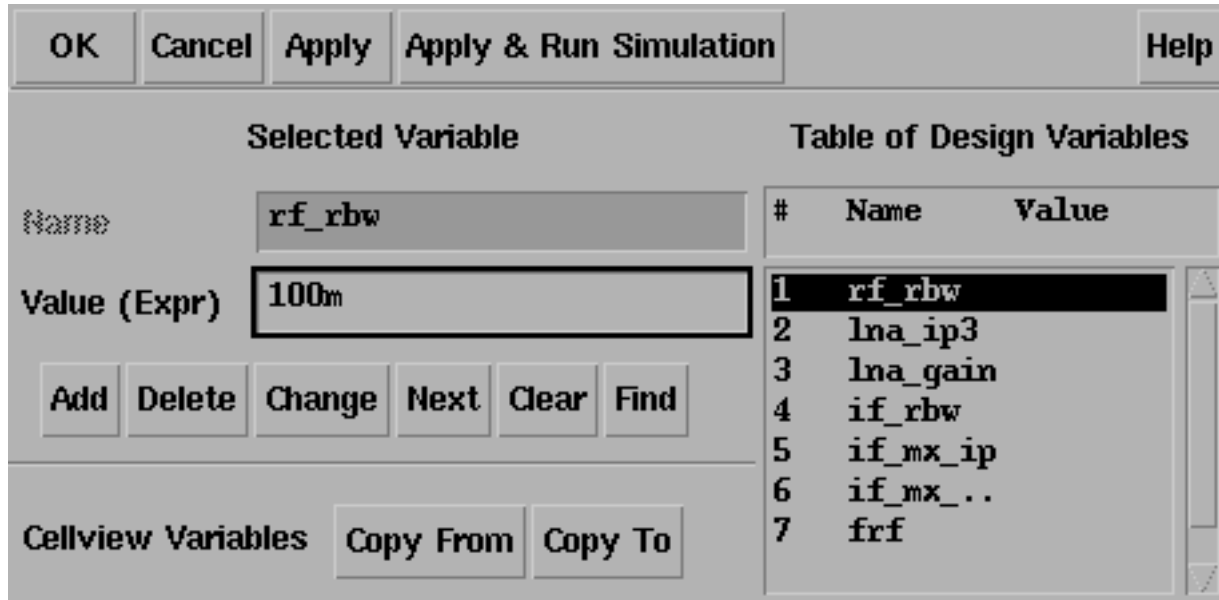
In the Simulation window, use *Variables — Copy From Cellview* to copy the variables from the receiver schematic to the Design Variables area on the Simulation window.

The copied variables display in the *Design Variables* area in the Simulation window.

Design Variables		
#	Name	Value
1	rf_rbw	
2	lna_ip3	
3	lna_gain	
4	if_rbw	
5	if_mx_ip	
6	if_mx_..	



1. In the Simulation window, choose *Variables — Edit* to open the Editing Design Variables form.



### Adding the Values to the Copied Variables

In the Editing Design Variables form, one by one, select each variable in the *Table of Design Variables* and associate with each one, the value listed in Table 8-9.

**Table 8-9 Values for Receiver Variables**

Variable	Value
lna_gain	15
lna_ip3	-5
if_mx_gain	10
if_mx_ip	35
frf	2.14G
flo1	2.354G
if_rbw	200m
rf_rbw	100m

To associate a value with a design variable

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

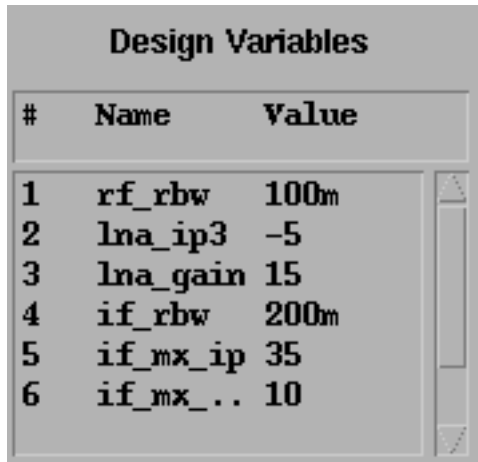
---

1. In the *Table of Design Variables*, click `lna_gain`.
2. `lna_gain` displays in the *Name* field.
3. In the *Value (Expr)* field, enter the number `15`, the value from [Table 8-9](#) on page 617.
4. Click *Change* to list the variable name and its value from the *Table of Design Variables*.

Selected Variable		Table of Design Variables		
Name	Value (Expr)	#	Name	Value
lna_gain	15	1	rf_rbw	
		2	lna_ip3	
		3	lna_gain	
		4	if_rbw	
		5	if_mx_ip	
		6	if_mx_..	
		7	frf	

5. Repeat these steps for the remaining variables listed in the *Table of Design Variables* to associate the values from [Table 8-9](#) on page 617 with the variable names.
6. Click *OK* in the Editing Design Variables form after you have added all the variable values.

7. The table of *Design Variables* in the Simulation window is updated and the Editing Design Variables form is closed.



The image shows a screenshot of a window titled "Design Variables". It contains a table with three columns: "#", "Name", and "Value". The table lists six variables:

#	Name	Value
1	rf_rbw	100m
2	lna_ip3	-5
3	lna_gain	15
4	if_rbw	200m
5	if_mx_ip	35
6	if_mx_..	10

## Setting Up and Running a Transient Analysis

1. In the Simulation window, choose *Analyses—Choose* to display the Choosing Analyses form.
2. In the Choosing Analyses form, if necessary, click *tran* to select a transient analysis.
3. In the Choosing Analyses form, enter 130u in the *Stop Time* field.

4. Highlight *moderate* for *Accuracy Defaults (errpreset)*.

OK Cancel Defaults Apply Help

Analysis  tran  dc  ac  noise  
 xf  sens  dcmatch  stb  
 sp  envlp  pss  pac  
 pnoise  pxf  psp  qpss  
 qpac  qpnoise  qpxf  qpssp

Transient Analysis

Stop Time 130u

Accuracy Defaults (errpreset)  
 conservative  moderate  liberal

Enabled  Options...

5. In the Choosing Analyses form, click *Options* to display the Transient Options form.

6. In the Transient Options form, enter 30u in the *outputstart* field.

SIMULATION INTERVAL PARAMETERS

start

outputstart 30u

autostop  yes  no

By delaying the output start, you remove start-up transients from the eye-diagrams and scatter plots.

Click *OK* in the Transient Options form.

7. Click *OK* in the Choosing Analyses form.
8. If you have not already done so, set up the simulator and model libraries with the following steps.
  - a. Set the simulator options from the Simulator window as described in “[Choosing Simulator Options](#)” on page 562.
  - b. Set up the model libraries from the Simulator window as described in “[Setting Up Model Libraries](#)” on page 564.
  - c. In the Simulation window, choose *Simulation—Netlist and Run*.
  - d. Messages display in the CIW. The simulation log window opens. Watch for messages stating that the simulation has completed successfully.
  - e. Watch the CIW for messages stating the simulation is running and that it has completed successfully.

## **Examining the Results: Eye Diagram, Histogram, and Scatter Plot**

In this section we examine the results of the transient analysis of the receiver.

### **Plotting the Eye Diagram (and Transient Response)**

First plot an eye diagram.

In the Simulation window, choose *Results—Direct Plot—Transient Signal*.

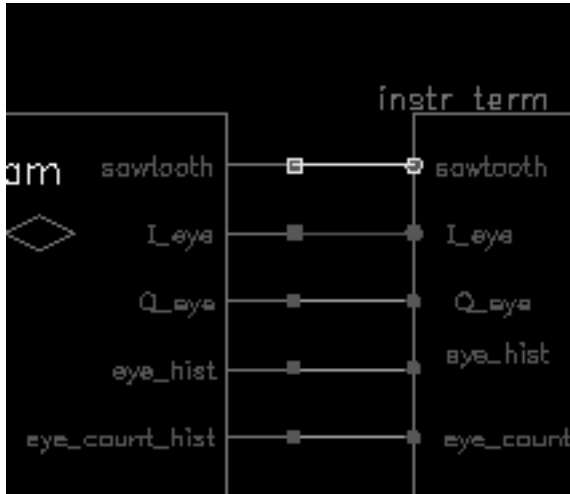
This displays the Waveform window.

1. Following the prompts at the bottom of the Waveform window,  
> *Select nodes or terminals, press <esc> to finish selection*

In the Schematic window:

- a. Click the *sawtooth* net from the instrumentation (*offset\_comms\_instr*) block.

- b. Click the *I\_eye* net from *offset\_comms\_instr* block.



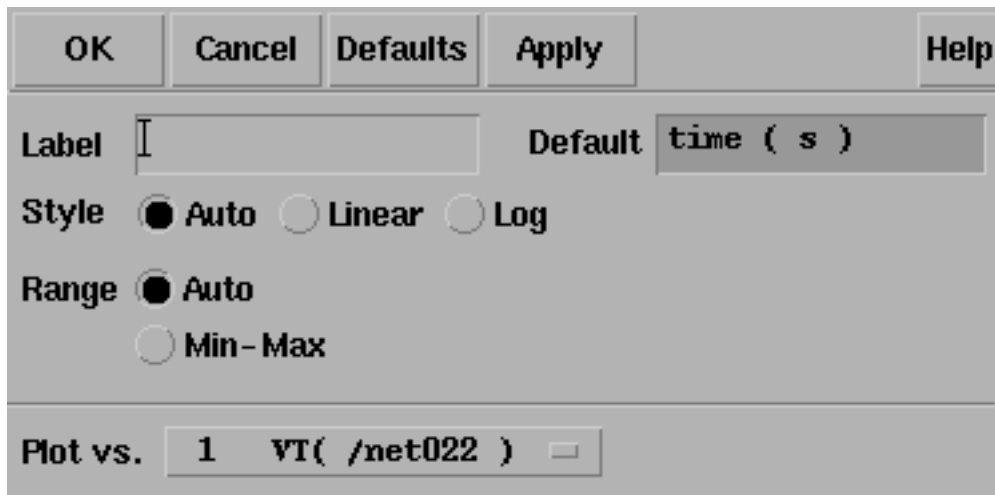
- c. Press *Esc* to indicate that you have finished selecting outputs.

This creates a plot of Transient Response in the Waveform window.

In the Waveform window, select *Axes — X Axis*.

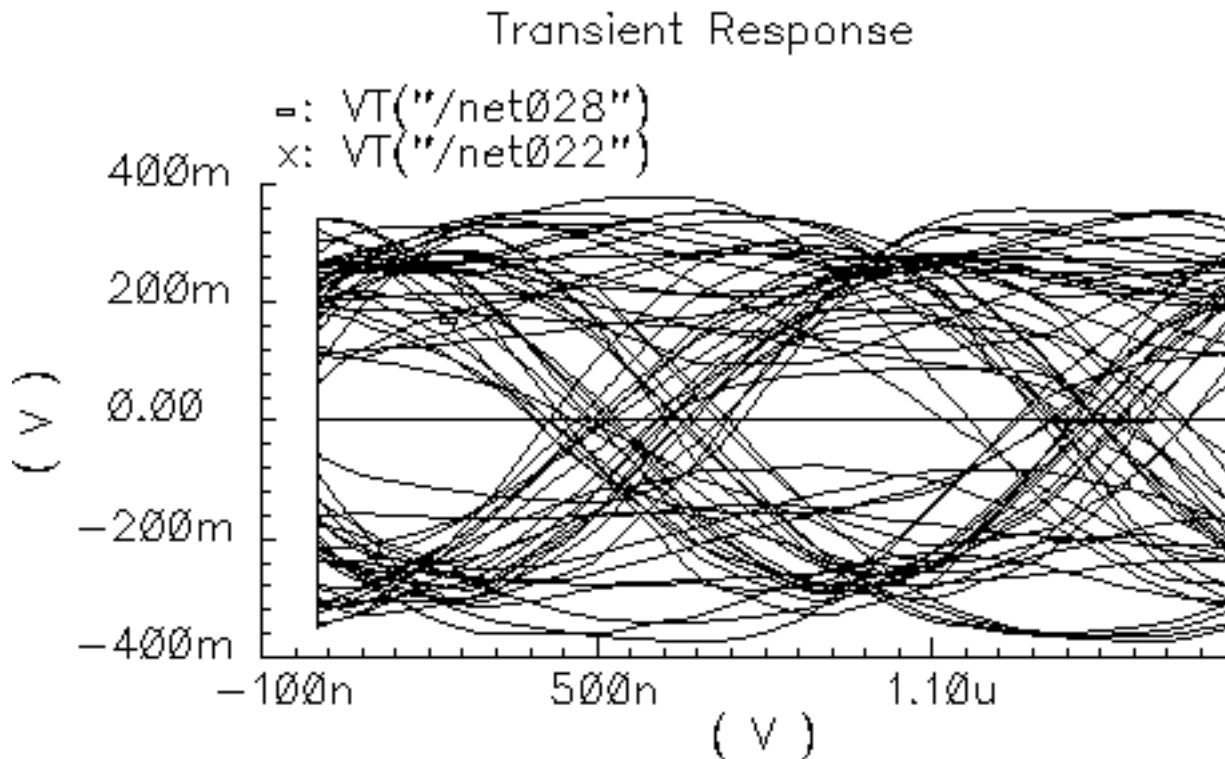
The X Axis form displays. In the X Axis form:

- d. In the *Plot vs. cyclic* field, select *1 VT( /net022 )* to plot the *sawtooth* output on the x-axis. (The net number you see may be different.)



- e. Click *OK*.
- f. You should see the eye-diagram shown in [Figure 8-8](#) on page 623.

Figure 8-8 Eye-diagram



The I-sampling delay parameter in the instrumentation block ( $I\_del$ ) is chosen with respect to this eye diagram. The delay is the time when the eye opens the widest.

The instrumentation block samples the input waveforms with this delay to compute all statistics and to produce scatter plots.

2. In the Waveform window, choose *Window — Close*.

### Generate the Histogram

Now generate a histogram of the I-voltage at the sampling times.

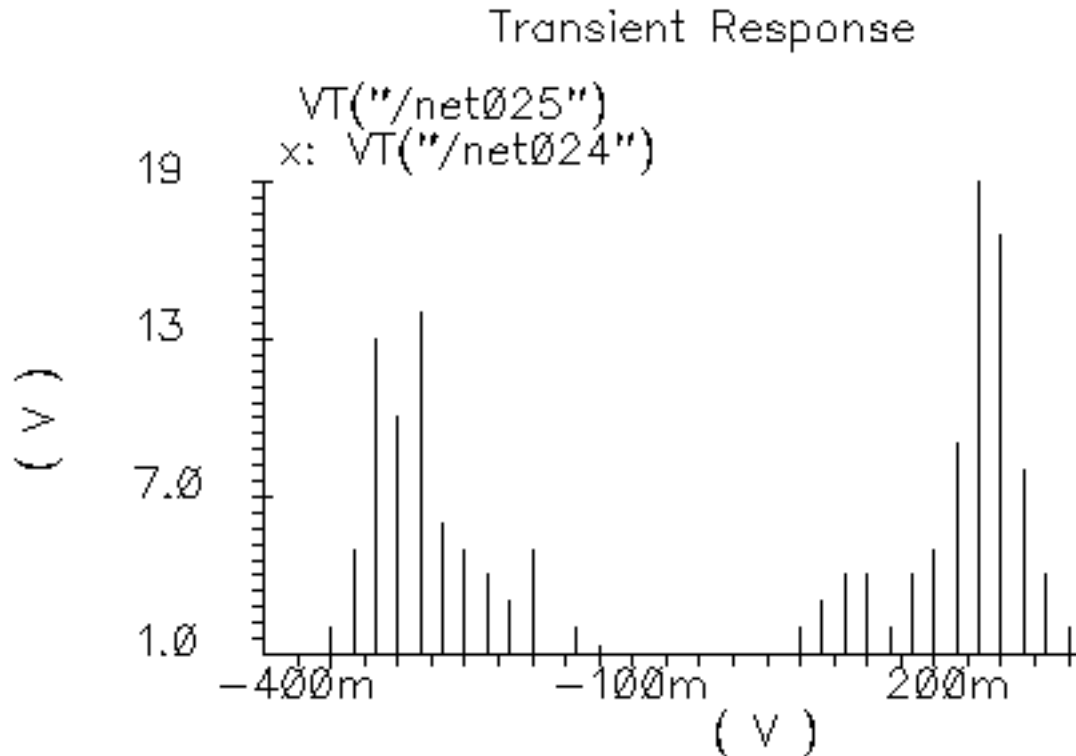
1. In the Waveform window, choose *Window—Reset* to clear the Waveform window.
2. In the Simulation window, choose *Results—Direct Plot—Transient Signal*.
3. This displays the Waveform window.
4. Following the prompts at the bottom of the Waveform window,

In the Schematic window:

- a. Click the *eye\_hist* net from the instrumentation (*offset\_comms\_instr*) block.
- b. Click the *eye\_count\_hist* net from *offset\_comms\_instr*.
- c. Press *Esc* to indicate that you have finished selecting outputs.
- d. This creates a plot in the Waveform window.
- e. In the Waveform window, select *Axes — X Axis*.
- f. The X Axis form displays. In the X Axis form:
- g. In the *Plot vs. cyclic* field, select *1 VT( /net029 )* to plot the *eye\_hist* output on the x-axis.
- h. Click *OK*.
- i. This creates a plot in the Waveform window.
- j. In the Waveform window, select *Curves — Options*.
- k. The Plot Style form displays. In the Plot Style form,
  - l. In the *Plotting Style* cyclic field, select *bar*.
- m. In the *Number of Ticks* field, enter 0.
- n. Click *OK*.
- o. You should see the histogram in [Figure 8-9](#) on page 625.



Figure 8-9 Histogram



5. In the Waveform window, choose *Window — Close*.

## Generating the Scatter Plot

### Generate a scatter plot of the received symbols.

In the Simulation window, choose *Results—Direct Plot—Transient Signal*.

This displays the Waveform window.

1. Following the prompts at the bottom of the Waveform window,

In the Schematic window:

- a. Click the *I\_scatter* output from the instrumentation (*offset\_comms\_instr*) block.
- b. Click the *Q\_scatter* output from *offset\_comms\_instr*.
- c. Click *Esc* to indicate that you have finished selecting outputs.

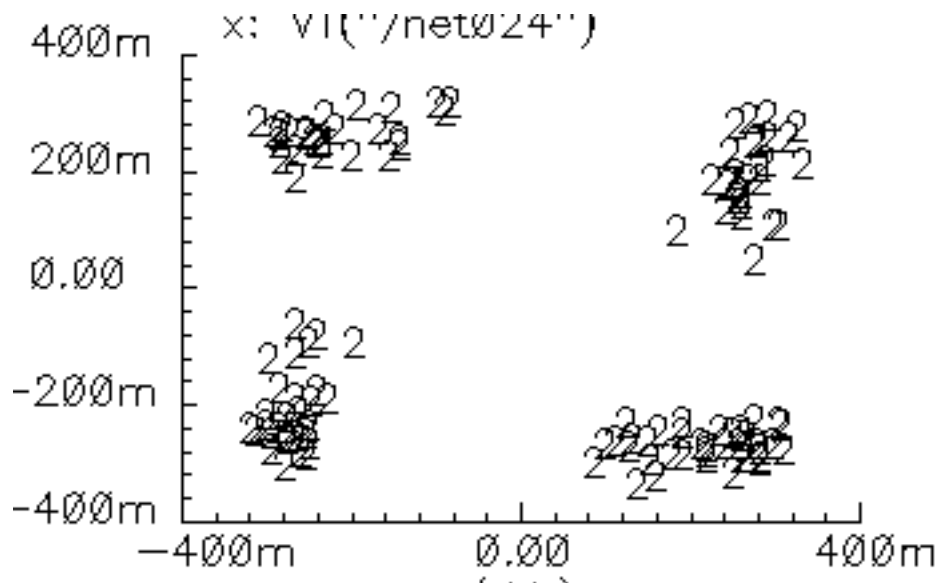
This creates a plot in the Waveform window.

In the Waveform window, select *Axes — X Axis*.

The X Axis form displays. In the X Axis form:

- d. In the *Plot vs. cyclic field*, select *1 VT( /net034 )* to plot the *I\_scatter* output on the x-axis.
- e. Click *OK*.
- f. In the Waveform window, select *Curves — Options*.
- g. The *Plot Style* form displays. In the *Plot Style* form:
- h. in the *Plotting Style* cyclic field, select *Data Points Only*.
- i. Click *OK*.
- j. You should see the scatter plot shown in [Figure 8-10](#) on page 626.

**Figure 8-10 Scatter Plot**



- 2. In the Waveform window, choose *Window — Close*.

## The Various Instrumentation Blocks

The CDMA source (*CDMA\_reverse\_xmit*) produced offset QPSK symbols. Offset QPSK modulation avoids traversing the origin by staggering the digital changes in the I and Q signals. Running the baseband trajectory through the origin increases spectral regrowth in the transmitters.

The instrumentation block (*offset\_comms\_instr*) samples the I and Q signals at different times then plots the two staggered samples against each other. The resulting scatter plot shows the received symbols. A scatter plot of the unstaggered samples reveals only what is happening in one dimension, either the I or Q dimension.

For non-offset QPSK and QAM modulation schemes, use the *comms\_instr* instrumentation block instead of the *offset\_comms\_instr* block.

## Measuring RMS EVM

You can use the same instrumentation block (*offset\_comms\_instr*) to compute root-mean-squared error vector magnitude (RMS EVM). The error vector is the vectorial difference between the ideal received symbol and the actual received symbol.

EVM (error vector magnitude) is the magnitude of the error vector.

RMS EVM is the root-mean-squared value of a sequence of EVMs.

RMS EVM is one measure of a receiver's quality. RMS EVM can account for as much or as little distortion and noise as you like. The trick is to figure out where the ideal received symbol lies. You can do this using the *I\_ref* and *Q\_ref* inputs to the *offset\_comms\_instr* instrumentation block.

To calculate RMS EVM, you

- Create a duplicate copy of the receiver chain from the *BB\_driver* to the *IQ\_demod\_BB* including these two blocks
- Place the duplicate copy below the original receiver chain in the Schematic window
- Modify the duplicate receiver chain to make it as *ideal* as you like by changing parameter values for the individual function blocks.
- For example, to see the effect of just the LNA's IP3 value on RMS EVM, in the duplicate receiver chain make the LNA's IP3 absurdly large.

## Constructing the Ideal Receiver Chain

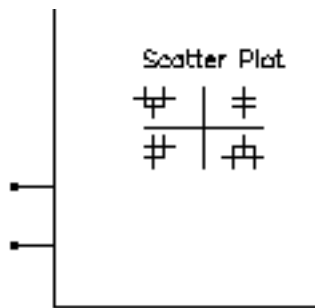
The *ideal* receiver chain (the duplicated and modified receiver chain) is driven from the same input, the *CDMA\_reverse\_xmit* block, as the original receiver chain. The output of the ideal receiver chain drives the instrumentation block's *I\_ref* and *Q\_ref* inputs.

In the ideal receiver chain, you copy the first receiver chain and make every block ideal.

Remove the *gnd* from the *I\_ref* and *Q\_ref* pins on the instrumentation block.

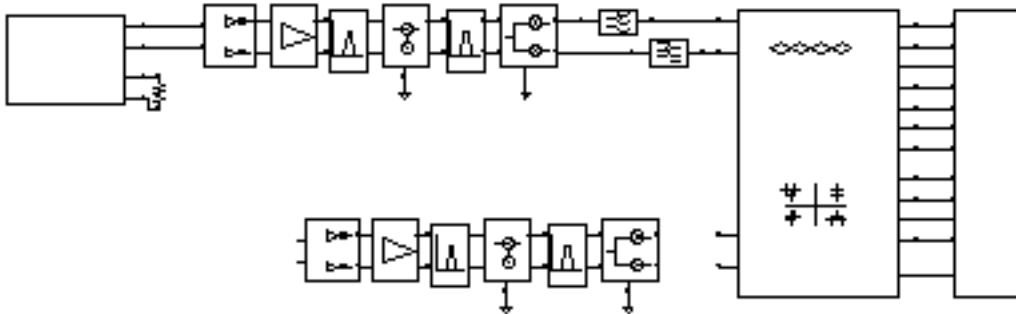
In the Schematic window, choose *Edit - Delete*.

In the Schematic window, click each wire and the *Gnd* symbol attached to the *I\_ref* and *Q\_ref* pins.



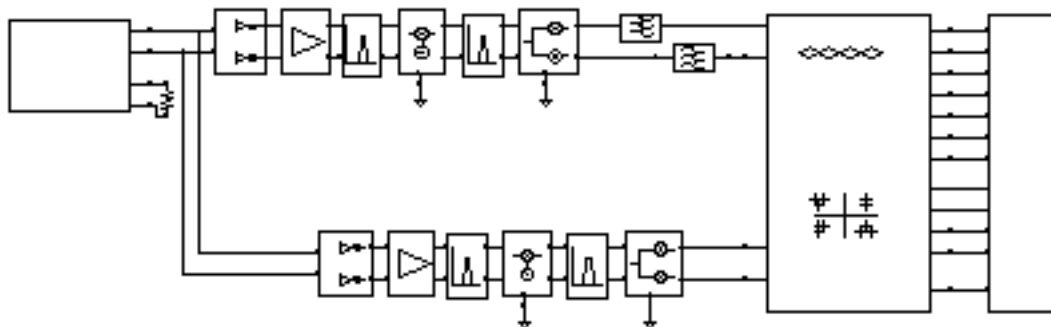
- a. Click *esc* to stop deleting.
- b. Duplicate the receiver chain from the *BB\_driver* to the *IQ\_demod\_BB* inclusive. Do not duplicate the filters. Follow the prompts at the bottom of the Schematic window.
- c. In the Schematic window, choose *Edit—Copy*.
- d. In the Schematic window, draw a box around the blocks to copy by clicking to the left of and above the *BB\_driver* block and dragging the cursor to a point below and to the right of the *IQ\_demod\_BB* block. Click again to complete the box.
- e. The blocks within the box are highlighted.
- f. Click within the highlighted area.
- g. A copy of the highlighted blocks in the receiver chain now moves with the cursor.

- h.** Place the duplicate receiver chain so that the output pins on the *IQ\_demod\_BB* are in line with the *I\_ref* and *Q\_ref* pins on the instrumentation block.



1. Wire the duplicate receiver chain to the CDMA signal source (*CDMA\_reverse\_xmit*) and the instrumentation block (*offset\_comms\_instr*).
2. In the Schematic window, choose *Add - Wire (narrow)*.
  - a. Connect the *IQ\_demod\_BB* outputs on the duplicate receiver to the *I\_ref* and *Q\_ref* pins on the instrumentation block.
  - b. Connect the output pins on *CDMA\_reverse\_xmit* to the input pins of the duplicate *BB\_driver*. This drives the duplicate receiver from the CDMA signal source.
  - c. Click *esc* to stop wiring.
  - d. The schematic with the duplicate receiver chain wired up looks like Figure 8-11.

**Figure 8-11 Receiver Model with Duplicated Receiver Chain**



### Modifying Parameter Values to Make the Blocks Ideal

Now modify the parameter values for each block in the duplicate receiver chain to create ideal blocks. Block names, parameter names, and parameter values are given in Table 8-10.

**Table 8-10 Parameter Values to Create an Ideal Receiver**

Block Names	Parameter Names	New Parameter Values
LNA_BB	<i>Input referred IP3 [dBm]</i> <i>{1,0,-1} for {cw, none, ccw}</i>	100 0
BB_butterworth_bp	<i>Cell Name</i>	BB_loss
dwn_cnvrt	<i>Input referred IP3 [dBm]</i>	100
BB_butterworth_bp	<i>Cell Name</i>	BB_loss
IQ_demod_BB	<i>I-[dBm] Input referred IP3</i> <i>Q-[dBm] Input referred IP3</i>	100 100

1. In the Schematic window, choose *Edit—Properties—Objects* to open the Edit Object Properties form.
2. In the Schematic window, select the *LNA\_BB* block.
3. The Edit Object Properties form changes to display properties for the *LNA\_BB* block.
4. Set *Input referred IP3 [dBm]* to 100.
  - a. Set *{1, 0,-1} for {cw, none, ccw}* to 0. (This eliminates AM/PM conversion.)
  - b. Click *Apply*.
5. In the Schematic window, select the first RF *BB\_butterworth\_bp* block.
6. The Edit Object Properties form changes to display properties for the *BB\_butterworth\_bp* block.
7. Change the *Cell Name* to *BB\_loss*.
8. Click *Apply*.
9. The properties and symbol change to those for the *BB\_loss* block. The sole purpose of the *BB\_loss* model is to replace a filter in an RMS EVM analysis.

10. The *Reference impedance* for the *BB\_loss* block should equal the *Output impedance* of the *BB\_butterworth\_bp* bandpass filter block it replaces. The value should be 50 ohms for both blocks and you should not have to change it.
11. The *BB\_loss* model retains the filter's loss but eliminates the filter's dynamics so you can see what, if any, affect the filter has on EVM through inter-symbol interference. To eliminate the loss as well as the dynamics, you might even replace the filter with straight wires. This example uses the *BB\_loss* block instead.
12. In the Schematic window, select the *dwn\_cnvrt* block.
13. The Edit Object Properties form changes to display properties for the *dwn\_cnvrt* block
  - a. Set *Input referred IP3 [dBm]* to 100.
  - b. Click *Apply*.
14. In the Schematic window, select the second *BB\_butterworth\_bp* block.
15. The Edit Object Properties form changes to display properties for the *BB\_butterworth\_bp* block
  - a. Change the *Cell Name* to *BB\_loss*.
  - b. Click *Apply*.
  - c. In the Schematic window, select the *IQ\_demod\_BB* block.
  - d. The Edit Object Properties form changes to display properties for the *IQ\_demod\_BB* block
    - a. Set *I-[dBm] input referred IP3* to 100.
    - b. Set *Q-[dBm] input referred IP3* to 100.
    - c. Click *Apply*.
16. In the Edit Object Properties form, click *OK* to close the form.
17. In the Schematic window, select *Design—Check and Save* to check and save your modifications to the circuit.

## Set Up and Run a Transient Analysis

Set up and run a transient analysis as described in “[Setting Up and Running a Transient Analysis](#)” on page 619. Set the *Stop Time* to 130u and the *outputstart* option to 30u. Click *OK* in both the Transient Options and Choosing Analyses forms. Choose *Simulation—Netlist and Run* to run the transient analysis.

Look for messages in the CIW stating that the simulation is starting. Watch the simulation log window for messages that the simulation has completed successfully.

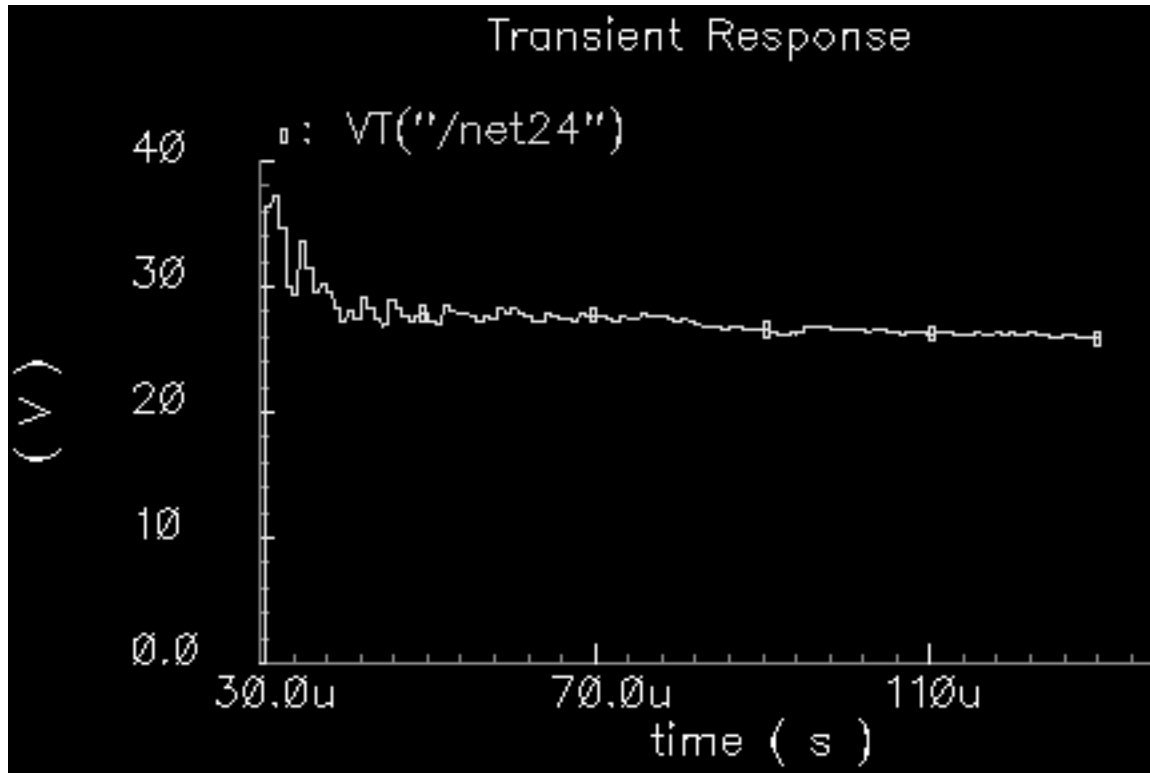
### **Plot the RMS EVM Output**

After the simulation, plot the RMS\_EVM output of the instrumentation block.

1. In the Simulation window, choose *Results—Direct Plot—Transient Signal*.
2. This displays the Waveform window.
3. Following the prompts at the bottom of the Waveform window.
4. In the Schematic window,
  - a. Click the *rms\_EVM* output net from the instrumentation (*offset\_comms\_instr*) block.
  - b. Click *Esc* to indicate that you have finished selecting outputs.
  - c. This creates the RMS EVM plot in the Waveform window as shown in [Figure 8-12](#) on page 633.



Figure 8-12 RMS EVM



The RMS EVM trace starts at 30us, which is the `statistics start time` parameter of the instrumentation block. The `statistics start time` parameter keeps start-up transients out of the statistics.

The trace settles out at 25.84 Volts. This means that after 130us of data is collected, and ignoring the first 30us, the RMS EVM is 25.84%. The EVM measurement is normalized to the RMS magnitude of the ideal symbol then multiplied by 100 to express the measurement as a percentage.

5. In the Waveform window, choose *Window — Close*.
6. Computing Minimized RMS Noise Using the Optimizer

There is one more construction step before proceeding to the Circuit Optimizer application. You set up the Circuit Optimizer to minimize RMS noise subject to performance constraints. This step replicates the receiver chain yet one more time to generate the noise measurement.

Duplicate the original receiver chain from the *BB\_driver* up to and including both low pass filters (*butterworth\_lp*). Follow the prompts at the bottom of the Schematic window.

In the Schematic window, choose *Edit—Copy*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

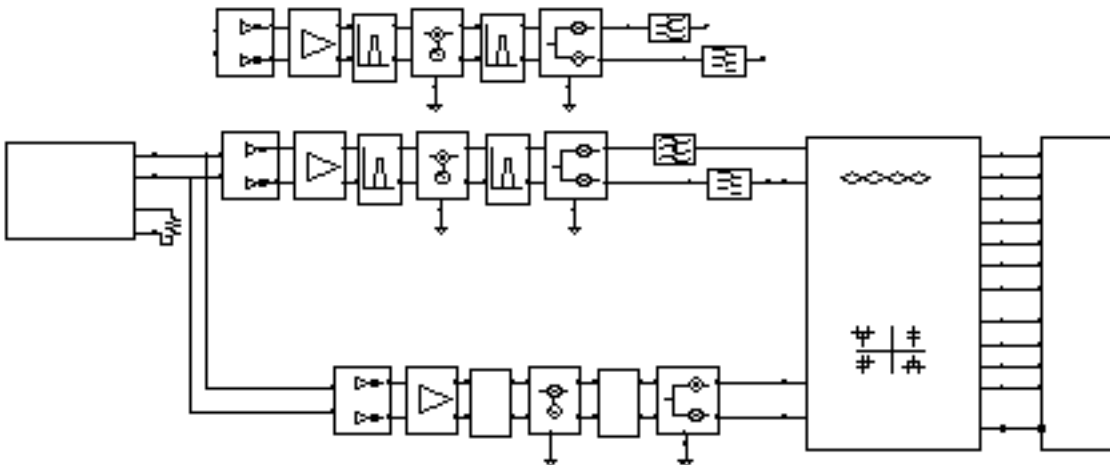
In the Schematic window, draw a box around the blocks to copy by clicking to the left of and above the *BB\_driver* block and dragging the cursor to a point below and to the right of the *butterworth\_lp* filter blocks. Click again to complete the box.

The blocks within the box are highlighted.

Click within the highlighted area.

A copy of the highlighted blocks in the receiver chain now moves with the cursor.

Place the duplicate receiver chain above the original receiver chain.



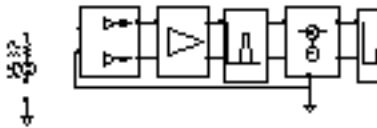
1. In the duplicate receiver chain, ground the *Q\_in* pin on the *BB\_driver* block.
2. In the Schematic window, choose *Add—Wire*.
3. Click the *Q\_in* pin and run the wire to the *Gnd* symbol below the *dwn\_cnvrt* block.
  - a. Click the *Gnd* symbol below the *dwn\_cnvrt* block.
  - b. Click *esc* to stop wiring.
  - c. Add a 50mV DC voltage source to the left of the *BB\_driver* block to drive the *I\_in* pin on the *BB\_driver* block. At the same time add a *gnd* symbol below the *port* in the schematic.
  - d. In the Schematic window, choose *Add—Instance* to display the Add Instance form.
  - e. In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

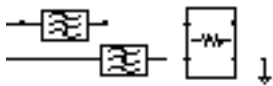
---

- f. In the Library Browser - Add Instance form, click *analogLib*.
- g. Scroll the elements in the *Cell* column and click *port*.
- h. The outline for the *port* symbol is attached to the cursor. Move the *port* symbol to the left of the *BB\_driver* block and click to place the *port* symbol.
- i. Return to the Library Browser - Add Instance form and scroll the elements in the *Cell* column and click *gnd*.
- j. The outline for the *gnd* symbol is attached to the cursor. Move the *gnd* symbol below the *port* symbol and click to place it there.
- k. Click *Esc* to remove the *gnd* symbol from the cursor.

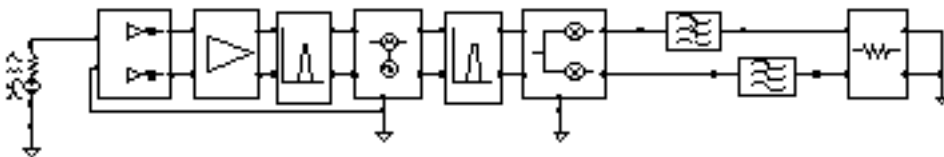


4. In the Schematic window, choose *Edit—Object—Properties* to modify the *port* symbol using the Edit Object Properties form.
5. In the Schematic window, click the *port* symbol.
6. The Edit Object Properties form changes to display information for the *port* symbol.
7. In the *Source Type* cyclic field, select *dc*.
8. In the *DC Voltage* field, enter 50m.
9. Highlight *Display small signal params* to display small signal parameters.
  - a. In the *AC Magnitude* field type 1V.
  - b. In the *AC Phase* field type 0.
  - c. Click *OK* in the Edit Object Properties form.
  - d. Load the low pass filters with a *res\_BB* model from the *top\_dwnBB* category of *rfLib*. Use the default parameters and ground the output pins.
  - e. In the Schematic window, choose *Add—Instance* to display the Add Instance form.
  - f. In the Add Instance form, click *Browse* to display the Library Browser - Add Instance form.
  - g. In the Library Browser - Add Instance form, click *rfLib*.

- h.** Scroll the elements in the *Cell* column and click *res\_BB*.
- i.** The outline for the *res\_BB* symbol is attached to the cursor. Move the *res\_BB* symbol to the right of the two low pass filters (*butterworth\_lp*) and click to place the *res\_BB* symbol.
- j.** Return to the Library Browser - Add Instance form and click *analogLib*.
- k.** Scroll the elements in the *Cell* column and click *gnd*.
- l.** The outline for the *gnd* symbol is attached to the cursor. Move the *gnd* symbol to the right of the *res\_BB* symbol and click to place it there.
- m.** Click *Esc* to remove the symbol from the cursor.



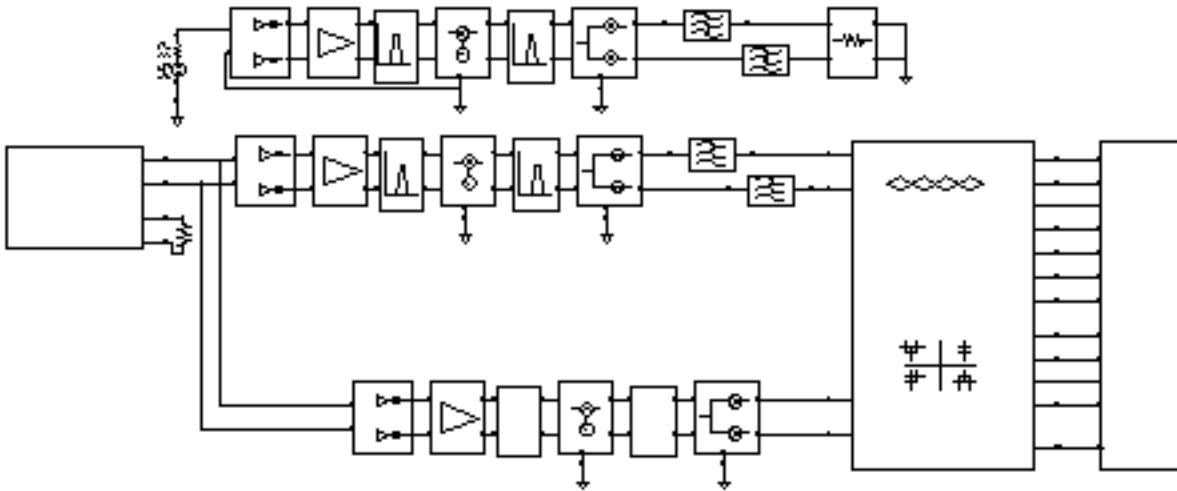
- 10.** In the duplicate receiver chain, wire the *port*, the *res\_BB* block, and their *gnd* blocks.
- 11.** In the Schematic window, choose *Add—Wire*.
- 12.** Click the *I\_in* pin on the *BB\_driver*, then click the top pin on the *port*.
- 13.** Click the bottom pin on the *port*, then click the *gnd* pin just below it.
- 14.** Click the *out* pin on the top *butterworth\_lp* filter, then click the *I\_in* pin on the *res\_BB* block.
- 15.** Click the *out* pin on the lower *butterworth\_lp* filter, then click the *Q\_in* pin on the *res\_BB* block.
- 16.** Click the *I\_out* pin on the *res\_BB* block. Then click the top pin on the *gnd* located to its right.
- 17.** click the *Q\_out* pin on the *res\_BB* block. Then click the top pin on the same *gnd*.



- a. Click `esc` to stop wiring.
- b. In the Schematic window, choose *Design—Check and Save* to check and save the schematic.

The schematic with the third receiver chain is shown in Figure 8-13.

**Figure 8-13 Schematic with Noise Generating Receiver**



### Set Up and Run Transient and Noise Analyses

Set up a transient analysis as described in “[Setting Up and Running a Transient Analysis](#)” on page 619. Set the *Stop Time* to `130u` and the *outputstart* option to `30u` and make sure that the transient analysis is enabled.

Set up a noise analysis as follows:

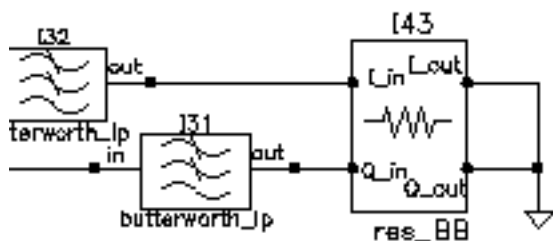
1. In the Choosing Analyses form, click *noise* to select a noise analysis.
2. For *Sweep Variable*, click *Frequency*.
3. For *Sweep Range*, click *Start-Stop*.
4. Set up the analysis to sweep frequency from 0 to 100 MHz.
5. For the starting frequency, in the *Start* field enter 0.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

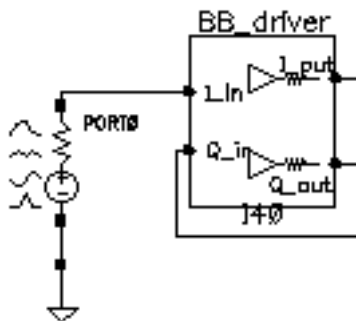
---

- a. For the stop frequency, in the *Stop* field enter 100M.
- b. Set up the *Output Noise* source.
- c. In the *Output Noise* cyclic field, select *voltage*.
- d. To select the *Positive Output Node*, click *Select* next to the *Positive Output Node* field. Then, in the Schematic window, click the net next to the *I\_in* pin on the *res\_BB* block.



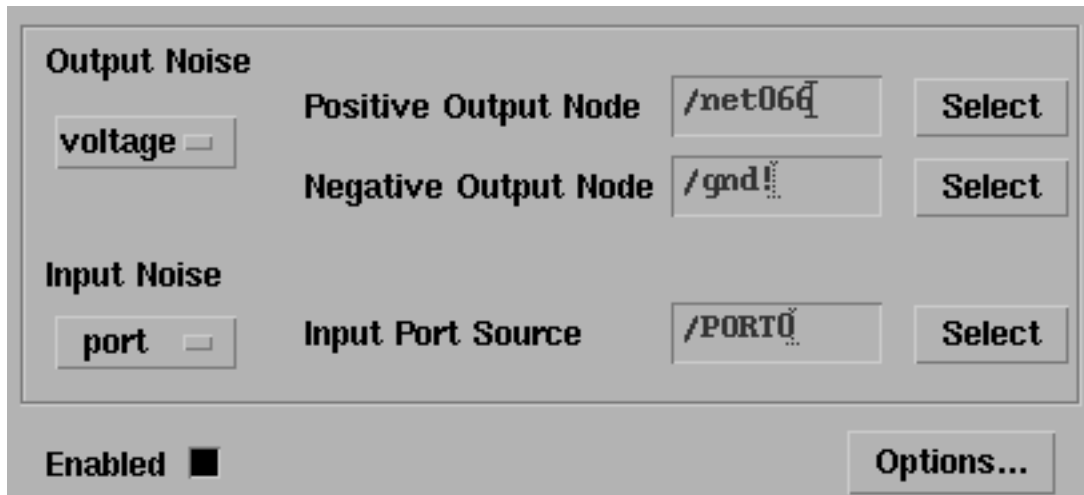
- e. To select the *Negative Output Node*, click *Select* next to the *Negative Output Node* field. Then, in the Schematic window, click the net next to the *I\_out* pin on the *res\_BB* block.

6. Set up the *Input Noise* source.
7. In the *Input Noise* cyclic field, select *port*.
8. To select the *Input Port Source*, click *Select* next to the *Input Port Source* field. Then, in the Schematic window, click the *DC input port* model.



9. Click Enabled.

10. The *Output Noise* and *Input Noise* sections of the Noise analysis form look as follows.



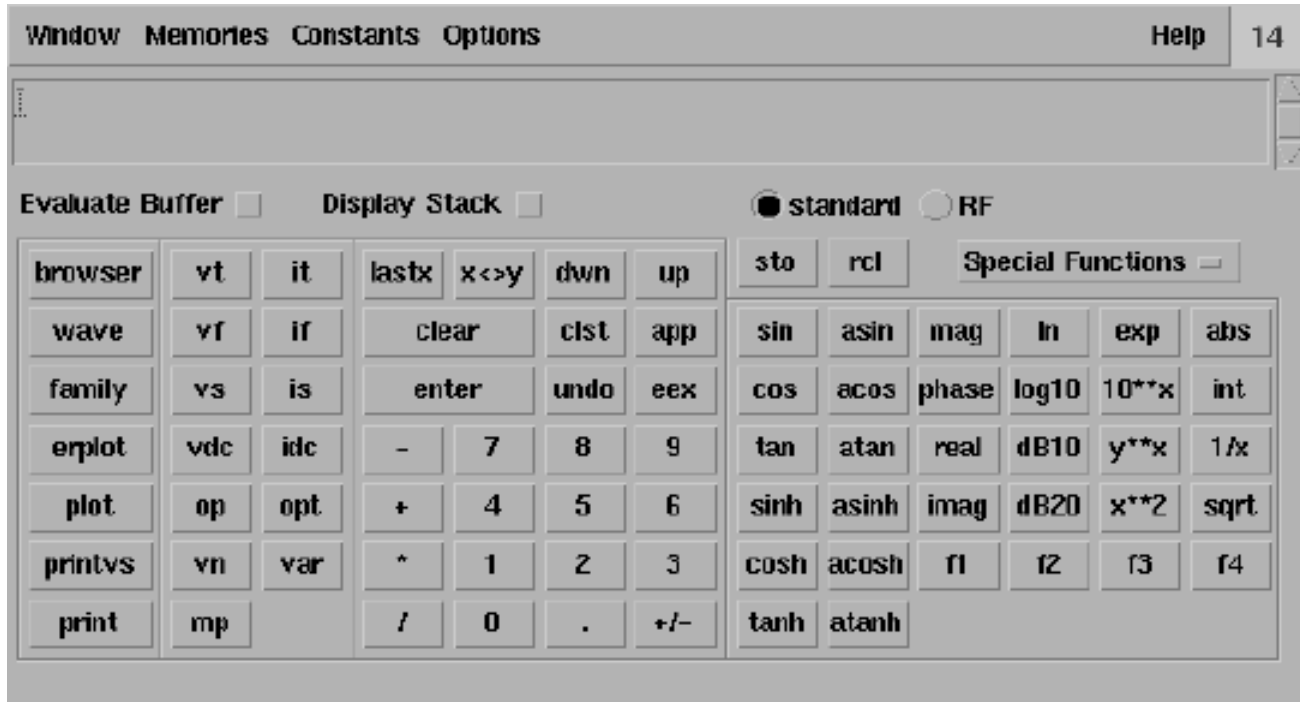
11. Verify that the noise analysis is *Enabled* and click *OK* in the Choosing Analyses form.
12. In the Simulation window, check the Analysis area to verify that both the transient and noise analyses are set up properly and that they are both enabled.

Analyses						
#	Type	Arguments.....				Enable
1	noise	0	100M	Auto..	Star..	yes
2	tran	30u	130u			yes

13. In the Simulation window, choose *Simulation—Netlist and Run* to start the simulations.
14. Watch the messages in the CIW to verify that everything is set up properly and that the simulations start. Check the simulation log window to see that the simulations run and complete properly.

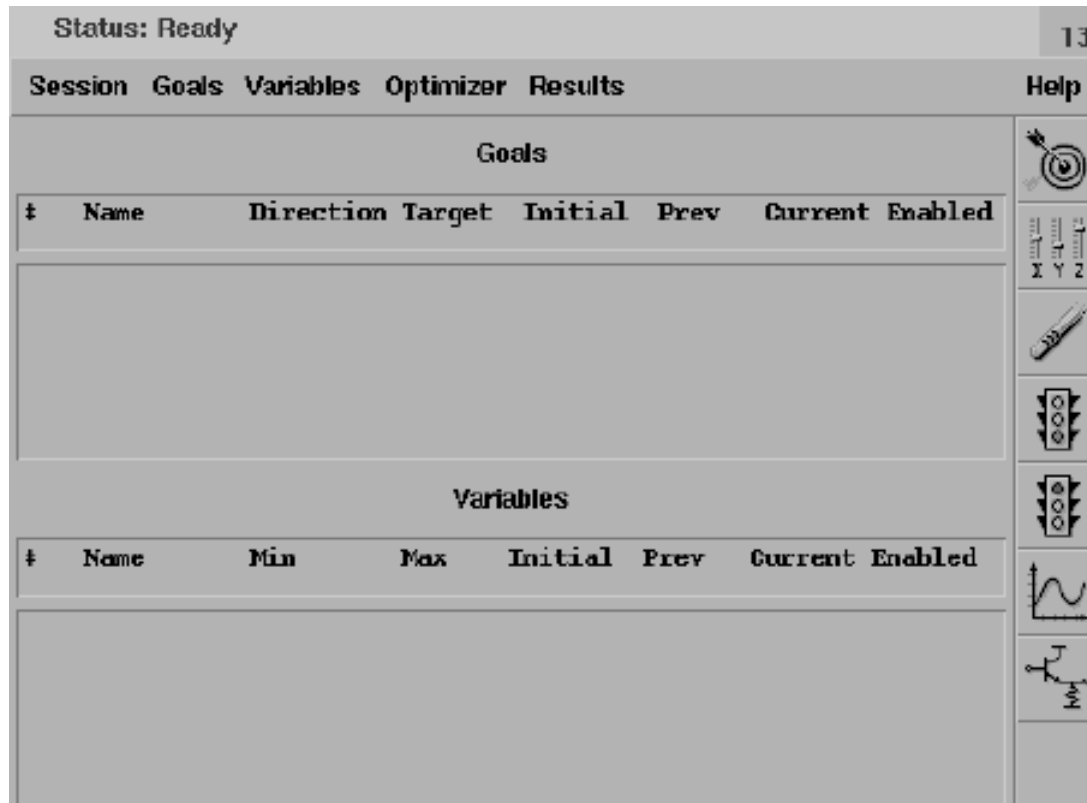
### Set Up to Run the Circuit Optimizer

1. In the Simulation window, choose *Tools—Calculator* to open the Waveform Calculator window.





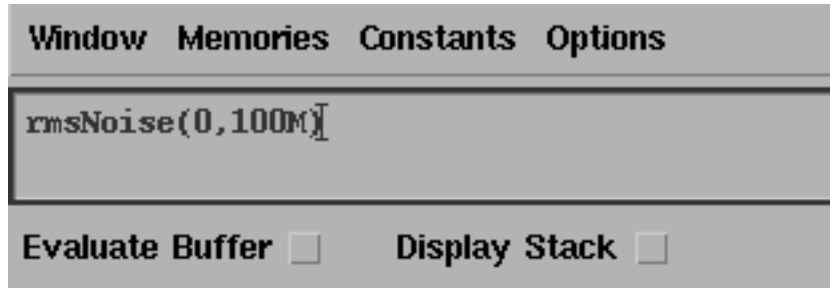
2. In the Simulation window, choose *Tools—Optimization* to open the Circuit Optimizer window.



### ***Add the First Goal***

1. In the Calculator window's *Special Function* cyclic field, select *rmsNoise*.
2. When the RMS Noise form opens.
3. In the *From* field, enter 0.
4. In the *To* field, enter 100M.
5. Click *OK*.

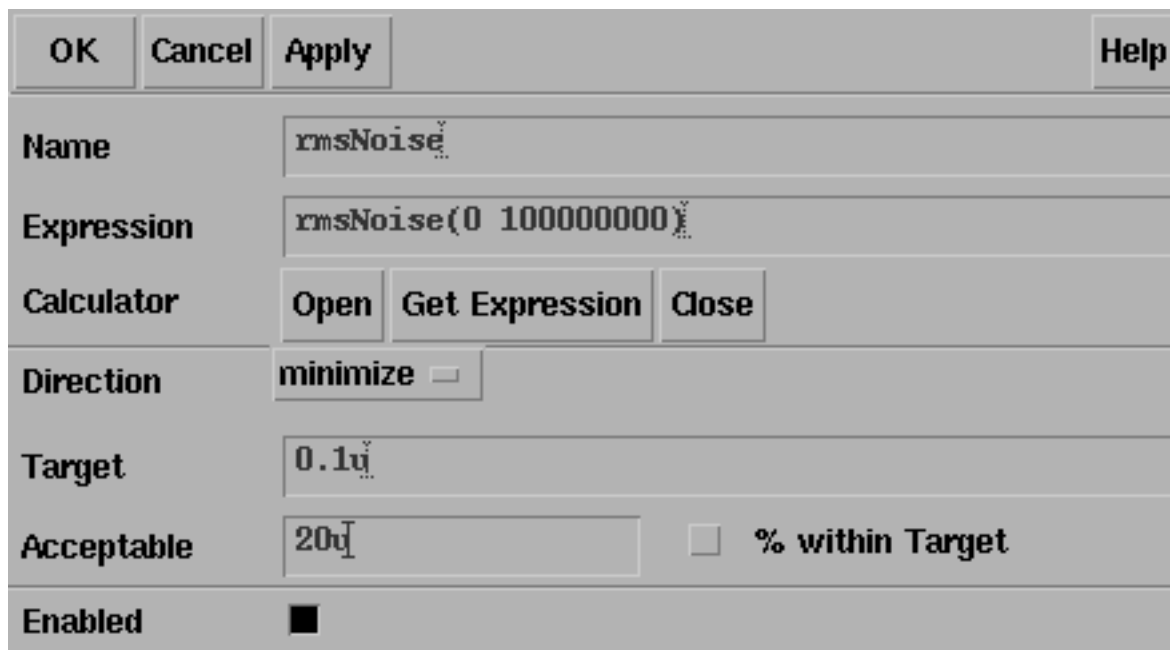
6. The expression `rmsNoise(0,100M)` displays in the Calculator window.



7. In the Circuit Optimizer, choose *Goals—Add* to add the first goal.

8. When the Adding Goals form opens

- a. In the *Name* field, enter `rmsNoise`.
- b. Click the *Get Expression* button to the right of the *Calculator* label.
- c. The expression `rmsNoise(0,100000000)` displays in the *Expression* field.
- d. In the *Target* field, enter `0.1u`.
- e. In the *Acceptable* field, enter `20u`.
- f. The Adding Goals form looks as follows.



- g. Verify that the form is enabled and click *OK*.
- h. The first goal is added to the Circuit Optimizer's Goals window.
- i. By default the Circuit Optimizer minimizes goals.

Goals							
#	Name	Direction	Target	Initial	Prev	Current	Enabled
1	rmsNoise	minimize	100n				yes

**Add the Second Goal**

1. In the Calculator window, click the *vt* button.
2. Then, in the Schematic window, click the *rms\_EVM* output net of the instrumentation block.
3. The expression `VT("/net0101")` displays in the Calculator window.
4. In the Calculator's *Special Functions* cyclic field, select *value*.
5. When the Value form appears
  - a. Enter `130u` in the *Interpolate At* field.
  - b. Click *OK*.
  - c. The expanded expression `value(VT("/net0101"),130u)` displays in the Calculator window.
6. In the Circuit Optimizer, choose *Goals—Add* to add the second goal.
7. When the Adding Goals form opens
  - a. In the *Name* field, enter `evm`.
  - b. Click the *Get Expression* button to the right of the *Calculator* label.
  - c. The expression `value(VT("/net0101") 0.00013)` displays in the Expression field.
  - d. in the *Direction* cyclic field, select `<=`
  - e. In the *Target* field, enter `25`.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

- f. In the *Acceptable* field, enter 10.
- g. Click the *% within Target* button.
- h. The Adding Goals form looks as follows.

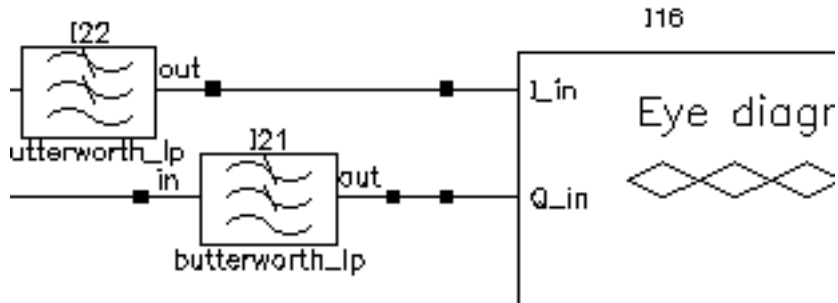
OK	Cancel	Apply		Help
<b>Name</b>	evm			
<b>Expression</b>	value(VT("/net24") 0.00013)			
<b>Calculator</b>	Open	Get Expression	Close	
<b>Direction</b>	<= <input type="checkbox"/>			
<b>Target</b>	25			
<b>Acceptable</b>	10	<input checked="" type="checkbox"/> % within Target		
<b>Enabled</b>	<input checked="" type="checkbox"/>			

- i. Verify that *Enabled* is active and click *OK*.
- j. The second goal is added to the Circuit Optimizer's *Goals* window.

Goals							
#	Name	Direction	Target	Initial	Prev	Current	Enabled
1	rmsNoise	minimize	100n				yes
2	evm	<=	25				yes

**Add the Third Goal**

1. In the Calculator window, click the *vt* button. Then, in the Schematic window, click the *I\_in* pin of the instrumentation block.

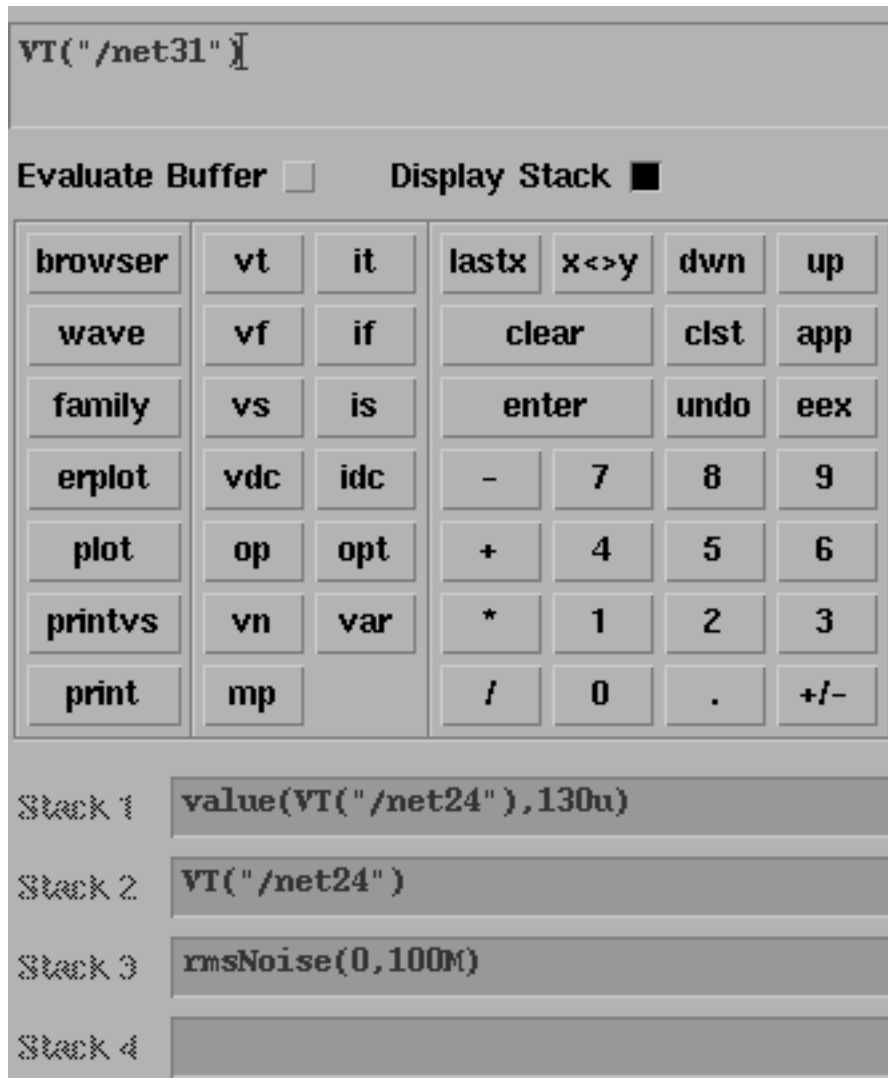


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

The expression `VT( "/net043" )` displays in the Calculator window. Notice that as new expressions are added to the calculator, the existing expressions move down the calculator's stack.



2. In the Calculator's *Special Functions* cyclic field, select *rms*.
3. The expanded expression `rms( VT( "/net043" ) )` displays in the Calculator window.

The objective is to keep the *rms* value of this signal level above 300 mV. Note that all goals must be scalars.

4. In the Circuit Optimizer, choose *Goals - Add* to add the third goal.
5. When the Adding Goals form opens

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

- a. In the *Name* field, enter `sig_level`.
- b. Click the *Get Expression* button to the right of the *Calculator* label.
- c. The expression `rms(VT("/net043"))` displays in the *Expression* field.
- d. in the *Direction* cyclic field, select `>=`
- e. In the *Target* field, enter `300m`.
- f. In the *Acceptable* field, enter `10`.
- g. Click the *% within Target* button.
- h. The Adding Goals form looks as follows.

OK	Cancel	Apply	Help
Name	sig_level		
Expression	rms(VT("/net31"))		
Calculator	Open	Get Expression	Close
Direction	>= <input type="checkbox"/>		
Target	300m		
Acceptable	10	<input checked="" type="checkbox"/> % within Target	
Enabled	<input checked="" type="checkbox"/>		

- i. Click *OK*.

j. The third goal is added to the Circuit Optimizer's *Goals* window.

Goals							
#	Name	Direction	Target	Initial	Prev	Current	Enabled
1	rmsNoise	minimize	100n				yes
2	evm	<=	25				yes
3	sig_level	>=	300m				yes

### ***Add the Circuit Variables to the Optimizer***

Add the variables to the Circuit Optimizer window.

1. In the Circuit Optimizer window, choose *Variables—Add/Edit*.
2. When the Editing Variables form opens
  - a. In the *Name* list box, click the *Ina\_ip3* variable.

The *Ina\_ip3* variable is highlighted in the list box and its current value  $-5$  displays in the *Initial Value* field.

In the *Minimum Value* field, enter  $-9$ .

In the *Maximum Value* field, enter  $10$ .

- b. If necessary, click *Enabled*.



c. The Editing Variables form appears as follows.

d. Click *Apply*.

e. Information for the *lna\_ip3* variable displays in the *Variables* section of the Circuit Optimizer.

Variables						
#	Name	Min	Max	Initial	Prev	Current Enabled
1	lna_ip3	-9	10	-5		yes

3. Repeat this procedure to add all the variables and values listed in Table [8-11](#)

**Table 8-11 rVariables and Values for the Optimize**

Variable Name	Initial Value	Minimum Value	Maximum Value
lna_ip3	-5	-9	10

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide












### Methods for Top-Down RF System Design

Variable Name	Initial Value	Minimum Value	Maximum Value
lna_gain	15	10	30
if_mx_gain	10	1	50
if_rbw	200m	50m	300m
rf_rbw	100m	50m	300m

4. Click *OK* to close the Editing Variables form.
5. In the Circuit Optimizer window, the variables and optimization goals appear as shown in [Figure 8-14](#) on page 650.

**Figure 8-14 Circuit Optimizer Setup**

Status: Ready 15

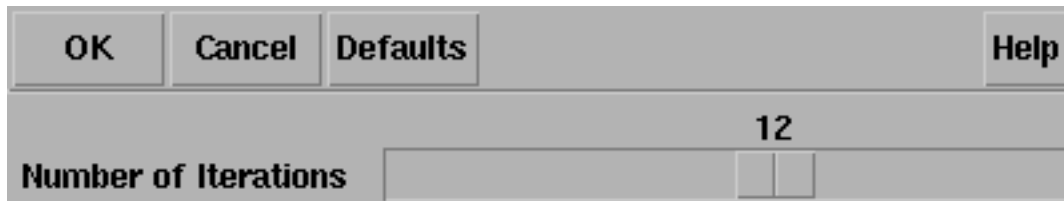
Session	Goals	Variables	Optimizer	Results	Help			
<b>Goals</b>								
#	Name	Direction	Target	Initial	Prev	Current	Enabled	
1	rmsNoise	minimize	100n				yes	
2	evm	<=	25				yes	
3	sig_level	>=	300m				yes	
<b>Variables</b>								
#	Name	Min	Max	Initial	Prev	Current	Enabled	
1	lna_ip3	-9	10	-5			yes	
2	lna_gain	10	30	15			yes	
3	if_nix_g..	1	50	10			yes	
4	rf_rbw	50m	300m	100m			yes	
5	if_rbw	50m	300m	200m			yes	

## Run the Circuit Optimizer

1. In the Circuit Optimizer, choose *Optimizer—run n* to display the *Run for Fixed Number of Iterations* form.

In the *Run for Fixed Number of Iterations* form

- a. In the *Number of Iterations* field, move the slider to the right until 12 displays.



- b. Click *OK*.

The *Run for Fixed Number of Iterations* form closes and the Circuit Optimizer starts running.

Watch for simulator startup messages in the CIW. Monitor the progress of the analysis in the log window.

The Waveform window opens when the first simulation completes. As simulations complete, the results are added to the plots in the Waveform window

**Note:** This Circuit Optimizer analysis might take up to several hours to complete.

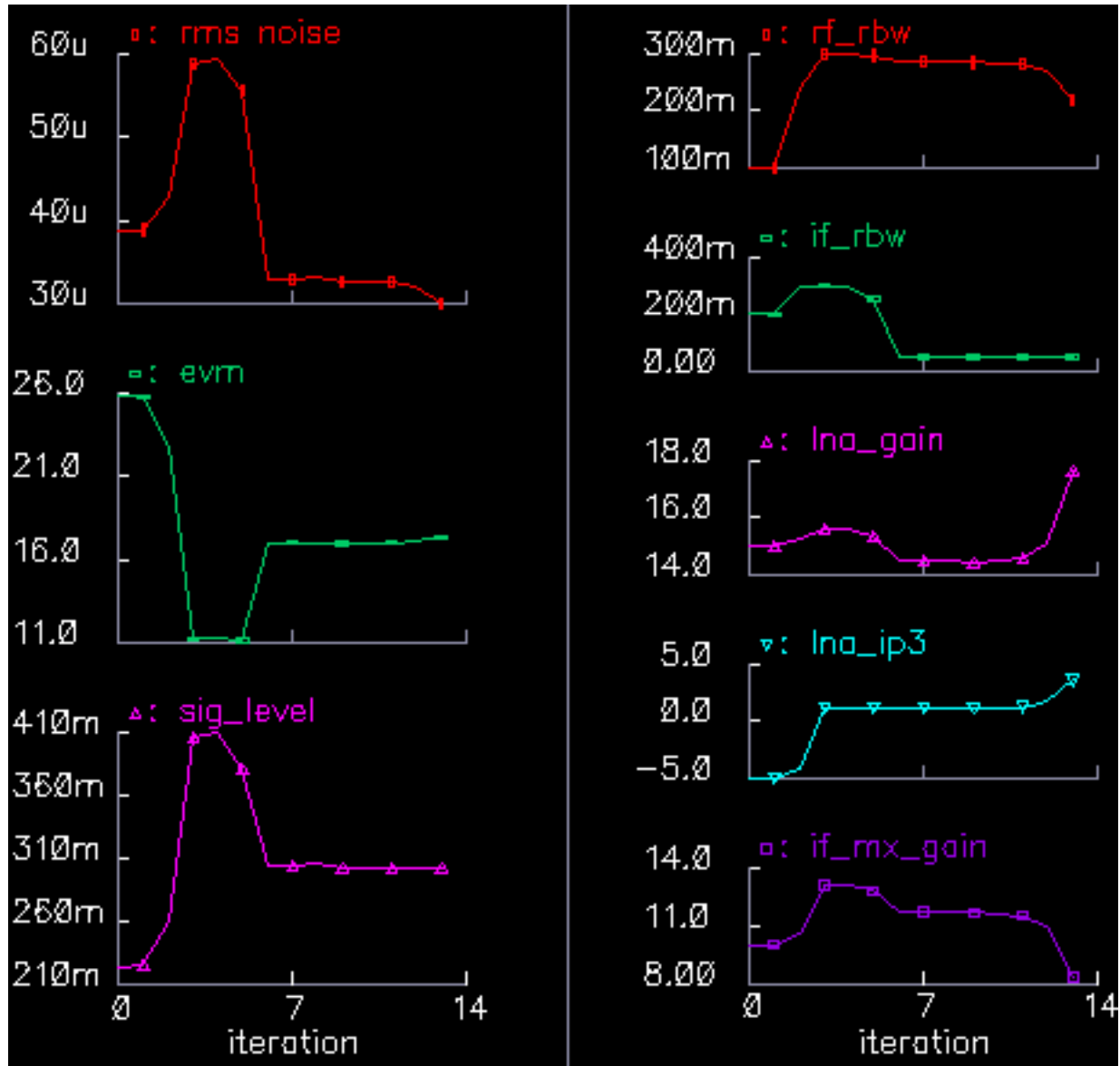
## Viewing the Circuit Optimizer Output

The Circuit Optimizer results displayed in the Waveform window are shown in [Figure 8-15](#) on page 652.

The traces on the left show the EVM and RMS output signal level, and the RMS noise at each Circuit Optimizer iteration.

The traces on the right show the variable function block specifications at each Circuit Optimizer iteration.

Figure 8-15 Circuit Optimizer Results After 12 Iterations



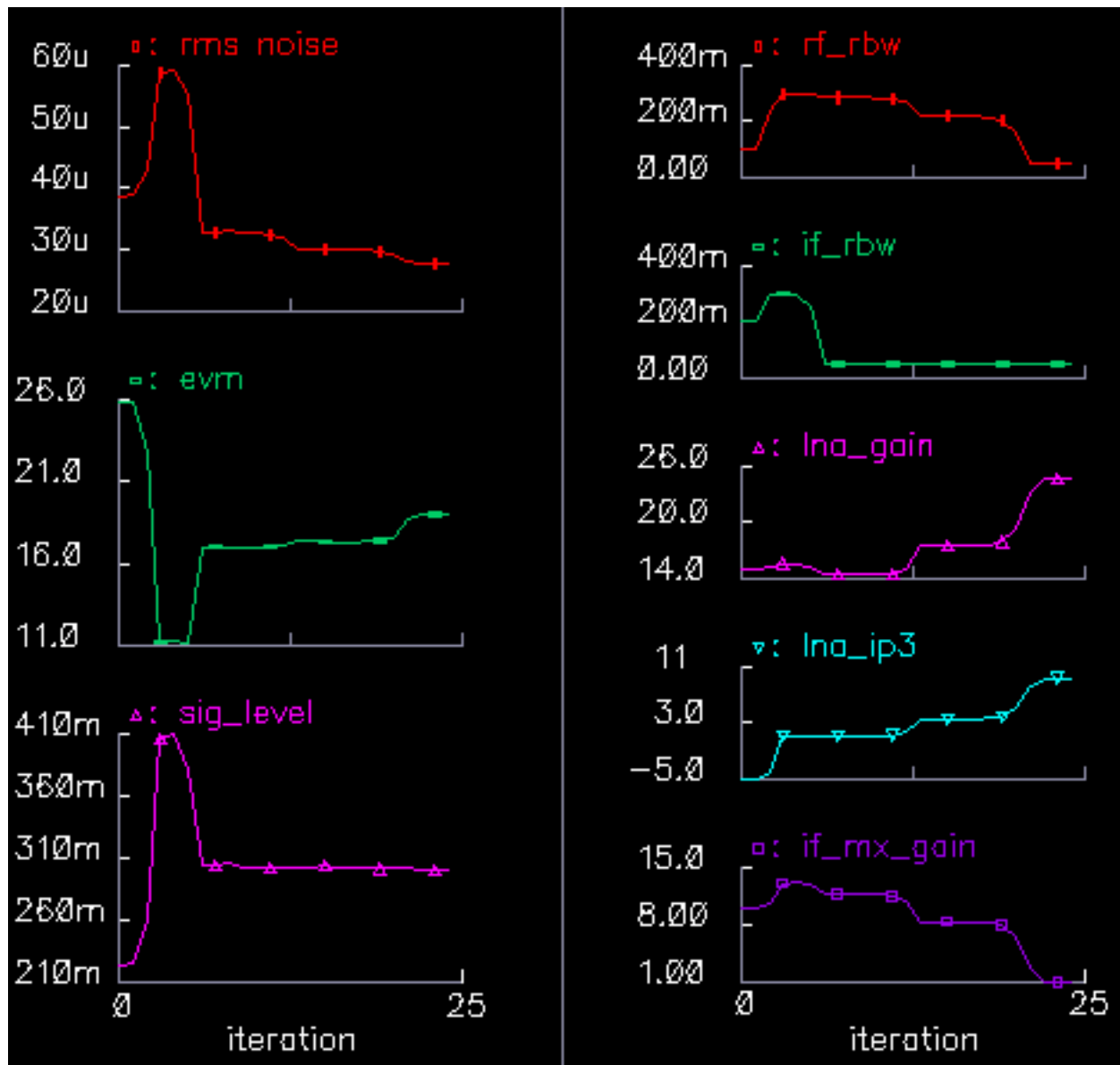
Although the example is contrived, it illustrates the use model. After the Circuit Optimizer met the constraints it tried to minimize RMS noise.

Save the initial state of the Analog Design Environment in case you want to start over.

Then in the Circuit Optimizer window, click *Results—Update Design*. The last click updates the variables in the Analog Design Environment window with the last set of variables found by the Circuit Optimizer. You use these states in the passband view.

1. If you have time to run the optimization for another 12 iterations you should see the results in [Figure 8-16](#) on page 653.

Figure 8-16 Circuit Optimizer results after 24 iterations



## Summarizing the Design Procedure

To summarize, the semi-automated design procedure consists of

- Setting up the measurements
- Placing tolerances on the block parameters
- Constraining the system performance
- Identifying a quantity to minimize (or maximize)
- Running the Optimizer
- Evaluating the results

This is why the process is called *semi*-automated. After evaluating results for the first or second time you probably have to

- Refine tolerances
- Refine goals
- Add or delete constraints
- Add or delete variables

Each simulation covers 100s, or about 80 CDMA symbols. The suppressed carriers are an RF carrier at 2.14 GHz, an LO carrier at 2.354GHz, and an IF carrier at 214 MHz. The symbol rate is 1.2288 Mega-symbols per second.

## Creating a Passband View of the Architectural Model

Once you have designed an architecture, you can quickly create a passband view of the architectural model. (Currently, the passband behavioral models in the *top\_dwnPB* category and in the passband view do not introduce any specifications that are not in the baseband models. In subsequent software releases of the library, the passband models may have the ability to introduce new specifications such as IIP2 and DC offsets.)

The passband view checks for problems that might have escaped detection in the baseband view. For example, although the baseband view quickly assesses what filters do to the baseband signal, baseband models do not indicate whether the filters are indeed removing undesired carrier harmonics.

Baseband modeling is also not the best way to evaluate image rejection. Although the baseband model accurately simulates how the desired signal propagates through an image

rejection receiver, it does not accurately simulate how much of the image signal propagates to the receiver output.

The passband view also creates a system testbench as mentioned in [“Top-Down Design of RF Systems”](#) on page 554.

### **Procedures for Creating the Passband Model of the Receiver**

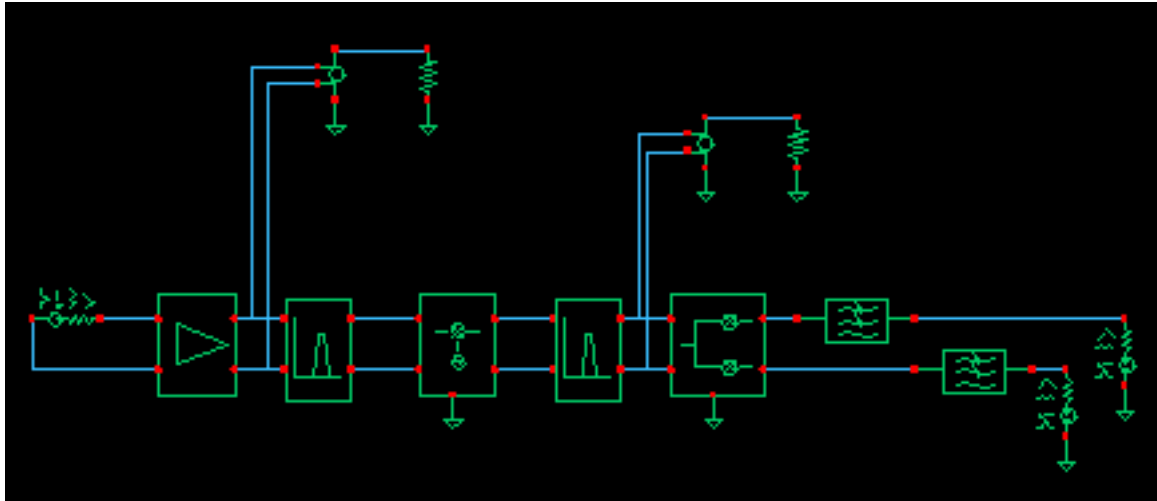
The procedures described in this section illustrate how to

- Switch from a baseband to a passband view
- Make an end-to-end RF measurement
- Measure the one dB compression point

The one dB compression point is usually a transmitter specification but it is used to demonstrate this flow because it is easier to set up.

1. Copy the original receiver model from the Circuit Optimizer analysis to a new schematic window. Copy everything from the LNA to the low pass filters.
2. Edit the properties of the IQ\_demodulator (*IQ\_demod\_BB*) to set the last parameter, *flo*, to  $-frf+flo1$ . The baseband view does not need the local oscillator frequency but the passband view does.
3. Load the low pass filters with ports.
4. Connect a port across the LNA\_BB inputs. Set the Frequency name to “fin”, the frequency to frf, and the amplitude to “power”. (Do not abbreviate power to “pwr”. “pwr” is a reserved variable and you do not get any warning. Spectre RF may complain about a mysterious indexed undefined variable that increments from run to run.)
5. Add loaded voltage-controlled-voltage sources as shown in [Figure 8-17](#) on page 656 to observe intermediate differential voltages.

Figure 8-17 Passband View

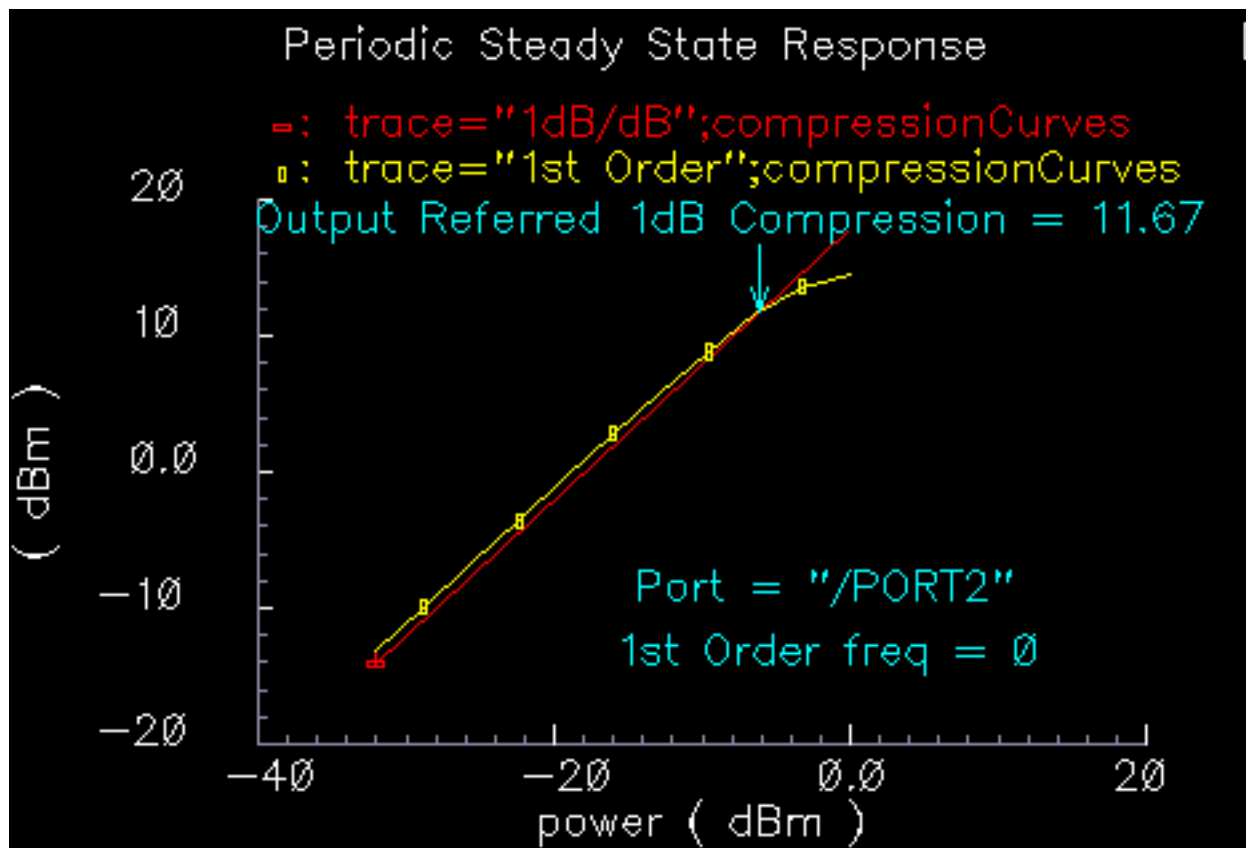


6. Check and save the schematic
7. Close the schematic window.
8. Bring up a Library Manager and select but do not open the schematic you just created.
9. In the Library Manager window click File->New->Cell View.
10. In the window that comes up, select Hierarchy Editor for the Tool. Click OK.
11. Type in "schematic" for the View then click "Use Template."
12. Set the Name to Spectre
13. Enter "veriloga\_PB" as the first item in the View List then click OK.
14. In the HE (Hierarchy Editor) window, click File->Save. This is important.
15. Click the Open button to bring up the schematic.
16. Bring up an Analog Environment tool and use the states from the last Circuit Optimizer iteration.
17. Recall the states you saved from the last Circuit Optimizer iteration. Add the "power" variable and set it to -16.
18. Delete the previous Analyses and set up a PSS analysis. Enter a new Fundamental Tone named "LO" and make it 2.354GHz. Auto Calculate the Beat Frequency and enter "1" for the Number of harmonics.



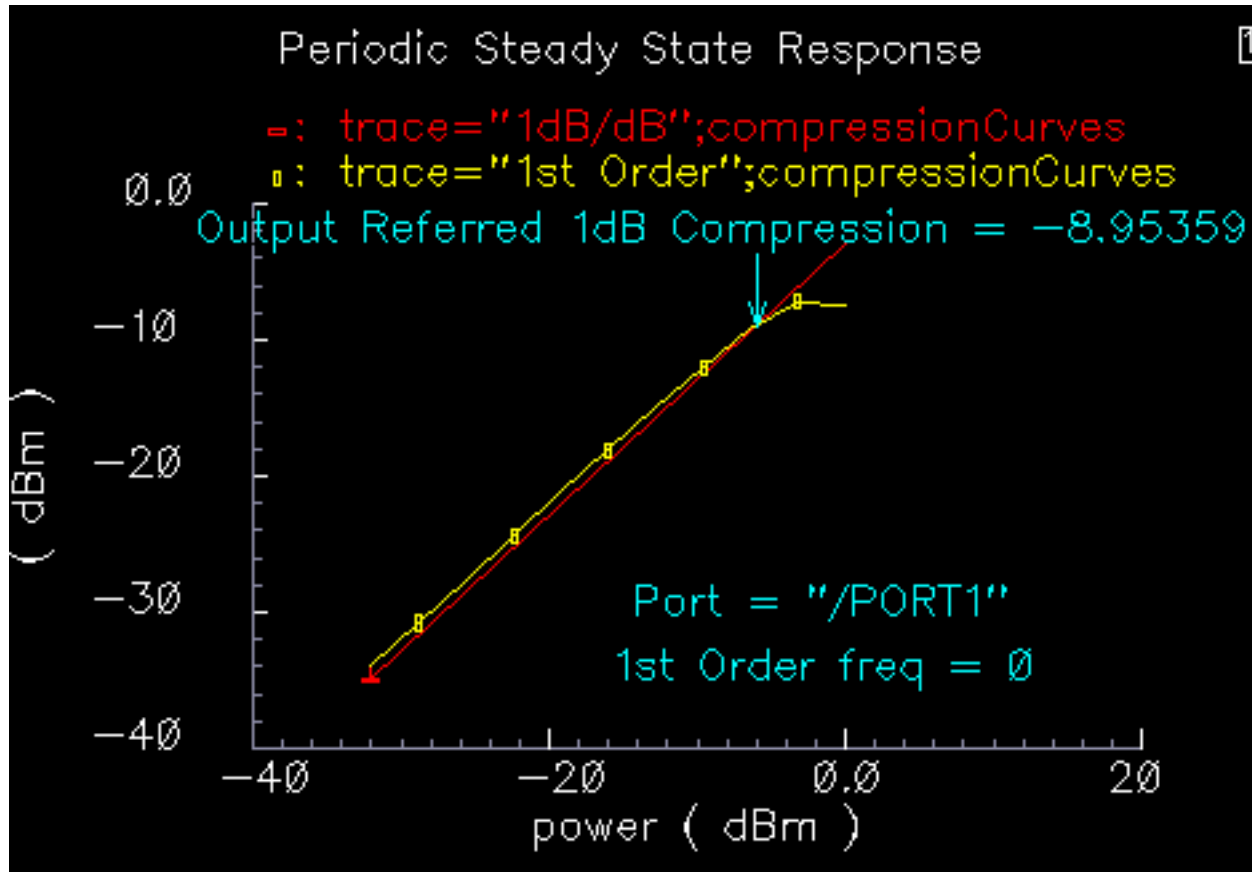
19. Run the analysis and observe the intermediate differential voltages. The model is indeed now a passband model. At the higher power levels the LNA output contains odd harmonics of the RF carrier. The filter reflects the odd harmonics back to the LNA and does not let them propagate forward. The baseband model does not simulate the odd harmonics but it does simulate the intermodulation term between the second harmonic and fundamental that falls at the fundamental. One reason to simulate the passband view is to check for peak voltage levels that might exceed voltage ratings. The baseband models only simulate peak voltage at the carrier fundamental, not the absolute peak.
20. Set up swept PSS analysis. Sweep "power" from -32 to 0 in 10 steps.
21. After the sweep finishes, click Results->Direct Plot in the Environment window, select Compression point, 1dB, Output Referred. Select the 0 harmonic because the end-to-end system produces a baseband output. Then click the port loading the top output low pass filter. You should see the compression point plot show in [Figure 8-18](#) on page 657.

**Figure 8-18 End-to-end RF measurement, one dB compression point at the I-baseband output.**



22. Repeat the last step but this time click the lower output port. You should see the compression plot in [Figure 8-19](#) on page 658.

**Figure 8-19** End-to-end RF measurement, one dB compression point at the Q-baseband output.



## Comparing Baseband and Passband Models

This section illustrates how to compare baseband and passband models by:

- Setting up a Transient analysis with the passband view
- Setting up a Transient analysis with the baseband view
- Directly comparing the baseband and passband models.
- You run one analysis of the baseband view and two analyses of the passband view. You perform the second passband analysis with tightened tolerances.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

1. Save the passband schematic under a different name. You use the new copy.
  2. Repeat steps 9 through 17 from the last recipe for the new copy but do not enter the “veriloga\_PB” view in the View List yet. You do a baseband analysis first.
  3. Delete the port driving the LNA.
  4. You can also delete the loaded voltage-controlled-voltage sources.
  5. You need to synthesize an antenna signal. Add an IQ\_mod\_BB from the top\_dwnBB category. Set the I and Q gains to 0 dB. Set the 1dB compression points to 1000 so that the modulator is ideal. Instantiate it in front of the LNA with the pins aligned then wire the pins straight across. Ground the “phase\_err” pin.
  6. Drive the “I\_in” pin of the IQ modulator with a port. Set the port frequency to 2MHz and name the frequency BB1. Set the amplitude to -16dBm.
  7. Do the same for the “Q\_in” modulator input.
  8. Load or duplicate the states from the 12-iteration Circuit Optimizer analysis but delete the Noise and Transient analyses.
  9. Remove AM/PM conversion from the LNA by setting the last parameter in the properties list to zero. It is not fair to compare passband and baseband views with AM/PM conversion because the passband view does not capture it.
  10. Set up a 1us Transient analysis with default options.
  11. Run the analysis and plot the filtered baseband outputs, the outputs of the low pass filters. Note how fast the simulation runs. Save the results so you can plot them again later. <sup>1</sup>
  12. Switch to the passband view by entering “veriloga\_PB” in the View List in the Hierarchy Editor. Click the update button in the HE.
  13. After you switch to the veriloga\_PB view, edit the IQ modulator properties to set “flo” to frf. Edit the demodulator properties and set its “flo” to flo1-frf.
  14. Click Results->Printing Plotting Options then click the “Overlay Plots” button.
  15. Overlay the passband results with the baseband results. You should see the waveforms in [Figure 8-20](#) on page 660. The comparison is not very good.
- 
1. Note that the baseband outputs are out of phase with each other, even though the baseband inputs are in phase. In the baseband model, changing the RF-IF mixer LO from “flo1” to “-flo1” fixes the sign problem. In the passband model, the IQ\_demodulator flo should be frf-flo1. To maintain the convention, in the baseband model the IF filter’s carrier frequency should be frf-flo1.

16. Rerun the analysis with conservative options and set reltol to 1e-6. This run takes a while.
17. Plot the results.
18. Recall the saved baseband results and overlay them with those from the last simulation. You should see the waveforms in [Figure 8-21](#) on page 661. The passband results now lay right on top of the baseband results but took much longer to compute! It was not obvious without the baseband results that the first passband simulation did not run with tight enough numerical tolerances.

**Figure 8-20 Passband and baseband results with default options in the passband analysis**

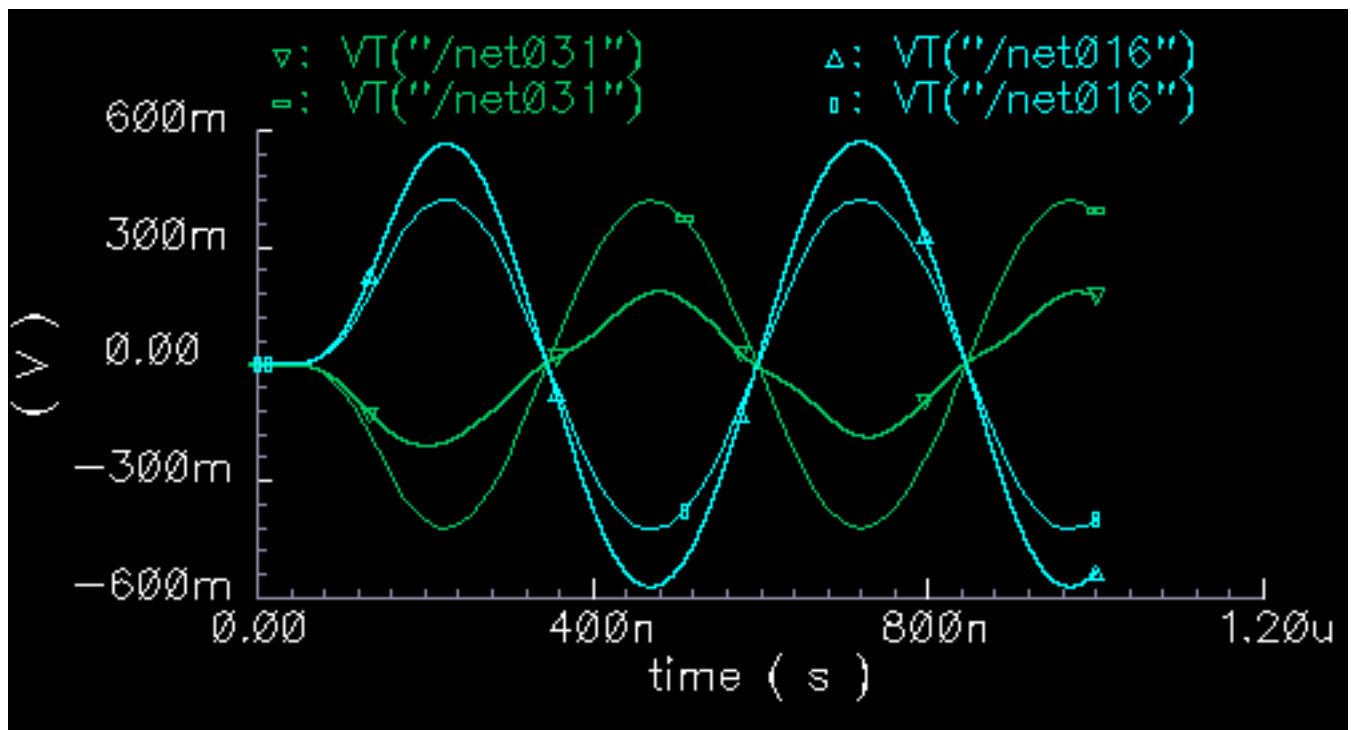
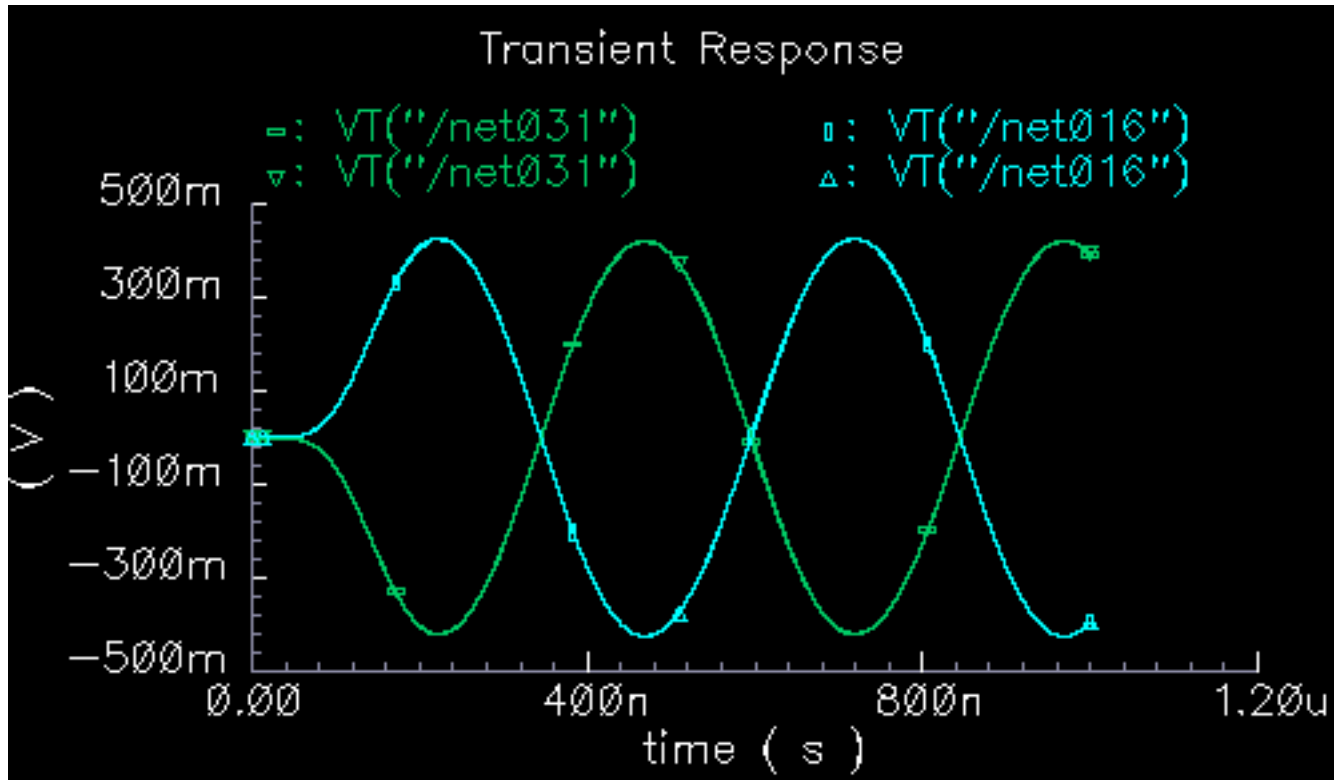


Figure 8-21 Passband and baseband results with tighter options in the passband analysis



## Relationship Between Baseband and Passband Noise

Noise analysis at baseband can be confusing because factors of two appear in a number of places throughout the calculations. For example:

- Each passband node becomes two equivalent baseband nodes.
- Does noise injected at a passband node split between the two equivalent baseband nodes?
  - If so, does the noise split evenly?
- As shown in the BB\_test\_bench example, baseband models simulate peak in-phase and peak quadrature components of the carrier.
- When analyzing signal-to-noise ratios, does that mean you have to use half the square of the baseband signals?

- The analog design environment displays single sided power spectral densities.
- Since the baseband power spectral density is the two-sided passband density shifted down, is there another factor of two because we can only see the baseband density for positive frequencies?
- White noise at the input of a mixer mixes up to the carrier from DC but there is also noise at twice the carrier that mixers down to the carrier.
- Does the baseband model account for this?

These questions are answered in this section.

Before sorting out the factors of two, note that baseband noise analysis is valid only for small signals. If any element in the architecture operates in a non-linear fashion, the noise analysis might be inaccurate. This is due to the fact that a baseband noise analysis follows a DC operating point analysis, rather than a PSS analysis.

Instantaneous incremental gain in a passband static non-linear model dithers at the carrier frequency.

When the carrier swings through zero, the incremental gain is large and noise at the input is amplified.

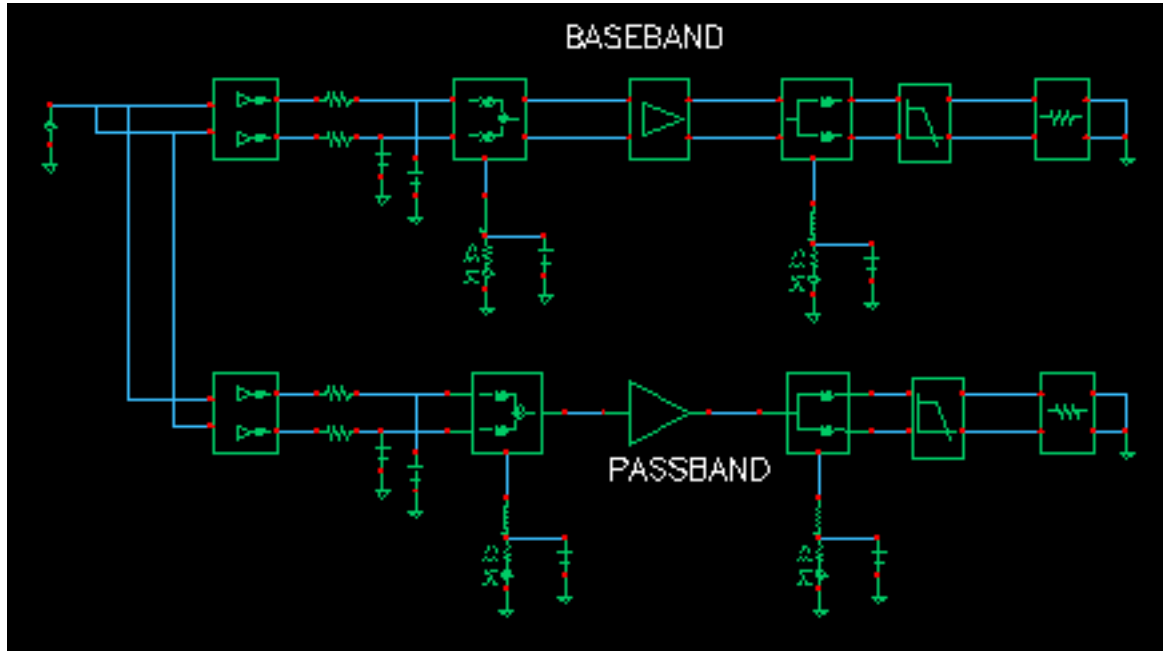
When the carrier reaches its peak and drives the circuit into saturation, the incremental gain is smaller.

The average gain is greater than the minimum gain. The baseband model remains in the non-linear region because it only simulates peak voltages. Consequently, the incremental gain is always a minimum and the baseband model under-estimates the amount of noise propagating to the output. If the peak input signal drives the model into saturation, be sure to scale the baseband noise results accordingly.

## Introduction to Analysis

The circuit discussed here is called *noise\_test\_circuit* and you can find it in the *rfExamples* library. The circuit looks as shown in [Figure 8-22](#) on page 663.

Figure 8-22 The `noise_test_circuit` in the `rfExamples` Library



The `noise_test_circuit` shows the relationship between baseband and passband noise. One branch consists of passband models. The other branch is a baseband equivalent of the first. You can assess noise at each of three observation points located in each branch of the circuit. At each observation point, you can examine both the noise and pnoise summaries.

The I and Q inputs are both driven by the same DC source so that you only have to view one baseband output, the other baseband output is identical by symmetry. Noise parameters in the passband and baseband models are identical. Aside from the behavioral blocks at the end of each branch, each behavioral block has noise injected at its input.

## Prep Steps for Analyses

**Setup a PSS analysis.** Since the local oscillator is inside the passband mixer models you have to manually enter the frequency (1GHz) into the PSS analysis form. Let the beat frequency be *autocalculated* and use 1 harmonic.

**Setup a pnoise analysis** with the start frequency equal to 0 Hz, the stop frequency equal to 100MHz, use a linear sweep with 100 steps. Set the Maximum sideband to 1. For the input source select *none* and for the output use *voltage*. For the positive output node, select the I-input of the IQ\_modulator in the passband branch. Use ground for the negative node.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

**Setup a noise analysis** that sweeps frequency from 0 to 100MHz linearly in 100 steps. Use voltage for the output noise and select the I-input of the *IQ\_mod\_BB* component in the baseband branch for the positive node. Select ground for the negative node. Set the input noise port to any one of the ports in the circuit. Noise from that port does not affect either passband or baseband branches.

#### **Run the analysis.**

When the analysis finishes, go to the analog design environment simulation window and click **results-print-pss noise summary**.

In the form that appears, include All types, set *Type* to *integrated noise*, look at noise from 0 to 100MHz, select *truncate by number*, and view the top 2 noise contributors. There are only 2 noise contributors at this point, noise at the I/Q inputs and noise from the low pass resistors.

Now print the noise summary. (This is different from the pnoise summary.) Again, print integrated noise (from 0 to 100MHz). Select All types and print noise from the top 2 noise contributors.

Figure 8-23 on page 665 shows the noise summaries. The two summaries agree because at this point in the circuit, both nodes are really baseband nodes.



**Figure 8-23 Noise Summaries for the First PSS PNoise and Noise Analyses**

Device	Param	Noise Contribution	% Of Total
/I2	IQ_modulator_i	1.56676e-12	70.06
/R20	rn	6.69678e-13	29.94
Integrated Noise Summary (in V <sup>2</sup> ) Sorted By Noise Contributors			
Total Output Noise = 2.23644e-12			
No input referred noise available			
Device	Param	Noise Contribution	% Of Total
/I22	IQ_mod_BB_i	1.56676e-12	70.06
/R22	rn	6.69679e-13	29.94
Integrated Noise Summary (in V <sup>2</sup> ) Sorted By Noise Contributors			
Total Output Noise = 2.23644e-12			
Total Input Referred Noise = Inf			

**Repeat the analyses** but this time select the “out” pins on the low pass filters as outputs in the appropriate noise analyses. Again print the same noise summaries but this time look at the top 12 noise contributors in each summary.

Figure 8-24 on page 666 shows the noise summaries for the second analysis. The top 7 noise contributors in the baseband and passband branches agree. The remaining noise contributors are negligible, and should be negligible for the circuit since it has no AM/PM conversion.

**Figure 8-24 Noise Summaries for the Second PSS PNoise Analyses**

Device	Param	Noise Contribution	% Of Total
/I77	PA_FB	7.49879e-10	82.29
/I2	IQ_modulator_i	9.69448e-11	10.64
/R20	rn	4.1437e-11	4.55
/I87	butterworth_lp_i_noise	1.52928e-11	1.68
/I84	IQ_demodulator	7.64648e-12	0.84
/PORT13	rn	3.77571e-14	0.00
/PORT14	rn	3.77571e-14	0.00
/I2	IQ_modulator_q	2.33139e-15	0.00
/R21	rn	9.96506e-16	0.00
/R22	rn	0	0.00
/R23	fn	0	0.00
/R23	rn	0	0.00

Integrated Noise Summary (in V<sup>2</sup>) Sorted By Noise Contributors  
 Total Output Noise = 9.11279e-10  
 No input referred noise available

Device	Param	Noise Contribution	% Of Total
/I76	PA_BB_i	7.49861e-10	82.29
/I22	IQ_mod_BB_i	9.69448e-11	10.64
/R22	rn	4.14371e-11	4.55
/I86	butterworth_lp_i_noise	1.52928e-11	1.68
/I84	IQ_demod_BB_i	7.64642e-12	0.84
/PORT15	rn	3.77571e-14	0.00
/PORT9	rn	3.77571e-14	0.00
/I76	PA_BB_q	1.90329e-14	0.00
/I22	IQ_mod_BB_q	2.33136e-15	0.00
/R23	rn	9.96423e-16	0.00
/I82	IQ_demod_BB_q	1.54184e-31	0.00
/R23	fn	0	0.00

Integrated Noise Summary (in V<sup>2</sup>) Sorted By Noise Contributors  
 Total Output Noise = 9.11279e-10  
 Total Input Referred Noise = 0.0051541

Now, repeat the analysis but this time change the Pnoise sweep to run from 900MHz to 1.1GHz in 200 linear steps and select the power amplifier output as the noise output node. For the noise analysis, leave the sweep at zero to 100MHz, use 100 steps as before, and change the output node to be the I-output of the power amplifier.

Look at the top 7 noise contributors in each analysis. This time, integrate noise from 900MHz to 1.1GHz for the pnoise run and integrate noise from 0 to 100MHz for the noise run. Figure 8-25 on page 667 shows the new summaries. Although noise analyses agree at either ends of the branches, noise analyses seem to disagree at a point where the baseband node is only a baseband equivalent, not a true baseband node.

**Figure 8-25 Noise Summaries for the Third PSS PNoise Analyses**

Device	Param	Noise Contribution	% Of Total
/I77	PA_PB	2.3769e-09	83.70
/I2	IQ_modulator_i	1.53663e-10	5.41
/I2	IQ_modulator_q	1.53663e-10	5.41
/R20	rn	6.568e-11	2.31
/R21	rn	6.568e-11	2.31
/I84	IQ_demodulator	2.42376e-11	0.85
/PORT13	rn	1.19682e-13	0.00

Integrated Noise Summary (in V<sup>2</sup>) Sorted By Noise Contributors  
 Total Output Noise = 2.83995e-09  
 No input referred noise available

Device	Param	Noise Contribution	% Of Total
/I76	PA_BB_i	2.3769e-09	83.69
/I22	IQ_mod_BB_i	3.07294e-10	10.82
/R22	rn	1.31347e-10	4.62
/I82	IQ_demod_BB_i	2.42376e-11	0.85
/PORT9	rn	1.19682e-13	0.00
/I76	PA_BB_q	5.71606e-14	0.00
/I22	IQ_mod_BB_q	7.38993e-15	0.00

Integrated Noise Summary (in V<sup>2</sup>) Sorted By Noise Contributors  
 Total Output Noise = 2.83997e-09  
 Total Input Referred Noise = 0.0148587

The *apparent* disagreement shown in [Figure 8-25](#) on page 667 requires an explanation. Let us examine the noise contributors and try to answer some of the questions we posed earlier.

**Noise at the power amplifier output due to noise injected at a passband node:**

- Passband model contributors: *PA\_PB* and *IQ\_demodulator*, port 9.
- Baseband model contributors: *PA\_BB\_i*, *IQ\_demod\_BB\_i*, port 13.

Passband and baseband counterparts contribute the same amount of noise. However, in the baseband model, from symmetry you see the same numbers if you look at the Q-node. This means the baseband model predicts twice as much total noise due to noise injected between the modulators.

This factor of two is intentionally introduced to maintain the correct signal-to-noise ratio. The baseband model simulates peak signals; the carrier is suppressed. Without the carrier, signal power equals the square of the peak rather than one half of the square. This factor of two is not as arbitrary as it seems. The baseband model predicts the correct noise after demodulation because the passband demodulator model includes an extra factor of two to offset the factor of two inherent in the demodulation process.

Let the modulated carrier be  $i(t) \cdot \cos(\omega c \cdot t) - q(t) \cdot \sin(\omega c \cdot t) + \text{noise}(t)$ .

Now consider the I-output. To generate the I-output, the demodulator multiplies the signal by  $\cos(\omega c \cdot t)$ . The only part that propagates through the subsequent filter is  $(1/2)i(t) + \text{noise}(t) \cdot \cos(\omega c \cdot t)$ . To recover  $i(t)$ , the passband demodulator model must scale this sum by two. (The baseband demodulator does not need to scale by two to extract the baseband signal because the carrier is suppressed.)

Thus, noise at passband demodulator model output equals  $2 \cdot \text{noise}(t) \cdot \cos(\omega c \cdot t)$ . The filtered noise power density is then  $4 \cdot (\text{input noise density}) / 2$ . The factor of 1/2 comes from the cosine. The filtered output noise density is twice the input noise density.

In the baseband model, doubling the noise injected at the passband nodes was not simply a matter of convenience.

**So, to answer questions 1 and 2**

- Yes, noise injected at a passband node splits evenly between the two equivalent baseband nodes  
  
but
- Since each split is doubled, the ratio of *peak* signal to total noise equals the true signal-to-noise ratio.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Methods for Top-Down RF System Design

---

- Question 3 is rendered moot by using the noise summary and integrating over the proper band. If the noise analysis only integrated from 1GHz to 1.1GHz, (instead of from 900MHz to 1.1GHz), there would be a mysterious factor of two error.
- In the baseband model, phase noise entering on the phase error pin propagates to both the I and Q outputs. In the passband model, the same noise power appears on just the one output. Again, the total noise in the baseband model is twice that of the passband model to maintain the correct signal-to-noise ratio.
- Noise injected at the modulator input (resistors and modulator noise):

Total noise in the baseband model due to modulator and input resistor noise is twice what it is from just one phase. Thus, the total noise due to sources on the input sides of the modulators differ by a factor of two. This occurs because the passband model is a real multiplier, which modulates the noise. If the peak signal voltage agrees with the baseband model, the passband modulator model attenuates input noise most of the time. The important thing is that the signal-to-noise ratios in the passband and baseband models agree anywhere in the system.

Now, copy the circuit so you can **remove the capacitors at the modulator inputs** and repeat the last set of noise analyses. You see that in the passband model, the input resistors, R20 and R21, together contribute twice as much as the baseband counterpart, R22. With the capacitors, R20 and R21 together contribute just as much as R22. Without the capacitors, the input noise is truly white over the frequencies of interest. The same thing happens to the modulator noise itself. [Figure 8-26](#) on page 670 shows the results.

In particular, the modulator now also has noise at twice the carrier frequency and that noise mixes down to the carrier frequency. The baseband model is just that, a baseband model. The answer to question 4 is *no*. The baseband models do not account for noise, or signals, at carrier harmonics. The baseband equivalent noise analysis is valid only if noise injected into the modulators has no power beyond the local oscillator frequency. Phase noise injected at the phase noise pins should also be band limited.

**Figure 8-26 Noise Summaries with the input capacitors removed.**

Device	Param	Noise Contribution	% Of Total
/I2	IQ_modulator_q	4.22571e-09	29.21
/I2	IQ_modulator_i	4.22571e-09	29.21
/I77	PA_PB	2.3769e-09	16.43
/R21	rn	1.80619e-09	12.49
/R20	rn	1.80619e-09	12.49
/I84	IQ_demodulator	2.42376e-11	0.17

Integrated Noise Summary (in V<sup>2</sup>) Sorted By Noise Contributors  
 Total Output Noise = 1.44651e-08  
 No input referred noise available

Device	Param	Noise Contribution	% Of Total
/I22	IQ_mod_BB_i	4.22561e-09	50.11
/I76	PA_BB_i	2.3769e-09	28.19
/R22	rn	1.80615e-09	21.42
/I82	IQ_demod_BB_i	2.42376e-11	0.29
/PORT9	rn	1.19682e-13	0.00
/I22	IQ_mod_BB_q	1.01619e-13	0.00

Integrated Noise Summary (in V<sup>2</sup>) Sorted By Noise Contributors  
 Total Output Noise = 8.43322e-09  
 Total Input Referred Noise = 9.76997e-11

---

## Cosimulation with MATLAB and Simulink

---

This chapter describes how to set up and use a cosimulation link between the MATLAB<sup>®</sup> and Simulink<sup>®</sup> system-level simulation environment and Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF). The sections in this chapter are:

- [Introduction to Cosimulation with MATLAB](#) on page 672
- [Software Requirements](#) on page 672
- [Setting Up and Running a Cosimulation](#) on page 672
- [Connecting the Coupler Block Into the System-Level Simulink Schematic](#) on page 673
- [Determining How You Want to Start and Run the Cosimulation](#) on page 677
- [Generating a Netlist for the Lower-Level Block](#) on page 677
- [Running the Cosimulation](#) on page 685

## Introduction to Cosimulation with MATLAB

Cosimulation combines the best of system-level simulation with lower-level analog and RF simulation. Simulink provides large libraries of DSP algorithms for generating complicated signals and post processing while Spectre RF supports transient and envelope analysis of common RF and communication circuits such as mixers, oscillators, sample and holds, and switched capacitor filters at both the transistor and behavioral levels.

Cosimulation makes it easier

- To detect concept errors early
- To detect design flaws before tape-out
- To quickly correct issues and resimulate

The system-level design in Simulink serves as a golden reference. System designers can use the Simulink system-level design as a testbench for implementing and simulating the design. Unfortunately, when system-level designs are simulated by themselves, the effects originating from subsystems are often not considered. With cosimulation, system designers can create low-level models of critical analog blocks and use these models one at a time to analyze the performance impact of individual blocks on the system-level simulation.

## Software Requirements

Cadence recommends that you use the following software versions: MMSIM6.1 USR1 or later, and MATLAB R14 or later, with Simulink and the signal process block set installed.

## Setting Up and Running a Cosimulation

To prepare for and run a cosimulation, you

1. Connect the coupler block into the system-level Simulink schematic.
2. Determine how you want to start and run the cosimulation.
3. Generate a netlist for the lower-level block that reflects how you want to start and run the cosimulation.
4. Run the cosimulation.

These steps are described in greater detail in the following sections.

Before you continue, however, be sure that the programs are ready to run.



- Add the Spectre RF and MATLAB/Simulink engine install paths to the `MATLABPATH` environment variable.

You can automate this step by adding the appropriate command to your `.cshrc` file. For example, if you are using a C shell, you can add the command

```
setenv MATLABPATH `cds_root spectre`/tools/spectre/simulink:${MATLABPATH}
```

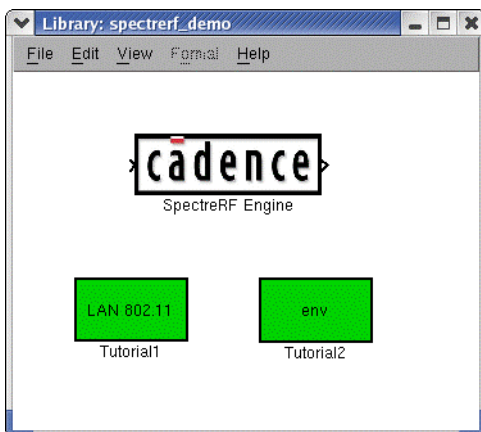
## Connecting the Coupler Block Into the System-Level Simulink Schematic

To prepare the Simulink part of the design for cosimulation,

1. Start MATLAB, by typing

```
matlab &
```

The Simulink library opens.



This library contains the coupler module (distinguished by the Cadence logo and labeled `SpectreRF Engine`). You can insert the coupler module in any Simulink design by dragging and dropping it from this library.

2. Open your testbench or high-level design.
3. Drag and drop the `SpectreRF Engine` coupler block into the testbench and place it in the design.
4. Connect the `SpectreRF Engine` block into the design.

To make a signal connection, move the mouse pointer over the module pin. The pointer changes to a cross. Press the left mouse button, move the cursor to the destination pin and release the button.

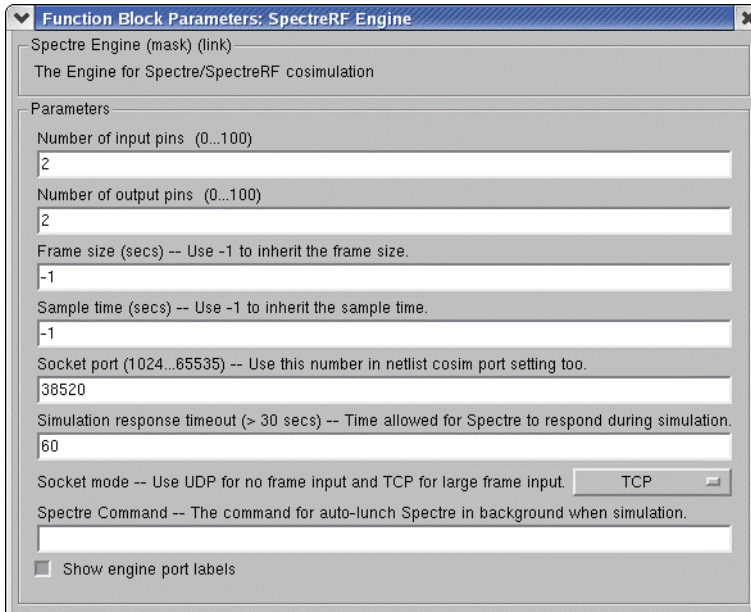
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Cosimulation with MATLAB and Simulink

---

#### 5. Double-click the SpectreRF Engine block.

This opens the Function Block Parameters: SpectreRF Engine window where you can edit the coupler block's parameter values.



The fields have the following meanings:

Field	Meaning
<i>Number of input pins</i>	The number of input pins to the SpectreRF Engine block (0...100). The input to the block is the input signal from the Simulink design.
<i>Number of output pins</i>	The number of output pins from the SpectreRF Engine block (0...100). The output from the block is the signal generated by the Spectre RF simulator.
<i>Frame size</i>	The number of samples per frame (any positive integer or -1; default: -1). Allows you to use the coupler in a loop where frame size inheritance does not work. Simulink automatically detects conflicting frame sizes. Set this value to -1 to inherit the frame size from the input.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Cosimulation with MATLAB and Simulink

<i>Sample time</i>	Sample period of the block, in seconds (default: -1). The <i>Sample time</i> is the frame period, not the time between the samples inside the frame. Allows the coupler to have only outputs, where the Simulink coupler acts as a source and defines a <i>Sample time</i> . Also allows the coupler to have only inputs. Set this value to -1 to inherit the sample period of the connected blocks. See the MATLAB and Simulink <i>Help</i> menu for more information on <i>Sample time</i> propagation.
<i>Socket port</i>	The number of the service that is identified by TCP/IP. Normally, the system reverses port numbers less than 1024, so this value can range from 1024 to 65535 (default: 38520).
<i>Simulation response timeout</i>	Maximum time, in seconds, to wait for an answer from the Spectre RF simulator during simulation. Increase this time if the simulator requires a long calculation time for each sample or frame (default: 120).
<i>Socket mode</i>	TCP or UDP. Choose UDP if frame size is less 50. TCP and UDP are two different translate protocols in TCP/IP. Without CRC, UDP is usually faster in a good net environment with small data packages and TCP is usually better for large data packages.
<i>Spectre command</i>	Used to run a Spectre RF simulation. This parameter enables a use mode where Spectre RF is called internally by the Simulink simulation.
<i>Show engine port labels</i>	If checked, the SpectreRF Engine shows label information.

6. Set the number of input pins and output pins as well as any other parameters you need to set.

7. Note the Spectre command field but do not change the value now.

Depending on how you choose to run the cosimulation, you might need to return to this field and type in a `spectre` command.

8. Click *OK*.

The form closes and the SpectreRF Engine block is updated with the correct number of pins.

9. (Optional) Make other necessary changes in the Simulink testbench window.

For example,

- ❑ Select *Format — Port/Signal Displays — Signal Dimensions*.

This switches the signal dimension display on so you can see details of framed signals.

- ❑ Select *View — Simulink library* to display the library browser. Double-click *Math Operations*.

The library appears. Here you can select Simulink converters, such as a `Complex to Real-Imag` block. Drag and drop blocks into the schematic and connect them as necessary.

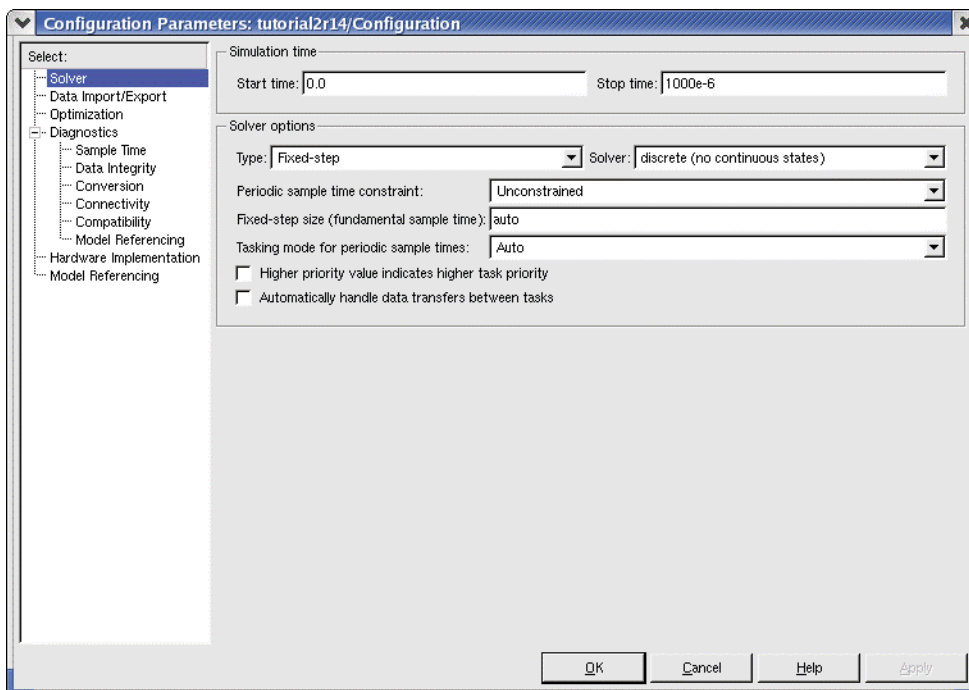
Double-click any placed blocks and set appropriate values.

10. Choose *Simulation — Configuration Parameters*.

The Configuration Parameters form opens.

11. In the *Stop time* field, set the stop time.

The Configuration Parameters form looks something like this:



12. Click *OK*.

The Configuration Parameters form closes.

13. Save the modified design by choosing *file* — *save as*.

This completes the necessary modifications to the Simulink testbench.

## Determining How You Want to Start and Run the Cosimulation

There are three ways to run a cosimulation, after all the setup is finished.

1. You can start the two applications (Spectre RF and MATLAB) separately.

This method is appropriate if you need to be able to modify both the system-level Simulink design and the analog circuit.

2. You can start ADE and arrange to have MATLAB start automatically.

This method is appropriate if you are an analog design who needs to validate a circuit with system-level design input and output.

3. You can start MATLAB and arrange to have Spectre RF start automatically.

This method is appropriate if you are a system-level designer, because, after set up, you can use the SpectreRF Engine just like another block in Simulink.

The setup differs for each of these approaches so it is useful to decide which is most appropriate for your design before continuing.

## Generating a Netlist for the Lower-Level Block

The previous section describes how to insert a coupler block into the system-level design. That defines one end of the connection but you must still establish a connection with the lower-level analog block that is simulated by Spectre RF. To do that, you insert a `cosim` statement into the netlist, either by hand or by using the ADE environment.

### Preparing the Netlist When Using ADE

This section describes how to prepare a netlist for cosimulation using the Analog Design Environment (ADE).

1. Start the `icms` tool.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

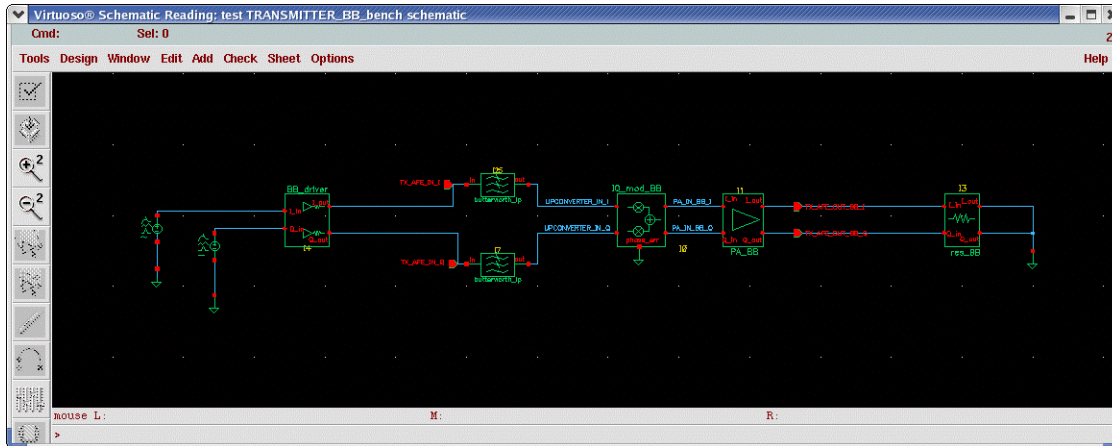
## Cosimulation with MATLAB and Simulink

icms &

The version of `icms` must be release IC5.1.4.1 USR4 ISR62 or later.

2. Open the schematic view of the cell.

For example,



3. Open the Virtuoso Analog Design Environment from the schematic editor by choosing *Tools — Analog Environment*.
4. Choose *Setup — Matlab/Simulink — Setting*.

The Cosimulation Options form appears.

5. In the Cosimulation Options form, click the *Select* button located beside *Cosimulation inputs*. Switch to the schematic viewer, where you see the following information below the schematic.

Select source instance as cosimulation inputs. Press Esc when done.

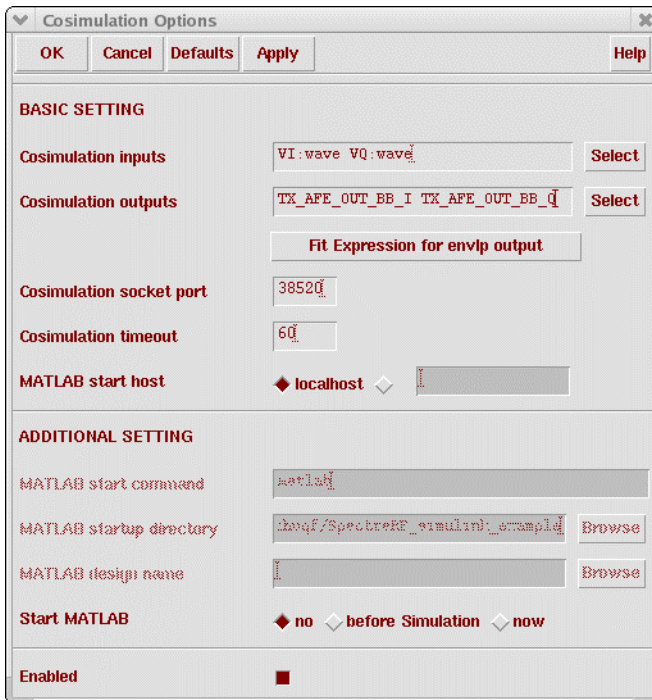
Select the sources (which are connected to the outputs from the Simulink level of the design), then press the *Esc* key.

6. In the Cosimulation Options form, click the *Select* button beside *Cosimulation outputs*. Switch to the schematic viewer, where you see the following information below the schematic.

Select Net/Terminal as cosimulation outputs. Press Esc when done

Select the outputs (which are connected to the inputs at the Simulink level of the design), then press the *Esc* key.

The Cosimulation Options form looks something like this:



7. Ensure that the value of *Cosimulation socket port* is the same as the port value of the SpectreRF Engine block defined in [“Connecting the Coupler Block Into the System-Level Simulink Schematic”](#) on page 673.
8. In the Cosimulation Options form, turn on *Enabled*.
9. Examine the possible values for the *Start MATLAB* field.

Value	Behavior
<i>now</i>	ADE launches a MATLAB session and opens a MATLAB desktop that is no different than it would be if you launched it manually. When you use the <i>now</i> value, you must start the simulation in the Simulink design before starting the Spectre RF simulation.  Using the <i>now</i> value is a good way to test whether ADE can start MATLAB and open the design successfully.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Cosimulation with MATLAB and Simulink

---

Value	Behavior
<i>before Simulation</i>	<p>ADE launches an internal MATLAB session and no MATLAB desktop appears. ADE opens the specified MATLAB design and initiates the MATLAB simulation before Spectre simulation starts. ADE starts MATLAB once but cannot start it again before you close the first session.</p> <p>Cadence suggests using the <i>now</i> value as a test before you run with the <i>before Simulation</i> value.</p> <p>The <i>now</i> and <i>before Simulation</i> values share one log file, which can be opened by choosing <i>Setup — Matlab/Simulink — Log file</i>. You can use the log file to monitor the simulation when no MATLAB desktop is visible.</p>
<i>no</i>	<p>The <i>MATLAB start command</i>, <i>MATLAB startup directory</i>, and <i>MATLAB design name</i> fields become active. When you use the <i>no</i> value, you must start the simulation in the Simulink design before starting the Spectre RF simulation.</p>

10. Continue the process of preparing the netlist according to how you want to start the applications that run the cosimulation.

To use this starting method...	Follow the guidance in this section...
Start the two applications (Spectre RF and MATLAB) separately.	<a href="#">“Preparing to Start the Two Applications Separately”</a> on page 680
Start ADE and arrange to have MATLAB start automatically.	<a href="#">“Preparing to Start ADE Manually and MATLAB Automatically”</a> on page 681
Start MATLAB and arrange to have Spectre RF start automatically.	<a href="#">“Preparing to Start MATLAB Manually and Spectre RF Automatically”</a> on page 682

### Preparing to Start the Two Applications Separately

1. Set the *start MATLAB* value to *no*.
2. Click *OK* to close the Cosimulation Options form.
3. Use the Choosing Analyses form to set up the analysis.

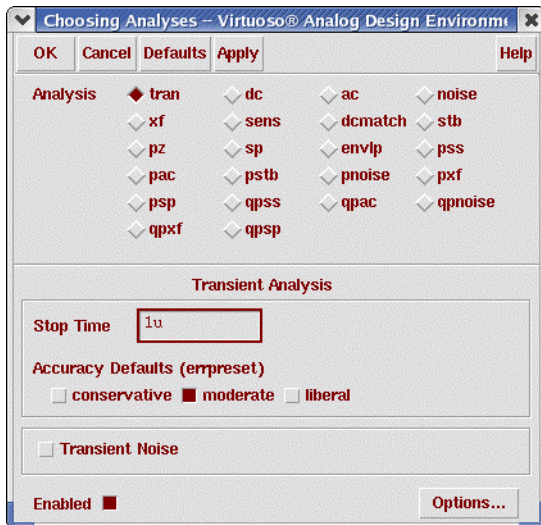


## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Cosimulation with MATLAB and Simulink

The *Stop Time* can be any value. The Spectre RF simulator synchronizes the stop time with Simulink.

The Choosing Analyses window looks something like this:



4. Create a netlist and make sure the Simulink `cosim` statement appears in it.

### Preparing to Start ADE Manually and MATLAB Automatically

1. Set the value of the *Start MATLAB* field to *before Simulation*.
2. Type the name of the MATLAB design into the *MATLAB design name* field.

Such designs have an extension of `.mdl`. For example, `env_d.mdl`. The MATLAB design must be in a location that is included in the `MATLABPATH` environment variable.

3. Set the value of the *Start MATLAB* field to *now*.

A new MATLAB application starts and helps you open the Simulink design.

4. Double-click the `SpectreRF Engine` in the design and set the parameters.
  - a. Set the *Sample time*. The Sample time can be set to -1, if you are uncertain about the appropriate time to use.
  - b. Set *Socket Port* to the value you set in the Cosimulation Options form.
  - c. Leave the *Spectre Command* field empty.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Cosimulation with MATLAB and Simulink

---

This field is used when you wish to start Spectre RF automatically after starting MATLAB manually.

Function Block Parameters: SpectreRF Engine

Spectre Engine (mask) (link)  
The Engine for Spectre/SpectreRF cosimulation

Parameters

Number of input pins (0...100)  
1

Number of output pins (0...100)  
1

Frame size (secs) -- Use -1 to inherit the frame size.  
-1

Sample time (secs) -- Use -1 to inherit the sample time.  
2.5e-4

Socket port (1024...65535) -- Use this number in netlist cosim port setting too.  
38525

Simulation response timeout (> 30 secs) -- Time allowed for Spectre to respond during simulation.  
60

Socket mode -- Use UDP for no frame input and TCP for large frame input. UDP

Spectre Command -- The command for auto-lunch Spectre in background when simulation.

Show engine port labels

OK Cancel Help Apply

- d. Click **OK** to close the SpectreRF Engine form.
5. In the Cosimulation Options form, set *Start MATLAB* to *before Simulation*.
6. Click **OK** to close the Cosimulation Options form.
7. Use the Choosing Analyses form to set up the analysis.
8. In the ADE form, choose *File* — *Save* to save the design. Then exit from the MATLAB that was opened by ADE.

### Preparing to Start MATLAB Manually and Spectre RF Automatically

1. Set the value of the *Start MATLAB* field to *now*.  
A new MATLAB application starts and helps you open the Simulink design.
2. Double-click the SpectreRF Engine in the design and set the parameters.
  - a. Set the *Sample time*. The Sample time can be set to -1, if you are uncertain about the appropriate time to use.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

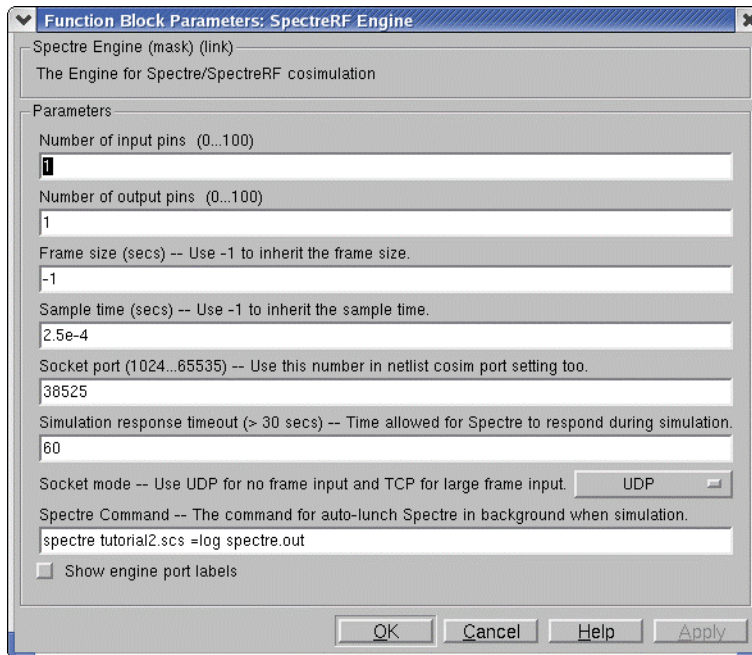
### Cosimulation with MATLAB and Simulink

---

- b. Set *Socket Port* to the value you set in the Cosimulation Options form.
- c. Type a command similar to the following into the *Spectre Command* field.

```
spectre netlist_file =log spectre.out
```

The Function Block Parameters: SpectreRF Engine window looks something like this:



- d. Click *OK* to close the Function Block Parameters: SpectreRF Engine form.
3. In the Cosimulation Options form, set *Start MATLAB* to *no*.
4. Click *OK* to close the Cosimulation Options form.
5. Use the Choosing Analyses form to set up the analysis.
6. In the ADE form, choose *File* — *Save* to save the design. Then exit from the MATLAB that was opened by ADE.

## Preparing the Netlist Without Using a Graphical User Interface

The steps in this section are necessary only if Spectre RF needs to start MATLAB automatically. If you are using the “Start MATLAB and arrange to have Spectre RF start automatically” starting method, ensure that there is no `cosim` statement in the netlist.

1. Edit the netlist file.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Cosimulation with MATLAB and Simulink

---

For example,

```
vi tutorial1/tutorial1.scs.
```

#### 2. Add a `cosim` statement to the netlist.

For example,

```
matlab cosim server="bj2lnx20" port=38525 inputs=["VI:wave" "VQ:wave"]
outputs=[TX_AFE_OUT_BB_I TX_AFE_OUT_BB_Q]
```

The parameters for the `cosim` statement are described below:

**Table 9-1 Parameters of the `cosim` Statement**

Parameter	Meaning
<code>design</code>	Refers to the file Simulink associates with the netlist. <code>design="env_d.mdl"</code> means the testbench whose file is <code>env_d.mdl</code> .
<code>inputs</code>	Input vector to identify the flow from MATLAB to SpectreRF Engine. The format is <code>["instance_name:wave" "instance_name1:wave" ...]</code> where <code>wave</code> , is a key word.  <b>Note:</b> The sequence of instances in square brackets follows the same one as the SpectreRF Engine port labels. This label information can be found by double-clicking the SpectreRF Engine and then by checking the <i>Show engine port labels</i> at the bottom in Function Block Parameters of the SpectreRF Engine.  After the inputs are set in the netlist, the sources associated with the inputs are meaningless and their parameters do not affect the simulation.
<code>outputs</code>	Output vector from SpectreRF Engine to MATLAB. This vector can be the net name or terminal name of interest. If net name is used, only voltage is given; if terminal name is used, only current is output.  To get current in an envlp analysis, you must add probes in the topology.
<code>port</code>	The port on the server running MATLAB. Its value should be the same as the value in the SpectreRF Engine in your current MATLAB design.

**Table 9-1 Parameters of the cosim Statement, *continued***

<code>server</code>	<p>The name of the machine that MATLAB starts. It can be set with the machine name or with the IP address of that machine. The accepted form is</p> <pre>server = "155.110.110.110"</pre> <p>and</p> <pre>server = "bj2lnx20"</pre> <p><code>server=localhost</code> means that Spectre and MATLAB use the same machine.</p>
<code>silent</code>	<p>Tells MATLAB whether to open the window during simulation. If <code>silent=yes</code>, the testbench window is opened.</p>
<code>timeout</code>	<p>The period of time MATLAB spends waiting for a response from the Spectre simulator.</p> <pre>timeout=60</pre> <p>stops MATLAB if it does not receive any response.</p>

**3. Close and save the netlist.**

After the above steps, the netlist is ready for simulation.

## Running the Cosimulation

With the coupler connected into the MATLAB design and with an appropriate netlist for the low-level design, you are ready to run the cosimulation. Note that these starting methods work only when the design is prepared as described earlier in this chapter.

<b>To use this starting method...</b>	<b>Follow the guidance in this section...</b>
Start the two applications (Spectre RF and MATLAB) separately.	<u><a href="#">"Starting the Two Applications Separately"</a></u> on page 686
Start Spectre RF and arrange to have MATLAB start automatically.	<u><a href="#">"Starting Spectre RF Manually and MATLAB Automatically"</a></u> on page 686
Start MATLAB and arrange to have Spectre RF start automatically.	<u><a href="#">"Starting MATLAB Manually and Spectre RF Automatically"</a></u> on page 687

## Starting the Two Applications Separately

1. Open the high-level design or testbench in the MATLAB design window.
2. In the MATLAB design window, click *Simulation — Start*.

The MATLAB desktop issues the following message.

```
block 'modified/SpectreRF Engine': (COSIM_OK) Waiting for incoming connection  
on port 38520, timeout: 60 sec ...
```

Then quickly do [step 3](#).

**Note:** The time interval between [step 2](#) and [step 3](#) must be within the *Simulation response timeout* defined as Function Block Parameters of the SpectreRF Engine (by double-clicking the SpectreRF Engine).

3. In the ADE window, click *Netlist and Run* or enter a spectre command at the command line. For example,

```
spectre tutorial1/tutorial1.scs
```

The cosimulation begins. The MATLAB desktop issues a message similar to the following when simulation ends.

```
block 'modified/SpectreRF Engine': (COSIM_OK) Simulation finished
```

## Starting Spectre RF Manually and MATLAB Automatically

To run the cosimulation by starting Spectre RF,

- If you are using ADE, open the low-level design in the ADE window.
  - a. Click *Netlist and Run*.

The cosimulation begins. The MATLAB desktop issues a message similar to the following when simulation ends.

```
block 'modified/SpectreRF Engine': (COSIM_OK) Simulation finished
```

- If you are using standalone Spectre RF,
  - a. At the command line, enter a command similar to the following.

```
spectre tutorial1/tutorial1.scs
```

The cosimulation begins.

## Starting MATLAB Manually and Spectre RF Automatically

- a. Start MATLAB.

```
matlab&
```

- b. Choose *Simulation* — *Start*.

The MATLAB desktop issues a message similar to the following.

```
block 'env_d/SpectreRF Engine': (COSIM_OK) Waiting for incoming connection  
on port 38525, timeout: 60 sec ...
```

```
block 'env_d/SpectreRF Engine': (COSIM_OK) Launch Spectre with commad  
'spectre tutorial2.scs =log spectre.out'
```

The cosimulation runs.

- c. After the cosimulation finishes, review the MATLAB/Simulink output, close the MATLAB desktop, and exit from MATLAB.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Cosimulation with MATLAB and Simulink

---



---

## Oscillator Noise Analysis

---

In RF systems, local oscillator phase noise can limit the final system performance. Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) lets you rigorously characterize the noise performance of oscillator elements. This appendix explains phase noise, tells how it occurs, and shows how to calculate phase noise using the Spectre RF simulator.

“Phase Noise Primer” on page 689 discusses how phase noise occurs and provides a simple illustrative example.

“Models for Phase Noise” on page 693 contains mathematical details about how the Spectre RF simulator calculates noise and how these calculations are related to other possible phase noise models. You can skip this section without any loss of continuity, but this section can help you better understand how Spectre RF calculates phase noise and better appreciate the drawbacks and pitfalls of other simple phase noise models. This section can also help in debugging difficult circuit simulations.

“Calculating Phase Noise” on page 703 provides some suggestions for successful and efficient analysis of oscillators and discusses the limitations of the simulator.

“Troubleshooting Phase Noise Calculations” on page 706 explains troubleshooting methods for difficult simulations.

“Frequently Asked Questions” on page 711 answers some commonly asked questions about phase noise and the Spectre RF simulator.

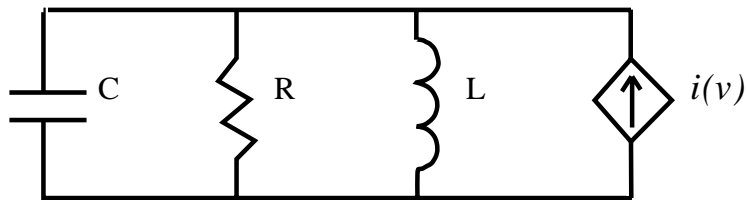
“Further Reading” on page 716 and “References” on page 716 list additional sources of information on oscillator noise analysis.

The procedures included in this appendix are intended for Spectre RF users who analyze oscillator noise. You must have a working familiarity with Spectre RF simulation and its operating principles. In particular, you must understand the Spectre RF PSS and Pnoise analyses. For information, see *Virtuoso Spectre Circuit Simulator RF Analysis Theory*.

Phase Noise Primer

Consider the simple resonant circuit with a feedback amplifier shown in Figure A-1, a parallel LC circuit with nonlinear transconductance. At small capacitor voltages, the transconductance is negative, and the amplifier is an active device that creates positive feedback to increase the voltage on the capacitor. At larger voltages, where the transconductance term goes into compression, the amplifier effectively acts as a positive resistor (with negative feedback) and limits the capacitor voltage.

**Figure A-1 A Simple Resonant Oscillator**



A simple model for the nonlinear transconductance is a cubic polynomial. We hypothesize a nonlinear resistor with a current-voltage relation given by

$$i(v) = -\left(\frac{v}{R}\right)(1 - \alpha v^2)$$

The effect of the resistor in parallel with the inductor and the capacitor can be lumped into this transconductance term. The parameter  $\alpha$  is a measure of the strength of the nonlinearity in the transconductance relative to the linear part of the total transconductance. Because the signal amplitude grows until the nonlinearity becomes significant, the value of this parameter does not affect the qualitative operation of the circuit.

For simplicity, for the remainder of this appendix

$$\alpha = 1/3$$

After some renormalization of variables, where time is scaled by

$$1/\omega_0$$

with

$$\omega_0 = \frac{1}{\sqrt{LC}}$$

and current is scaled by

$$\sqrt{C/L}$$

You can write the differential equations describing the oscillator in the following form

$$\frac{dv}{dt} = -i + \frac{1}{Q}(1 - \alpha v^2)v + \xi(t)$$

and

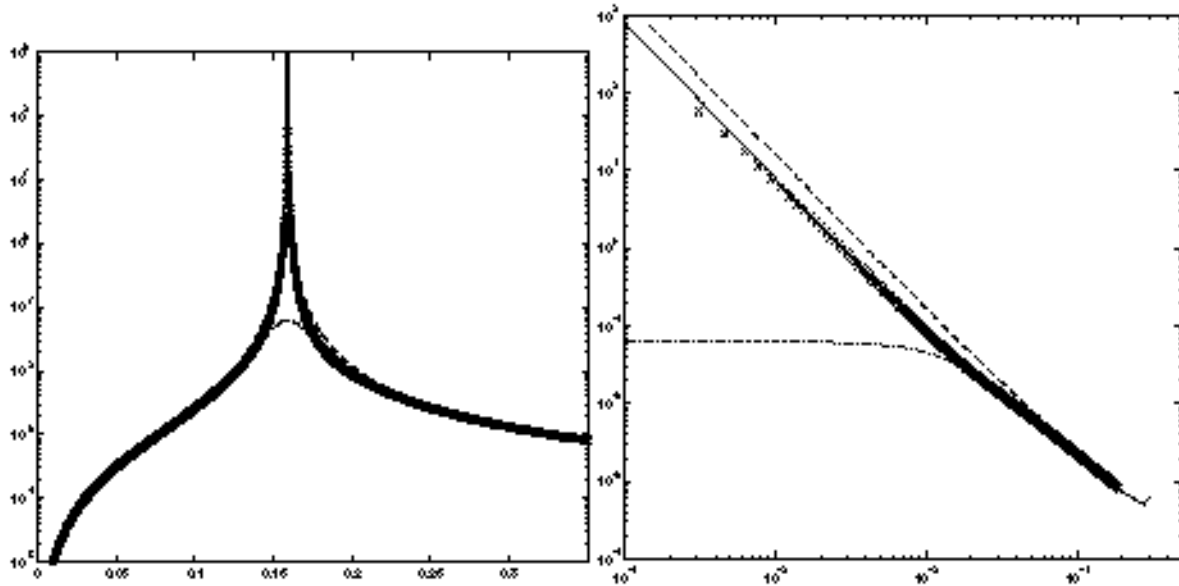
$$\frac{di}{dt} = v$$

In these equations,  $v$  and  $i$  represent the normalized capacitor voltage and inductor current, respectively, and  $\xi(t)$  is a small-signal excitation such as white Gaussian noise,  $Q = R/\omega_0 L$  is the quality factor of an RLC circuit made by replacing the nonlinear transconductance by a positive resistance  $R$ .

The equations just discussed describe the familiar Van der Pol oscillator system. This model includes many of the qualitative aspects of oscillator dynamics, yet it is simple enough to analyze in detail. Many more complicated oscillators that operate in a weakly nonlinear mode can be approximated with this model by using the first few terms in the Taylor series expansion of the relevant transconductances.

As a brute-force method of calculating the noise properties of this circuit, the nonlinear stochastic differential equations that describe the current and voltage processes were numerically integrated [1], and the noise power was obtained using a standard FFT-periodiagram technique. This technique requires several hundred simulations of the oscillator over many thousands of periods. Consequently, it is not a feasible approach for practical circuits, but it is rigorously correct in its statistical description even though it requires no knowledge of the properties of oscillators, noise, periodicity, or signal amplitudes. Figure [A-2](#) shows the total time-averaged noise in the voltage variable.

Figure A-2 Noise in a Simple Van der Pol System



By plotting *Power Spectral Density* against *Normalized Noise Frequency Offset* for a  $Q = 5$  system, Figure [A-2](#) shows noise in a simple Van der Pol system.

The left half of Figure [A-2](#) shows noise as a function of absolute frequency.

The right half of Figure [A-2](#) shows noise as a function of frequency offset from the oscillator fundamental frequency.

The dashed line is LC-filtered white noise, the dash-dot line is RLC-filtered white noise, the solid line is Spectre RF phase noise, and (x) marks are noise power from a full nonlinear stochastic differential equation solution.

The resulting noise power spectral density looks much like the voltage versus current response of a parallel LC circuit. The oscillator in steady-state, however, does not look like an LC circuit. As you see in the following paragraphs, this noise characteristic similarity occurs because both systems have an infinite number of steady-state solutions.

The characteristic shape of the small-signal response of an LC circuit results because an excitation at the precise resonant frequency can introduce a drift in the amplitude or phase of the oscillation. The magnitude of this drift grows with time and is potentially unbounded. In the frequency domain, this drift appears as a pole on the imaginary axis at the resonant frequency. The response is unbounded because no restoring force acts to return the

amplitude or phase of the oscillation to any previous value, and perturbations can therefore accumulate indefinitely.

Similarly, phase noise exists in a nonlinear oscillator because an autonomous oscillator has no time reference. A solution to the oscillator equations that is shifted in time is still a solution. Noise can induce a time shift in the solution, and this time shift looks like a phase change in the signal (hence the term *phase noise*). Because there is no *resistance* to change in phase, applying a constant white noise source to the signal causes the phase to become increasingly uncertain relative to the original phase. In the frequency domain, this corresponds to the increase of the noise power around the fundamental frequency.

If the noise perturbs the signal in a direction that does not correspond to a time shift, the nonlinear transconductance works to put the oscillator back on the original trajectory. This is similar to AM noise. The signal uncertainty created by the amplitude noise remains bounded and small because of the action of the nonlinear amplifier that created the oscillation. The LC circuit operates differently. It lacks both a time (or phase) reference and an amplitude reference and therefore can exhibit large AM noise.

Another explanation of the similarity between the oscillator and the LC circuit is that both are linear systems that have poles on the imaginary axis at the fundamental frequency,  $\omega_0$ . That is, at the complex frequencies  $s = i\omega_0$ . However, the associated transfer functions are not the same. In fact, because of the time-varying nature of the oscillator circuit, multiple transfer functions must be considered in the linear time-varying analysis.

Understanding the qualitative behavior of linear and nonlinear oscillators is the first step towards a complete understanding of oscillator noise behavior. Further understanding requires more quantitative comparisons that are presented in [Models for Phase Noise](#). If you are not interested in these mathematical details, you might skip ahead to [“Calculating Phase Noise”](#) on page 703.

## Models for Phase Noise

This section considers several possible models for noise in oscillators. In the engineering literature, the most widespread model for phase noise is the Leeson model [2]. This heuristic model is based on qualitative arguments about the nature of noise processes in oscillators. It shares some properties with the LC circuit models presented in the previous section. These models fit well with an intuitive understanding of oscillators as resonant RLC circuits with a feedback amplifier. In the simplest treatment, the amplifier is considered to be a negative conductance whose value is chosen to cancel any positive real impedance in the resonant tank circuit. The resulting linear time-invariant noise model is easy to analyze.

## Linear Time-Invariant (LTI) Models

To calculate the noise in a parallel RLC configuration, the noise of the resistor is modeled as a parallel current source of power density

$$S(\omega) = \frac{4k_B T}{R}$$

Where  $k_B$  is Boltzman's constant. In general, if current noise excites a linear time-invariant system, then the noise power density produced in a voltage variable is given by [3] as follows

$$S_v(\omega) = |H(\omega)|^2 S_i(\omega)$$

where  $H(\omega)$  is the transfer function of the LTI transformation from the noise current source *input* to the voltage *output*. The transfer function is defined in the standard way to be

$$H(\omega) = \frac{v_0(\omega)}{i_s(\omega)}$$

where  $i_s$  is a (deterministic) current source and  $v_0$  is the measured voltage between the nodes of interest.

It follows that the noise power spectral density of the capacitor voltage in the RLC circuit is, as a function of frequency  $\omega = \omega_0 + \omega'$  with  $\omega' \ll \omega_0$ .

$$\frac{4k_B T}{R} \frac{1}{1 + 4(\omega'/\omega_0)^2 Q^2}$$

where the quality factor of the circuit is

$$Q = \frac{R}{\omega_0 L}$$

The parallel resistance is  $R$  (the source of the thermal noise), and  $\omega_0$  is the resonant frequency.

If a noiseless negative conductance is added to precisely cancel the resistor loss, the noise power for small  $\omega' / \omega_0$  becomes

$$S_v(\omega') = \frac{k_B TR}{(\omega'/\omega_0)^2 Q^2}$$

This linear time-invariant viewpoint explains some qualitative aspects of phase noise, especially the  $(\omega_0 / Q\omega')^2$  dependencies. However, even for this simple system, a set of complicating arguments is needed to extract approximately correct noise from the LTI model. In particular, we must explain the 3 dB of excess amplitude noise inside the resonant bandwidth generated by an LC model but not by an oscillator (see [“Amplitude Noise and Phase Noise in the Linear Model”](#) on page 699). Furthermore, many oscillators, such as relaxation and ring oscillators, do not naturally fit this linear time-invariant model. Most oscillators are better described as time-varying (LTV) circuits because many phenomena, such as upconversion of  $1/f$  noise, can only be explained by time-varying models.

## Linear Time-Varying (LTV) Models

For linear time-invariant systems, the noise at a frequency  $\omega$  is directly due to noise sources at that frequency. The relative amplitudes of the noise at the system outputs and the source noise are given by the transfer functions from noise sources to the observation point. Time-varying systems exhibit frequency conversion, however, and each harmonic  $k\omega_0$  in the oscillation can transfer noise from a frequency  $\omega \pm k\omega_0$  to the observation frequency  $\omega$ . In general, for a stationary noise source  $\xi(t)$ , the total observed noise voltage is [3]

$$S_v(\omega) = \sum_k |H_k(\omega)|^2 S_\xi(\omega + k\omega_0)$$

Each term in the series represents conversion of current power density at frequency  $\omega + k\omega_0$  to voltage power density at frequency  $\omega$  with gain  $|H_k(\omega)|^2$ . As an example, return again to the Van der Pol oscillator with  $\alpha = 1/3$  and notice how a simple time-varying linear analysis of noise proceeds.

The first analysis step for the Van der Pol oscillator is to obtain a large-signal solution, so you set  $\xi(t) = 0$ . In the large-Q limit, the oscillation is nearly sinusoidal and so it is a good approximation to assume the following

$$v(t) = a \sin \omega_0 t$$

The amplitude,  $a$ , and oscillation frequency can be determined from the differential equations that describe the oscillator. Recognizing that

$$i(t) = \left(\frac{a}{\omega_0}\right)\cos(\omega_0 t)$$

and substituting into the equation for  $dv/dt$ ,  $a$  and  $\omega_0$  are determined by the following

$$a\omega_0\cos(\omega_0 t) - \left(\frac{1}{Q}\right)\left(a\sin(\omega_0 t) - \frac{a^3}{3}\sin^3(\omega_0 t)\right) - \left(\frac{a}{\omega_0}\right)\cos(\omega_0 t) = 0$$

Substituting

$$\sin^3(\omega_0 t) = \frac{3\sin(\omega_0 t) - \sin(3\omega_0 t)}{4}$$

and using the orthogonality of the sine and cosine functions, it follows that

$$a - \left(\frac{a^3}{4}\right) = 0$$

and

$$\omega_0 - \left(\frac{1}{\omega_0}\right) = 0$$

(The  $\sin(3\omega_0 t)$  term is relevant only when we consider higher-order harmonics of the oscillation.) Therefore, to the lowest order of approximation,  $a = 2$  and  $\omega_0 = 1$ .

The only nonlinear term in the Van der Pol equations is the current-voltage term,  $v^3/3$ . This term differentiates the Van der Pol oscillator from the LC circuit. The small-signal conductance is the derivative with respect to voltage of the nonlinear current

$$\frac{-(1-v^2)}{Q}$$



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

With

$$v(t) = 2 \sin t$$

the small-signal conductance as a function of time is

$$(1/Q)(1 - \cos 2t)$$

Because there is a nonzero, time-varying, small-signal conductance, the Periodic Time Varying Linear (PTVL) model is different from the LTI LC circuit model. In fact, the time-average conductance is not even zero. However, the time-average power dissipated by the nonlinear current source is zero, a necessary condition for stable, sustained oscillation.

Oscillators are intrinsically time-varying elements because they trade off excessive gain during the low-amplitude part of the cycle with compressive effects during the remainder of the cycle. This effect is therefore a generic property not unique to this example.

To complete the noise analysis, write the differential equations that the small-signal solution  $i_{s(t)}$ ,  $v_{s(t)}$  must satisfy,

$$\frac{dv_s}{dt} = -i_s + \frac{1}{Q}(1 - 3\alpha v^2(t))v_s + \xi(t)$$

and

$$\frac{di_s}{dt} = v_s$$

From the large signal analysis,  $v(t) = 2 \sin t$ , and so

$$\frac{dv_s}{dt} = -i_s + \frac{1}{Q}(2 \cos 2t - 1)v_s + \xi(t)$$

and

$$\frac{di_s}{dt} = v_s$$

The time-varying conductance can mix voltages from a frequency  $\omega$  to  $\omega - 2$ . For small  $\omega'$ , if an excitation is applied at a frequency  $\omega = 1 + \omega'$ ,  $i_s$  and  $v_s$  are expected to have components at  $1 + \omega'$  and  $-1 + \omega'$  for the equations to balance. (Higher-order terms are again presumed to be small.) Writing

$$i_s(t) = i_+ e^{i(1+\omega')t} + i_- e^{i(-1+\omega')t}$$

and substituting into the small-signal equations with

$$\xi(t) = c_+ e^{i(1+\omega')t}$$

leads to the following system of equations for  $i_+$  and  $i_-$

$$\begin{bmatrix} 1 - (1 + \omega')^2 + \left(\frac{i}{Q}\right)(1 + \omega') & -\left(\frac{i}{Q}\right)(-1 + \omega') \\ -\left(\frac{i}{Q}\right)(1 + \omega') & 1 - (-1 + \omega')^2 + \left(\frac{i}{Q}\right)(-1 + \omega') \end{bmatrix} \begin{bmatrix} i_+ \\ i_- \end{bmatrix} = \begin{bmatrix} c_+ \\ 0 \end{bmatrix}$$

Solving these equations gives the transfer function from an excitation at frequency  $1 + \omega'$  to the small-signal at frequency  $1 + \omega'$  that we call  $H_0(\omega')$ . A similar analysis gives the other significant transfer function, from noise at frequency  $-1 + \omega'$  of amplitude  $C_-$  to the small-signal response at frequency  $1 + \omega'$ , that we call  $H_{-2}(\omega')$ . In the present case, for small  $\omega'$ ,

$$H_0^2 \cong H_{-2}^2 \cong \frac{R^2}{16Q^2 \left(\frac{\omega'}{\omega_0}\right)^2}$$

For a general Van der Pol circuit with a parallel resistor  $R$  that generates white current noise,  $\xi(t)$ , with  $S_\xi(\omega) = 4 k_B T/R$ ,

$$S_v(\omega') = \frac{k_B TR}{2\left(\frac{\omega'}{\omega}\right)^2 Q^2}$$

Note that this is precisely one-half the noise predicted by the LC model.

You can gain additional insight about phase noise by analyzing the time-domain small-signal response. The small-signal current response is.

$$i_s(t) = \frac{c_+ e^{j\omega' t} + c_- e^{-j\omega' t}}{2\omega'} \sin t$$

Notice that  $c_+$  and  $c_-$  are complex random variables that represent the relative contribution of white noise at separate frequencies. As white noise has no frequency correlations, they have uncorrelated random phase, and thus zero amplitude expectation, and unit variance in amplitude. Because the large-signal current is  $i(t) = 2\cos t$ , and the sine and cosine functions are orthogonal, the total noise for small  $\omega'$  that we computed is essentially all phase noise.

## Amplitude Noise and Phase Noise in the Linear Model

Occasional claims are made that in oscillators, “Half the noise is phase noise and half the noise is amplitude noise.” However, as the simple time-varying analysis in the previous section shows, in a physical oscillator the noise process is mostly phase noise for frequencies near the fundamental. It is true that in an LC-circuit half the total noise power corresponds to AM-like modulation and the other half to phase modulation. In the literature, the AM part of the noise is sometimes disregarded when quoting the oscillator noise although this is not always the case. (The Spectre RF simulator computes the total noise generated by the circuit; see “[Details of the Spectre RF Calculation](#)” on page 700).

However, a *linear* oscillator does not really exist. Physical oscillators operate with a tradeoff of gain that causes growing signal strength and nonlinear compressive effects that act to limit the signal amplitude. For noise calculation, the oscillator cannot be considered a linear time-invariant system because there are intrinsic nonlinear effects that produce large phase noise but limited amplitude noise. Oscillators are time-varying, and they therefore require a time-varying small-signal analysis.

Arguments that start with stationary white noise and pass it through a linear model in a forward-analysis fashion produce incorrect answers. This is true because they neglect the time-variation of the conductances (and possibly the capacitances) in the circuit. In the simple cases considered here, the conductances vary in time in a special way so as to produce no amplitude noise, only phase noise.

They have that special variation because they result from linearization about an oscillator limit cycle. An oscillator in a limit cycle has a large response to phase perturbations, but not to amplitude perturbations. The amplitude perturbations are limited by the properties of the nonlinear amplifier, but the phase perturbations can persist. The Spectre RF simulator calculates the correct phase noise because it *knows* about the oscillator properties.

Similarly, arguments [13] that start with noise power and derive phase noise in a backwards fashion also usually produce incorrect results because they cannot correctly account for frequency correlations in the noise of the oscillator. These frequency correlations are introduced by the time-varying nature of the circuit.

Occasionally, a netlist appears in which a negative resistance precisely cancels a positive resistance to create a pure LC circuit. Because such a circuit has an infinite number of oscillation modes, the Spectre RF simulator cannot correctly calculate the noise because it assumes a unique oscillation. Such a circuit is not physically realizable because adding or subtracting a microscopically small amount of conductance makes the circuit either go into nonlinear operation (amplifier saturation) or become a damped LC circuit that has a unique final equilibrium point. This equilibrium point is the zero-state solution. Trying to create the negative resistance oscillator is like trying to bias a circuit on a metastable point. Any amplitude oscillation can exist, depending on the initial conditions, as long as the amplitude is less than the amplifier saturation point.

## Details of the Spectre RF Calculation

This section contains the mathematical details of how the Spectre RF simulator computes noise in oscillators. Understanding the material in this section can help you troubleshoot and understand difficult oscillator problems.

The analysis the Spectre RF simulator performs is similar to the simple analysis in the section [“Linear Time-Varying \(LTV\) Models”](#) on page 695. During analysis, the Spectre RF simulator

1. First finds the periodic steady state of the oscillator using the PSS analysis.
2. Then linearizes around this trajectory.
3. The resulting time-varying linear system is used to calculate the noise power density. The primary difference between the Spectre RF calculation and the previous analysis is that the basis functions used for the Spectre RF calculation are not just a few sinusoids, but rather a collection of many piecewise polynomials. The use of piecewise polynomials allows the Spectre RF simulator to solve circuits with arbitrary waveforms, including circuits with highly nonlinear behavior.

Noise computations are usually performed with a small-signal assumption, but a rigorous small-signal characterization of phase noise is complicated because the variance in the

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

phase of the oscillation grows unbounded over time. From a mathematical viewpoint, an oscillator is an autonomous system of differential equations with a stable limit cycle. An oscillator has phase noise because it is neutrally stable with respect to noise perturbations that move the oscillator in the direction of the limit cycle. Such *phase* perturbations persist with time, whereas transverse fluctuations are damped with a characteristic time inversely proportional to the quality factor of the oscillator.

Further care is necessary because, in general, the two types of excitations (those that create phase slippage and those responsible for time-damped fluctuations) are not strictly those that are parallel or perpendicular, respectively, to the oscillator trajectory, as is sometimes claimed (for example, in [4]).

However, one must realize that the noise powers at frequencies near the fundamental frequency correspond to correlations between points that are widely separated on the oscillator envelope. In other words, they are long-time signal effects. In fact, asymptotically (at long times), the ratio of the variance of any state variable to its power at the fundamental frequency is unity for any magnitude of the noise excitation. Therefore, in practical cases, you can consider only small deviations in the state variables when describing the phase noise.

The first step in the noise analysis is to determine the oscillator steady-state solution. This is done in the time domain using shooting methods [5]. Once the periodic steady-state is obtained, the circuit equations are linearized around that waveform in order to perform the small-signal analysis.

The time-varying linear system describing the small-signal response  $v_s(t)$  of the oscillator to a signal  $w(t)$  can be written in general form as [6, 7]

$$\left[ C(t) \frac{d}{dt} + G(t) \right] v_s \equiv L(t) v_s(t) = w(t)$$

where  $C(t)$  and  $G(t)$  represent the linear, small-signal, time-varying capacitance and conductance matrixes, respectively. These matrixes are obtained by linearization about the periodic steady-state solution (the limit cycle). To understand the nature of time-varying linear analysis, the concept of Floquet multipliers is introduced.

Suppose  $x(t)$  is a solution to the oscillator circuit equations that is periodic with period  $T$ . If  $x(0)$  is a point on the periodic solution  $x_L(t)$ , then  $x(T) = x(0)$ . If  $x(0)$  is perturbed slightly off the periodic trajectory,  $x(0) = x_L(0) + \delta x$ , then  $x(T)$  is also perturbed, and in general for small  $\delta x$ ,

$$x(T) - x_L(T) \approx \frac{\partial x(T)}{\partial x(0)} \delta x$$

The Jacobian matrix

$$\frac{\partial x(T)}{\partial x(0)}$$

is called the sensitivity matrix. The Spectre RF simulator uses an implicit representation of this matrix both in the shooting method that calculates the steady-state and in the small-signal analyses. To see how the sensitivity matrix relates to oscillator noise analysis, consider the effect of a perturbation at time  $t = 0$  several periods later, at  $t = nT$ . From the above equation,

$$x(nT) - x_L(nT) \approx \left[ \frac{\partial x(T)}{\partial x(0)} \right]^n \delta x$$

so

$$x(nT) - x_L(nT) \approx \sum_i C_i \lambda_i^n \phi_i$$

where  $\phi_i$  is an eigenvector of the sensitivity matrix. The  $C_i$  are the expansion coefficients of  $\delta x$  in the basis of  $\phi_i$ . If  $\psi_i$  is a left eigenvector (an eigenvector of its transpose) of the sensitivity matrix, then

$$C_i = \psi_i^T \delta x$$

Let  $\lambda$  be an eigenvalue of the sensitivity matrix. In the context of linear time-varying systems, the eigenvalues  $\lambda$  are called *Floquet multipliers*. If all the  $\lambda$  have magnitude less than one (corresponding to left-half-plane poles), the perturbation decays with time and the periodic trajectory is stable. If any  $\lambda$  has a magnitude greater than one, the oscillation cannot be linearly stable because small perturbations soon force the system away from the periodic trajectory  $x_L(t)$ .

A stable nonlinear physical oscillator, however, must be neutrally stable with respect to perturbations that move it in the direction of the orbit. These are not necessarily perturbations in the direction of the orbit because, in general,

$$\Psi \neq \phi_1$$

This is true because a time-shifted version of the oscillator periodic trajectory still satisfies the oscillator equations. In other words, one of the Floquet multipliers must be equal to unity. This Floquet multiplier is responsible for phase noise in the oscillator. The associated eigenvector determines the nature of the noise.

If  $\lambda = e^{\eta}$  is a Floquet multiplier, then  $\eta + ik\omega_0$  is a pole of the time-varying linear system for any integer  $k$ . Therefore, because of the unity Floquet multiplier, the time-varying linear system has poles on the imaginary axis at  $k\omega_0$ . This is very similar to what occurs in a pure LC resonator, and it explains the identical shape of the noise profiles.

Because operator  $L(t)$  has poles at the harmonics of the oscillation frequency, numerical calculations of the noise at nearby frequencies become inaccurate if treated in a naive manner [8, 9]. To correctly account for the phase noise, the Spectre RF simulator finds and extracts the eigenvector that corresponds to the unity Floquet multiplier. To correctly extract the phase noise component, both the right and left eigenvectors are required. Once these vectors are obtained, the singular (phase noise) contribution to the noise can be extracted. The remaining part of the noise can be obtained using the usual iterative solution techniques [6] in a numerically well-conditioned operation.

In [Figure A-2](#) on page 692, you can see that the Spectre RF PTVL analysis correctly predicts the total noise, including the onset of 3 dB amplitude noise outside the bandwidth of the resonator. Note that this simulation was conducted at

$$E\left\{\xi^2(t)\right\} = 10^{-3}$$

which represents a very high noise level that is several orders of magnitude higher than in actual circuits. The good match of the PTVL models to the full nonlinear simulation shows the validity of the PTVL approximation.

## Calculating Phase Noise

The following sections suggest simulation parameters, give you tips for using these parameters, and advise you about checking for accuracy.

## Setting Simulator Options

The Spectre RF time-varying small-signal analyses are more powerful than the standard large-signal analyses (DC, TRAN) but, like any precision instrument, they also have greater sensitivity to numerical errors. For many circuits, particularly oscillators, more simulator precision is needed to get good results from the PAC, PXF, and Pnoise calculations than is needed to get good DC or TRAN results.

The small-signal analyses operate by linearizing around the periodic steady state solution. Consequently, the oscillator noise analysis, and the periodic small-signal analyses in general, inherit most of their accuracy properties from the previous PSS simulation. You must be sure the PSS simulation generates a sufficiently accurate linearization. See [“What Can Go Wrong”](#) on page 707 for a discussion.

Table [A-1](#) recommends simulator options for various classes of circuits.

**Table A-1 Recommended Spectre RF Parameter Values**

Circuit	errpreset	reltol	vabstol	iabstol
Easy	moderate	1.0e <sup>-4</sup>	default	default
Hard-I	conservative	1.0e <sup>-5</sup>	10n	1p
Hard-II		1.0e <sup>-6</sup>	1n	0.1p
Hard-III		1.0e <sup>-7</sup>	0.1n	0.1p

- *Easy* circuits are low- $Q$  (about  $Q < 10$ ) resonant oscillators, ring oscillators, and weakly-nonlinear relaxation oscillators. Most textbook circuits are in this category. This is the default setting when you set `errpreset=moderate`.
- *Hard-I* circuits are most high- $Q$  resonant oscillators; circuits with complicated AGC, load, or bias circuitry; and relaxation or ring oscillators that exhibit moderate to strong nonlinear or *stiff* effects. This is the best general-purpose set of options and it is the default when you set `errpreset=conservative`.
- *Hard-II* and *Hard-III* circuits include a few particularly difficult circuits.
- Usually these options are used only in a convergence study or for circuits that previously failed a convergence study using less strict options.
- Circuits in this category often exhibit some form of unusual behavior (see [“What Can Go Wrong”](#) on page 707).



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

- Sometimes this behavior results from circuit properties, for example, some very high- $Q$  crystal oscillators and some very stiff relaxation oscillator circuits. Occasionally, the behavior reflects a design flaw.

Usually setting `method=gear2only` is recommended for the PSS simulation (but see “[What Can Go Wrong](#)” on page 707).

The parameters in Table A-1 are used in error control at two places.

At the local truncation error (LTE) at each transients integration. The LTE control formula is

$$v_n(t) - v_{n, pred}(t) < lteratio \times [reltol \times v_{n, max} + vabstol]$$

- To control the periodicity error over the period. The periodicity error control formula is

$$v_n(0) - v_n(T) < lteratio \times steadyratio \times [reltol \times v_{n, max} + vabstol]$$

The default value for `vabstol` is  $1.0e^{-6}$ , The default value for `iabstol` is  $1.0e^{-12}$  which is accurate enough for most cases. Tighten `iabstol` when necessary. Refer to section 4.3 in [17] for detailed discussion about transient integration.

In order to speed up transients integration during the `tstab` stage, the default values at the `tstab` stage are `reltol=1.0e-3`, `lteratio=3.5`, `relref=sigglobal`, `maxstep=T/25`, `method=traonly`. After the `tstab` stage, those parameter are set back according to `errpreset`.

For circuits, such as extremely high  $Q$  oscillators, that need very high accuracy, you can gain further accuracy by turning on the highorder refinement `highorder=yes` and `errpreset=conservative`. This runs multi-interval Chebyshev PSS refinement after the shooting phase of the PSS analysis.

An effective and fast method to start the oscillator is to run the new autonomous envelope analysis and to save the simulation results. Use the results of the autonomous envelope analysis as the initial condition for the PSS analysis.

A longer `tstab` stage helps with PSS convergence. However a longer `tstab` stage can slow the simulation. Using autonomous envelope analysis to establish `tstab` is considerably faster than using the transient analysis for `tstab`. See [Virtuoso Spectre Circuit Simulator RF Analysis Theory](#) for information on the autonomous envelope analysis.

For releases before MMSIM60 USR1, an effective method to kick the oscillation is to add a kicker to the circuit. The kicker can be either a voltage or current source. To effectively start

the oscillation, the kicker has to be placed at the most sensitive place which usually is close to the oscillating transistor. The kicker can be either a PWL source or a damped sinusoidal source with frequency set to the oscillation frequency. The kicker must die down and remain stable after oscillation is established to avoid affecting the PSS analysis.

## Troubleshooting Phase Noise Calculations

The Spectre RF simulator calculates noise effectively for most oscillators. However, circuits that are very stiff, very nonlinear, or just poorly designed can occasionally cause problems for the simulator. Stiff circuits exhibit dynamics with two or more very different time scales; for example, a relaxation oscillator with a square-wave-like periodic oscillation. Over most of the cycle, the voltages change very slowly, but occasional rapid transitions are present. This section describes some of the reasons for the problems, what goes wrong, how to identify problems, and how to fix them.

See [“Details of the Spectre RF Calculation”](#) on page 700 for help troubleshooting particularly difficult circuits.

### Known Limitations of the Simulator

Any circuit that does not have a stable periodic steady-state cannot be analyzed by the Spectre RF simulator because oscillator noise analysis is performed by linearizing around a waveform that is assumed to be strictly periodic.

For example, oscillators based on IMPATT diodes generate strong subharmonic responses and cannot be properly analyzed with the Spectre RF simulator. As another example, Colpitts oscillators, properly constructed, can be made to exhibit chaotic as well as subharmonic behavior.

Similarly, any circuit with significant large-signal response at tones other than the fundamental and its harmonics might create problems for the simulator. Some types of varactor-diode circuits might fit this category. In addition, some types of AGC circuitry and, on occasion, bias circuitry can create these effects.

The Spectre RF simulator cannot simulate these circuits because simulation of an autonomous circuit with subharmonic or other aperiodic components in the large signal response essentially requires foreknowledge of which frequency components are important. Such foreknowledge requires Fourier analysis of very long transient simulations and cannot be easily automated. Such simulations can be very expensive.

## What Can Go Wrong

The Spectre RF simulator can have problems in the following situations.

### Generic PSS Simulation Problems

Any difficulties in the underlying PSS analysis affect the phase noise computation. For example, underestimating the oscillator period or failing to start the oscillator properly can cause PSS convergence problems that make running a subsequent Pnoise analysis impossible.

### Hypersensitive Circuits

Occasionally, you might see circuits that are extremely sensitive to small parameter changes. Such a circuit was a varactor-tuned VCO that had the varactor bias current, and therefore the oscillation frequency, set by a 1 T $\Omega$  resistor. Changing to a 2 T $\Omega$  resistor, which is a  $1e^{-12}$  relative perturbation in the circuit matrixes, changed the oscillation frequency from 125 MHz to 101 Mhz. Such extreme circuit sensitivity results in very imprecise PSS simulations. In particular, the calculated periods have relatively large variations. If precise PSS simulations are impossible, precise noise calculations are also impossible. In such a case, you must fix the circuit.

### Subharmonics or Parametric Oscillator Modulation

Sometimes bias and AGC circuitry might create small-amplitude parasitic oscillations in the large signal waveform. You can identify these oscillations by performing a transient simulation to steady-state and then looking for modulation of the envelope of the oscillation waveform. For high- $Q$  circuits and/or low-frequency parasitics, this transient simulation might be very long.

In this case, because the oscillator waveform is not actually periodic, the PSS simulation can only converge to within approximately the amplitude of the parasitic oscillation. If the waveform possesses a parasitic oscillation that changes amplitude, over one period, around  $10^{-5}$  relative to the oscillator envelope, then convergence with `reltol < 10-5` is probably not possible (assuming `steadyratio` is one or less).

These effects might also appear as a parametric sideband amplification phenomenon.

See [“Frequently Asked Questions”](#) on page 711 for more information.

### **Small-Signal Frequency is Much Higher than the Fundamental Frequency**

The same timesteps are used for both the small-signal analysis and the PSS analysis. If the small-signal frequency is much higher than the fundamental frequency, much smaller timesteps might be required to accurately resolve the small-signal than are needed for the large signal. To force the Spectre RF simulator to take sufficiently small timesteps in the PSS simulation, be sure the `maxacfreq` parameter is set correctly.

### **Wide Timestep Variation**

Occasionally, in simulations that generate PSS waveforms with timesteps that vary over several orders of magnitude, the linear systems of equations that determine the small-signal response become ill-conditioned. As a result, the noise analysis is inaccurate. Usually this occurs because you have requested excessive simulator precision; for example, nine-digit precision. You can sometimes eliminate this problem using `method = traponly` in the PSS solution. You might also set `maxstep` to a very small value in the PSS analysis or you might specify a very large `maxacfreq` value.

### **Problems with Device Models**

When the device models leave their physically meaningful operating range during the large-signal PSS solution, the noise calculations are usually inaccurate. Similarly, when the models are discontinuous, or have discontinuous derivatives, the small-signal analysis might be inaccurate.

### **Problems Resolving Floquet Multipliers in Stiff Relaxation Oscillators**

Sometimes in very stiff relaxation oscillators, the PSS solution rapidly and easily converges; but the numerically calculated Floquet multiplier associated with the PSS solution is far from unity. Typically, this multiplier is real and has a magnitude much larger than unity. The Spectre RF simulator prints a warning (see [“Message III”](#) on page 710). It is interesting that sometimes the phase noise is quite accurate even with low simulation tolerances. If you have this problem, perform a convergence study.

### **Problems Resolving Floquet Multipliers in High-Q Resonant Circuits**

In a physical oscillator, there is one Floquet multiplier equal to unity. In an infinite- $Q$  linear resonator, however, the multipliers occur in complex conjugate pairs. A very high- $Q$  nonlinear oscillator has another Floquet multiplier on the real axis nearly equal to, but slightly less than, one. In the presence of numerical error, however, these two real Floquet multipliers can appear to the simulator as a complex-conjugate pair. The phase noise is computed using the

Floquet vector associated with the unity Floquet multiplier. When the two multipliers appear as a complex pair, the relevant vector is undefined. When the Spectre RF simulator correctly identifies this situation, it prints a warning (see [“Message III”](#) on page 710). The solution is usually to simulate using the next higher accuracy step (see [Table A-1](#) on page 704). Sometimes varying `tstab` can also help with this problem.

If the circuit is really an infinite- $Q$  resonator (for example, a pure parallel LC circuit), the multipliers always appear as complex conjugate pairs and the noise computations are not accurate close to the fundamental frequency. Such circuits are not physical oscillators, and the Spectre RF simulator is not designed to deal with them; see [“Amplitude Noise and Phase Noise in the Linear Model”](#) on page 699 and [“Frequently Asked Questions”](#) on page 711.

## Phase Noise Error Messages

Spectre RF displays error messages when it encounters several types of known numerical difficulty. To interpret the error messages produced by the phase noise analysis, you must know the material in [“Details of the Spectre RF Calculation”](#) on page 700.

### Message I

**The Floquet eigenspace computed by spectre PSS analysis appears to be inaccurate. PNOISE computations may be inaccurate. Consider re-running the simulation with smaller reltol and method=gear2only.**

The eigenvector responsible for phase noise was inaccurately computed and the PSS simulation tolerances might be too loose. Try simulating the circuit at the next higher accuracy setting (see [Table A-1](#) on page 704) and then compare the calculated noise in the two simulations.

### Message II

**The Floquet eigenspace computed by spectre PSS analysis appears to be ill-defined. PNOISE computations may be inaccurate. Consider re-running the simulation with smaller reltol, different tstab(s), and method=gear2only. Check the circuit for unusual components.**

This can be an accuracy problem, or it can result from an unusual circuit topology or sensitivity. Tighten the accuracy requirements as much as possible (see [Table A-1](#) on page 704). If this message appears in all simulations, the noise might be incorrect even if the simulations agree.

### Message III

**The Floquet eigenspace computed by spectre PSS analysis appears to be inaccurate and/or the oscillator possesses more than one stable mode of oscillation. PNOISE computations may be inaccurate. Consider re-running the simulation with smaller reltol, different tstab(s), and method=gear2only.**

All the real Floquet multipliers were well-separated from unity, suggesting that the PSS simulation tolerances might be too loose. Simulate the circuit at the next higher accuracy setting (see [Table A-1](#) on page 704) and then compare the calculated noise in the two simulations. If the calculated noise does not change, it is probably correct even if this message appears in both simulations.

### The tstab Parameter

Because Spectre RF performs the PSS calculation in the time domain by using a *shooting method*, an infinite number of possible PSS solutions exist, depending on where the first timepoint of the PSS solution is placed relative to the oscillator phase.

The placement of the first timepoint is determined by the length of the initial transient simulation, which you can control using the `tstab` parameter. If the `tstab` value causes the edges of the periodic window to fall on a point where the periodic oscillator waveform is making very rapid transitions, it is very difficult for PSS to converge. Similarly, the results of the small-signal analyses are probably not very accurate. Avoid such situations. If the start of the PSS waveform falls on a very fast signal transition, you usually need to view the results of further small-signal analyses with some skepticism.

Although a poor choice of the `tstab` parameter value can degrade convergence and accuracy, appropriate use of `tstab` can help to identify problem circuits and to estimate the reliability of their noise computations.

If you perform several PSS and Pnoise computations that differ only in their `tstab` parameter values, the results should be fairly similar, within a relative deviation of the same order of magnitude as the simulator parameter `reltol`. If this is not the case, you might not have set the simulator accuracy parameters sufficiently tight to achieve an accurate solution; and you need to reset one or more of the parameters `reltol`, `vabstol`, or `iabstol`. The circuit might also be poorly designed and very sensitive to perturbations in its parameters.

If the calculated fundamental period of the oscillator varies with `tstab` even when you set `reltol`, `iabstol`, and `vabstol` to very small (but not vanishingly small) values, the circuit is probably poorly designed, exhibiting anomalous behavior, or both. (see [“Known Limitations of the Simulator”](#) on page 706).

## Frequently Asked Questions

The following questions are similar to those commonly asked about oscillator noise analysis with the Spectre RF simulator.

### **Does Spectre RF simulation calculate phase noise, amplitude noise, or both?**

Spectre RF simulation computes the total noise of the circuit, both amplitude and phase noise. What the analog circuit design environment plots as *phase noise* is really the total noise scaled by the power in the fundamental oscillation mode. Close enough to the fundamental frequency, the noise is all phase noise, so what the analog circuit design environment plots of *phase noise* is really the phase noise as long as it is a good ways above the noise floor.

Some discussions of oscillator noise based on a simple resonator/amplifier description describe the total noise, at small frequency offsets from the fundamental, as being half amplitude noise and half phase noise. In reality, for physical oscillators, near the fundamental nearly all the noise is phase noise. Therefore, these simple models overestimate the total noise by 3 dB. For a detailed explanation, see the phase noise theory described in [“Details of the Spectre RF Calculation”](#) on page 700 and the detailed discussion of the Van der Pol oscillator [“Linear Time-Varying \(LTV\) Models”](#) on page 695.

### **I have a circuit that contains an oscillator. Can I simulate the oscillator separately and use the phase noise Spectre RF calculates as input for a second PSS/PNOISE simulation?**

No. Oscillators generate noise with correlated spectral sidebands. Currently, Spectre RF simulation output represents only the time-average noise power, not the correlation information, so the noise cannot be input to a simulation that contains time-varying elements that might mix together noise from separate frequencies.

If the second circuit is a linear filter (purely lumped linear time-invariant elements, such as resistors, capacitors, inductors, or a linearization of a nonlinear circuit around a DC operating point) that generates no frequency mixing, then you can use the output of the Spectre RF Pnoise analysis as a *noisefile* for a subsequent NOISE (not Pnoise) analysis.

### **How accurate are the phase noise calculations? What affects the errors?**

Initially, it is important to distinguish between modeling error and simulation (numerical) error. If the device models are only good to 10% the simulation is only good to 10% (or worse). So, for the rest of this appendix, we discuss numerical error introduced by the approximations in the algorithms.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

You must also distinguish between absolute and relative signal frequencies in the noise analysis. When the noise frequency is plotted on an absolute scale, the error is primarily a function of the variance in the calculated fundamental period. This is true because of the singular behavior, in these regions, of the phase noise near a harmonic of the fundamental. To see this behavior, note that for the simple oscillator driven by white noise, the noise power is proportional to the offset from the fundamental frequency,

$$S_v(\omega) \propto \frac{1}{(\omega - \omega_0)^2}$$

If you make a small error in the calculation of  $\omega_0$ , the error  $\Delta S_v$  in the noise is proportional to  $S / \omega_0$

$$\Delta S_v(\omega) \propto \frac{\Delta \omega_0}{(\omega - \omega_0)^3}$$

This error can be very large even if  $\Delta \omega_0$ , the error in  $\omega_0$ , is small. However, because of the way Spectre RF simulation extracts out the phase noise, the calculated phase noise, as a function of offset from the fundamental frequency, can be quite accurate even for very small offsets.

Now consider how much error is present in the calculated fundamental frequency. Because the numerical error is related to many simulation variables, it is difficult to quantify, without examination, how much is present. However, as a rough approximation, if we define the quantity

$$r = \min \left\{ reltol, \frac{iabstol}{\max(i)}, \frac{vabstol}{\max(v)} \right\}$$

where  $\max(i)$  and  $\max(v)$  are the maximum values of current and voltage over the PSS period, then, under some assumptions,  $\Delta \omega_0$ , the error in the fundamental  $\omega_0$ , probably satisfies

$$r\omega_0 < \delta\omega_0 < Mr\omega_0$$



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

where  $M$  is the number of timesteps taken for the PSS solution. This analysis assumes that `steadyratio` is sufficiently tight, not much more than one, and also that `iabstol` and `vabstol` are sufficiently small.

If you require a good estimate of the accuracy in the fundamental, run the PSS simulation with many different accuracy settings, initial conditions and `tstab` values (See “[The tstab Parameter](#)” on page 710). For example, to estimate how much numerical error remains in the calculated fundamental frequency for a given simulation, run the simulation; reduce `reltol`, `iabstol`, and `vabstol` by a factor of 10 to 100; rerun the simulation; and then compare the calculated fundamental frequencies. For the sorts of parameters we recommend for oscillator simulations, four to five digits of precision seems typical. Past that point, round off error and anomalous effects introduced by vastly varying timesteps offset any gains from tightening the various accuracy parameters.

For phase noise calculations, again it is unrealistic to expect relative precision of better than the order of `reltol`. That is, if `reltol` is  $10^{-5}$  and the oscillator fundamental is about 1 GHz, the Spectre RF numerical fuzz for the calculated period is probably about 10 KHz. Therefore, when plotted on an absolute frequency scale, the phase noise calculation exhibits substantial variance within about 10 KHz of the fundamental.

However, when plotted on a frequency scale *relative* to the fundamental, the phase noise calculation might be more precise for many oscillators. If the circuit is strongly dissipative (that is, low- $Q$ , such as ring oscillators and relaxation oscillators), the phase noise calculation is probably fairly accurate up to very close to the fundamental frequency even with loose simulation tolerance settings. High- $Q$  circuits are more demanding of the simulator and require more stringent simulation tolerances to produce good results. In particular, circuits that use varactor diodes as tuning elements in a high- $Q$  tank circuit appear to cause occasional problems. Small modifications to the netlist (runs with different `tstab` values and minor topology changes) can usually tell you whether (and where) the simulator results are reliable.

Simulation accuracy is determined by how precisely Spectre RF simulation can solve the augmented nonlinear boundary value problem that determines the periodic steady-state. The accuracy of the BVP solution is controlled primarily by the simulation variables `reltol`, `iabstol`, `vabstol`, `steadyratio`, and `lteratio`. Typically, `steadyratio` and `lteratio` are fixed, so `reltol` is usually the variable of interest.

Occasionally accuracy might be somewhat affected by other variables such as `relref`, `method`, the number of timesteps, and `tstab`. Again, the physical properties of the circuit might limit the accuracy.

**I have a circuit with an oscillator and a sinusoidal source. Can I simulate this circuit with Spectre RF simulation?**

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

In general, Spectre RF simulation is not intended to analyze circuits that contain autonomous oscillators and independent periodic sources.

If the circuit contains components that could potentially oscillate autonomously and also independent large-signal sinusoidal sources, Spectre RF simulation works properly only if two conditions are fulfilled. The system must be treated as a driven system, and the coupling from the sinusoidal sources to the oscillator components must be strong enough to lock the oscillator to the independent source frequency. (In different contexts, this is known as *oscillator entrainment* or *phase-locking*) The normal (nonautonomous) PSS and small-signal analyses function normally in these conditions.

If the autonomous and driven portions of the circuit are weakly coupled, the circuit waveform might be more complicated; for example, a two-tone (quasi-periodic) signal with incommensurate frequencies. (Incommensurate frequencies are those for which there is no period that is an integer multiple of the period of each frequency.) Even if PSS converges, further small-signal analyses (PAC, PXF, Pnoise) almost certainly give the wrong answers.

#### What is the significance of total noise power?

First, you must understand that Spectre RF simulation calculates and measures noise in voltages and currents. The total power in the phase process is unbounded, but the power in the actual state variables is bounded.

Oscillator phase noise is usually characterized by the quantity

$$d(f) = \frac{S_V(f)}{P_1}$$

where  $P_1$  is the power in the fundamental component of the steady state solution and  $S_V(f)$  is the power spectral density of a state variable  $V$ .

For an oscillator with only white-noise sources,  $L(f)$  has a Lorentzian line shape,

$$L(f) = \frac{1}{\pi} \frac{a}{a^2 + f^2}$$

where  $a$  is dependent on the circuit and noise sources, and thus the total phase noise power

$$\int L(f)df = 1$$

Because

$$\text{var}_v(t) = R_v(t,t) = \int_{-\infty}^{\infty} S_v(f)df$$

we are led to the uncomfortable, but correct, conclusion that the variance in any variable is 100 percent of the RMS value of the variable, *irrespective of circuit properties or the amplitude of the noise sources*.

Physically, this means that if a noise source has been active, since  $t = -\infty$ , then the voltage variable in question is randomly distributed over its whole trajectory. Therefore, the relative variance is one. Clearly, the variance is not a physically useful characterization of the noise, and the total noise power must be interpreted carefully. What is actually needed is the variance as a function of time, given a fixed reference for the signal in question; or, more often, the rate at which the variance increases from a zero point; or, sometimes, the increment in the variance from cycle to cycle. That is, we want to specify the phase of the oscillator signal at a given time point and to find a statistical characterization of the variances relative to that time. But because of the non-causal nature of the Fourier integral, quantities like the total noise power give us information about the statistical properties of the signal over all time.

### **What's the story with pure linear oscillators (LC circuits)?**

Oddly enough, Spectre RF simulation is not set up to do Pnoise analysis on pure LC circuits.

Pure LC circuits are not physically realizable oscillators, and the mathematics that describes them is different from the mathematics that describes physical oscillators. A special option must be added to the code in order for Pnoise to handle *linear oscillators*. See [“Models for Phase Noise”](#) on page 693, and, in particular, [“Amplitude Noise and Phase Noise in the Linear Model”](#) on page 699. Because the normal NOISE analysis is satisfactory for these circuits and also much faster, it is unlikely that Pnoise is modified.

### **Why doesn't the Spectre RF model match my linear model?**

As is discussed in [“Amplitude Noise and Phase Noise in the Linear Model”](#) on page 699, the difference between the Spectre RF model (the correct answer) and the linear oscillator model is that in the linear oscillator, both the amplitude and the phase fluctuations can become large. However, in a nonlinear oscillator, the amplitude fluctuations are always bounded, so the noise is half as much, asymptotically.

We emphasize that computing the correct total noise power requires using the time-varying small signal analysis. An oscillator is, after all, a time-varying circuit by definition. Time-invariant analyses, like the *linear oscillator model*, can sometimes be useful, but they can also be misleading and should be avoided.

### **There are funny sidebands/spikes in the oscillator noise analysis. Is this a bug?**

Very possibly this is parametric small-signal amplification, a real effect. This sometimes occurs when there is an AGC circuit with a very long time constant modulating the parameters of circuit elements in the oscillator loop. Sidebands in the noise power appear at frequencies offset from the oscillator fundamental by the AGC characteristic frequency.

Similarly, any elements that can create a low-frequency parasitic oscillation, such as a bias inductor resonating with a capacitor in the oscillator loop, can create these sorts of sidebands.

## **Further Reading**

The best references on the subject of phase noise are by Alper Demir and Franz Kaertner. Alper Demir's thesis [10], now a Kluwer book, is a collection of useful thinking about noise. Kaertner's papers [11, 12, 9] contain a reasonably rigorous and fairly mathematical treatment of phase noise calculations.

The book by W. P. Robins [13] has a lot of engineering-oriented thinking. However, it makes heavy use of LTI models, and much of the discussion about noise cannot be strictly applied to oscillators. As a consequence, you must interpret the results in this book with care.

Hajimiri and Lee's paper [4] is worth reading, but their analysis is superseded by Kaertner's.

Other references include [8, 14, 15, 16].

## **References**

- [1] P. Kloeden and E. Platen, *Numerical Solution of Stochastic Differential Equations*. Springer-Verlag, 1995.
- [2] D. Leeson, "A simple model of feedback oscillator noise spectrum," *Proc. IEEE*, vol. 54, pp. 329–330, 1966.
- [3] W. A. Gardner, *Introduction to random processes*. McGraw Hill, 1990.
- [4] A. Hajimiri and T. Lee, "A general theory of phase noise in electrical oscillators," *IEEE Journal of Solid. State Circuits*, vol. 33, pp. 179–193, 1998.
- [5] R. Telichevesky, J. White, and K. Kundert, "Efficient steady-state analysis based on matrix-free krylov-subspace methods," in *Proceedings of 32rd Design Automation Conference*, June 1995.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Oscillator Noise Analysis

---

- [6] R. Telichevesky, J. White, and K. Kundert, "Efficient AC and noise analysis of two-tone RF circuits," in *Proceedings of 33rd Design Automation Conference*, June 1996.
- [7] M. Okumura, T. Sugaware, and H. Tanimoto, "An efficient small-signal frequency analysis method of nonlinear circuits with two frequency excitations," *IEEE Transactions on Computer-Aided Design*, vol. 9, pp. 225–235, 1990.
- [8] W. Anzill and P. Russer, "A general method to simulate noise in oscillators based on frequency domain techniques," *IEEE Transactions on Microwave Theory and Techniques*, vol. 41, pp. 2256–2263, 1993.
- [9] F. X. Kärtner, "Noise in oscillating systems," in *Proceedings of the Integrated Nonlinear Microwave and Millimeter Wave Circuits Conference*, 1992.
- [10] A. Demir, *Analysis and simulation of noise in nonlinear electronic circuits and systems*. PhD thesis, University of California, Berkeley, 1997.
- [11] F. X. Kaertner, "Determination of the correlation spectrum of oscillators with low noise," *IEEE Trans. Microwave Theory and Techniques*, vol. 37, pp. 90–101, 1989.
- [12] F. X. Kaertner, "Analysis of white and f-a noise in oscillators," *Int. J. Circuit Theory and Applications*, vol. 18, pp. 485–519, 1990.
- [13] W. P. Robins, *Phase Noise in Signal Sources*. Institution of Electrical Engineers, 1982.
- [14] A. A. Abidi and R. G. Meyer, "Noise in relaxation oscillators," *IEEE J. Sol. State Circuits*, vol. 18, pp. 794–802, 1983.
- [15] B. Razavi, "A study of phase noise in cmos oscillators," *IEEE J. Sol. State Circuits*, vol. 31, pp. 331–343, 1996.
- [16] K. Kurokawa, "Noise in synchronized oscillators," *IEEE Transactions on Microwave Theory and Techniques*, vol. 16, pp. 234–240, 1968.
- [17] K. Kundert, "The Designer's Guide to Spice & Spectre," *Kluwer Academic Publishers*, 1995.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Oscillator Noise Analysis

---

---

## Using PSS Analysis Effectively

---

Periodic steady-state (PSS) analysis is a prerequisite for all periodic small-signal analyses such as the Periodic AC (PAC), Periodic Transfer Function (PXF), Periodic S-Parameter (PSP) and Periodic Noise (Pnoise) analyses provided by Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF). PSS provides a rich set of parameters to help you adapt it to your own applications. For most circuits, PSS converges with the default parameter values. However, for some difficult circuits, changing the values of some parameters is necessary to achieve convergence.

This appendix describes methods you can use to remedy nonconvergence. This appendix also tells you how to improve convergence and efficiency using hierarchical PSS runs.

This appendix is divided into the following three main sections:

“General Convergence Aids” on page 719 describes techniques you can use to resolve PSS nonconvergence with both driven and autonomous circuits

“Convergence Aids for Oscillators” on page 721 describes techniques you use only with autonomous circuits such as oscillators

“Running PSS Analysis Hierarchically” on page 722 describes how to run a sequence of PSS analyses to improve the convergence, efficiency, and quality of the PSS solution used in subsequent periodic small-signal analyses

### General Convergence Aids

You can use the convergence aids described in this section to remedy PSS nonconvergence with driven as well as autonomous circuits. Autonomous circuits are usually harder to converge than driven circuits.

#### Adjusting the `steadyratio` and `tstab` Parameters

You can converge most difficult circuits by manipulating the `steadyratio` and `tstab` parameters.

The `steadyratio` parameter guards against false convergence. However, in unusual situations, the default value for `steadyratio` might be too conservative ( $1.0e^{-3}$  is the default).

The PSS convergence criteria (for voltage-valued variables) is roughly

$$|\Delta v| < (reltol \times |vsig| + vabstol) \times steadyratio \times lteratio$$

To solve the periodic steady-state problem, Spectre RF simulation replaces the time derivatives in the one-period time interval with discrete differences. This turns the nonlinear, continuous differential equations into a set of discrete nonlinear equations that the Spectre RF simulator can solve.

The `steadyratio` parameter specifies the accuracy requirements for the discrete system. The discrete system might have its own solution (a steady state of the discrete difference equations) independent of what is happening in the continuous limit. In other words, the discrete system might be solved to zero tolerance. This happens frequently, particularly in driven circuits which is why setting a conservative `steadyratio` value generally works quite well.

In some cases, however, the solution to zero tolerance might not occur oscillators seem to be especially problematic. Avoid setting the convergence tolerance too tightly. For example, if `reltol` =  $1.0e^{-6}$  and `steadyratio` =  $1.0e^{-3}$ , the relative tolerance for solving the discrete system is approximately `reltol` × `steadyratio` =  $1.0e^{-9}$ . This tolerance level approaches the limit of precision the simulator can provide. In this situation, loosen `steadyratio` to 1.0 or 0.1 and reduce the precautions against false convergence.

Providing a larger value for `tstab` usually improves convergence. Occasionally, you must set `tstab` to a value equal to or greater than the time needed for the circuit to reach approximate steady state.

## Additional Convergence Aids

Below is a list of additional suggestions that sometimes help convergence.

- Carefully evaluate and resolve any notice, warning, or error messages.
- While trapezoidal rule ringing is simply annoying in transient analysis, in PSS analysis it can cause the shooting iteration to stall before convergence is achieved. You can remedy this problem by changing the PSS options `method` parameter from `traponly` to either `trap`, `gear2`, or `gear2only`.



- Help convergence by increasing the `maxperiods` parameter to increase the maximum iterations for shooting method to use. Sometimes a PSS analysis might simply need more than the default number of iterations to converge. However, in some situations convergence does not occur regardless of the number of iterations. In this case, increasing the iteration limit simply causes the simulation to take longer to fail.
- Decrease the maximum allowed time step to help convergence. To adjust the time step, either decrease the `maxstep` parameter or increase the `maxacfreq` parameter.
- Use the `errpreset` parameter properly. Use `liberal` for digital circuits; use `moderate` for typical analog circuits; use `conservative` for sensitive analog circuits (for example, charge storage circuits).

## Convergence Aids for Oscillators

Oscillator circuits are usually harder to converge than their driven counterparts. In addition to manipulating the `steadyratio` parameter, as discussed in [“General Convergence Aids”](#) on page 719, set the `tstab` parameter large enough so that the oscillation amplitude increases almost to its steady-state value and most other transients die out. You can estimate the required value of `tstab` by performing a transient analysis, or in the PSS analysis itself, set `saveinit = yes`.

For some circuits, the oscillation might die out before the oscillator builds up a final value, or the circuit might oscillate temporarily but then return to a zero state. Setting `saveinit = yes` lets you view the initial transient waveforms to identify the problem. This problem might be due to difficulty in starting the oscillator, or it might be caused by artificial numerical losses introduced by very large time steps. The latter is particularly likely if you set the `method` parameter to `gear2only`, `gear2`, or `euler`. In this case, you might try using `method = traponly`. If the problem persists, force the simulator to use smaller step sizes by decreasing `reltol` or by setting the `maxstep` parameter.

With autonomous PSS analysis, exclusive use of the trapezoidal rule can lead to ringing that spans the length of the oscillation period and causes convergence problems. When you set `method = trap`, the Spectre RF simulator occasionally takes a backward Euler step, which acts to damp the ringing. The `gear2` and `gear2only` methods use Gear's backward difference method, which is not subject to ringing. Each of these alternatives to `traponly` avoids trapezoidal rule ringing and the attendant convergence problems at the expense of adding a small amount of artificial numerical damping. This damping slightly reduces the computed Q of the oscillator.

Be sure that the method you choose to start your oscillator is effective. It must *kick* the oscillator hard enough to start the oscillation and make the oscillator respond with a signal level that is between 25 and 100 percent of the expected final level. Avoid kicking the oscillator

so hard that it responds in an unnecessarily nonlinear fashion. Also try to avoid exciting response modes in the circuit that are unrelated to the oscillation, especially those associated with long time constants.

Try to improve your estimate of the period. The simulator uses your estimate of the period to determine the length of the initial transient analysis interval. This interval is used to measure the oscillation period. If you specify a period that is too short, the estimate of the oscillation period is not accurate, and the PSS analysis might fail. Overestimation of the period is not a serious problem because the only disadvantage is a longer initial transient interval. However, significant overestimation can result in an excessively long simulation time.

Sometimes the analysis might need more than the default number of iterations (`maxperiods = 50`) to converge. This is more likely to occur with high-Q circuits. You can increase the maximum iterations for shooting methods using the `maxperiods` parameter.

## Running PSS Analysis Hierarchically

For most circuits, a single PSS analysis run is sufficient to find the periodic steady-state solution. However, for some difficult circuits it is preferable, or even necessary, to run PSS analysis multiple times to find the steady-state solution with the parameter settings you want (for example, with a tight `reltol`).

The biggest obstacle to PSS convergence is poorly chosen initial conditions. The backbone of PSS analysis is Newton's method. Theoretically, when the initial guess is close enough to the solution and the problem is not ill-conditioned, Newton's method is guaranteed to converge because of its contraction property. Consequently, it is very important to provide the best initial conditions that you can to ensure rapid convergence.

To run PSS hierarchically, you

1. Start by running a minimally accurate PSS analysis to obtain, with high likelihood, a coarse-grid solution that converges in an acceptably short simulation time.
2. Use the coarse-grid solution as the initial condition for another PSS run to achieve a fine-grid solution.
3. This practice significantly increases the chance that the fine-grid PSS analysis converges promptly. As you might expect, it is also more efficient. Running PSS hierarchically often reduces the total simulation time needed to find a periodic steady-state solution because it reduces the number of time-consuming fine-grid PSS iterations.

The `writefinal` and `readic` parameters serve as threads between hierarchical PSS runs.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSS Analysis Effectively

---

- When you set `writefinal` to *SomeFileName*, both the associated time and period information (for autonomous PSS) and the final transient solution at PSS steady-state are saved to the file, *SomeFileName*.
- When you run another similar PSS analysis and you set `readic` to *SomeFileName* and `skipdc` to `yes`, setting `skipdc` to `yes` forces the simulator to use the initial conditions in the file *SomeFileName* as the initial transient solution for the first PSS iteration.

For example, if you want to find the periodic steady-state solution with a tight `reltol` such as  $1.0e^{-5}$ , you might

- Run an initial PSS analysis with a looser tolerance; for example, `reltol = 1.0e-3`
- Use the `writefinal` parameter in the initial PSS analysis to write out the final results to a file. A PSS analysis runs faster with the looser tolerance because fewer time points are generated during each transient integration performed during each PSS iteration.
- Run a second PSS analysis with the tighter tolerance, `reltol = 1.0e-5`
- Use the `readic` and `skipdc` in the second PSS analysis to read in the final results of the first PSS analysis as the initial conditions.
- After the second PSS analysis, you can run small-signal analyses such as PAC.

```
set1 set reltol=1.0E-3
pss1 pss ... writefinal="SomeFile"
set2 set reltol=1.0E-5
pss2 pss ... readic="SomeFile" skipdc=yes
... pac ... ..
```

Always use a sequence of PSS runs when you need a tight tolerance PSS solution. If necessary, you can run more than two PSS analyses in the hierarchical process. You choose the tolerance sequence for the continuation. The multiple PSS approach usually produces a better periodic steady-state solution for subsequent small-signal analyses.

The above procedure is often called a continuation on the simulation parameter `reltol`. However, you can use continuation with many other simulation parameters. For example, in order to achieve PSS convergence at a high input power that causes nonconvergence, you might gradually increase the input power at an RF port.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using PSS Analysis Effectively

---

---

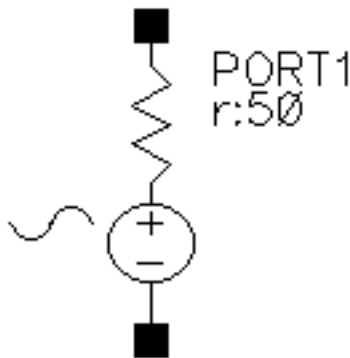
## Using the psin Component

---

This appendix describes how to use the *psin* component from the *analogLib* library in Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) simulations within the analog design environment.

### Independent Resistive Source (*psin*)

The *psin* component, located in the *analogLib* library, can be used in all RF circuits for Spectre RF and Spectre S-parameter simulations.



When you netlist *psin* in the Analog Design Environment using the SpectreS simulator, you can see that *psin* is the *port* component in Spectre simulation. A port is a resistive source that is tied between positive and negative terminals. It is equivalent to a voltage source in series with a resistor, and the reference resistance of the port is the value of the resistor.

### Capabilities of the *psin* Component

While the *psin* component is generally useful as a stimulus in high-frequency circuits, it also has the following three unique capabilities.

It defines the ports of the circuit to the S-parameter analysis

- It has an intrinsic noise source that lets the noise analysis directly compute the noise figure of the circuit
- Is the only source for which you can specify the amplitude in terms of power

### Terminating the `psin`

Be aware that when you specify the voltage on a `psin`, Spectre RF assumes that the `psin` is properly terminated in its reference resistance. The specified voltage value is not the voltage on the internal voltage source, which is actually set to twice the value specified on the `psin`. So, if you use a `psin` source to drive an open circuit, the voltage (for *DC*, *transient*, *AC*, and *PAC* signals) is double its specified value. However, you can alternatively specify the amplitude of the sine wave in the transient and *PAC* analyses as the power in dBm delivered by the `psin` when terminated with the reference resistance.

The top of the `psin` component Edit Properties form is shown in [Figure C-1](#) on page 727.

Figure C-1 Top of the psin Component Edit Properties Form

Property	Value	Display
Library Name	analogLib	off
Cell Name	psin	off
View Name	symbol	off
Instance Name	PORT1	off

User Property	Master Value	Local Value	Display
IvsIgnore	TRUE		off

CDF Parameter	Value	Display
Frequency name		off
Second frequency name		off
Noise file name		off

## Parameter Types for the psin Component

The *psin* component parameters described here grouped by parameter types, rather than by the order they appear on the *psin* Edit Properties form.

### Name Parameters

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using the psin Component

---

Frequency name

Second frequency name

### **psin Instance Parameter**

DC voltage

### **General Waveform Parameters**

Source type

Delay time

### **Sinusoidal Waveform Parameters**

Sine DC level

Amplitude

Amplitude (dBm)

Frequency

Initial phase for Sinusoid

Amplitude 2

Frequency 2

Phase for sinusoid 2

### **Amplitude Modulation Parameters**

AM modulation frequency

AM modulation phase

AM modulation index

### **FM Modulation Parameters**

FM modulation frequency

FM modulation index



# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using the psin Component

---

Damping factor

### **Noise Parameters**

Noise file name

Number of noise/freq pairs

### **Port Parameters**

Resistance

Port number

Multiplier

### **Temperature Effect Parameters**

Temperature coefficient 1

Temperature coefficient 2

Nominal temperature (See the following discussion about the effect of temperature parameters on the voltage level.)

### **Small-Signal Parameters**

AC magnitude

AC phase

XF magnitude

PAC magnitude

PAC magnitude (dBm)

PAC phase

**Note:** The “Additional Notes” section on page 741 contains information about active parameters in analyses and terminating the psin.

## Name Parameters

*Frequency Name* and *Second frequency name* let you assign names to fundamental tones. After you save the schematic, the names you assign appear in the *Fundamental Tones* list box on the Choosing Analyses form.

## psin Instance Parameter

### DC voltage

Sets the DC level of the source for DC analysis. The value must be a real number. If you do not specify the DC value, it is assumed to be the *time* =0 value of the waveform. Default: 0  
Units: V

Because all small signal analyses (*ac*, *xf*, and *noise*) use DC analysis results, the *DC voltage* level also affects small-signal analyses. Transient analysis is not affected unless you specify *type*=dc or use dc as a default for the other waveform types.

## General Waveform Parameters

### Source type

Lets you specify several different waveform shapes and quickly switch between them by changing the value in the *type* field. The typical source types used in Spectre RF analyses are *dc* and *sine*. Possible source type values are dc, pulse, or sine.

If you set the source to *type* = dc, you get only DC values even if you specify a sinusoid later. For *type*=dc, the *dc* and *tc* parameters are active and set the DC level for all analyses. In DC analysis, this setting also determines the DC level generated by the source regardless of what type you specify.

With the setting *type*=sine, you create a sinusoidal waveform whose parameters you can alter. These parameters are described in more detail in the *Sinusoidal Waveform Parameters* section below.

You cannot specify the parameters for *type*=pulse in the *psin Edit Properties* or *Add Component* forms. Source types *pwl* and *exp* are not supported for the analogLib *psin* component.

Delay time

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the psin Component

---

The waveform delay time, or the time that the source stays at the DC level before it starts generating waveforms (assuming the source is set to *sine*). The value must be a real number. Default: 0 Units: `seconds`

## Sinusoidal Waveform Parameters

The *psin* component can generate up to two sinusoids simultaneously. They are denoted 1 and 2. You can set the amplitude, frequency and phase for both individually. The amplitude can be set to either a voltage or a power level. You can also specify sinusoidal AM or FM modulation of sinusoid 1.

The first sinusoid is described by the parameters *Amplitude*, *Amplitude dBm*, *Frequency*, *Initial phase for sinusoid*, *AM or FM modulation terms*, and *Damping factor*.

The second sinusoid is described by the parameters *Amplitude2*, *Amplitude 2 (dBm)*, *Frequency 2*, and *Initial phase for sinusoid 2*.

The second sinusoid cannot be modulated.

### Sine DC level

Sets the DC level for sinusoidal waveforms in transient analyses. This parameter is used when the sinusoid has a different average level than the one specified for the DC analyses. If not specified, the average value of the sinusoid is the same as that of the DC level of the source. The value must be a real number. Default: `dc` Units: `v`

## Amplitude

Peak amplitude of the first sinusoidal waveform that you generate. The value must be a real number. Default: 1 Units: `v`

Remember, when you specify the voltage on a *psin*, you are specifying the voltage *when the psin is properly terminated*, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *psin*.

### Amplitude (dBm)

Amplitude of the first sinusoidal waveform when specified in dBm (alternative to Amplitude). The value must be a real number. Units: `dBm`



### Caution

**Set either Amplitude or Amplitude (dBm). Do not set them both! If you specify both Amplitude (in peak volts) and Amplitude (dBm) in the same source, Spectre RF simulation does not give you any errors or warning messages. It uses Amplitude (in peak volts) and ignores the Amplitude (dBm) field. If you specify Amplitude, verify that the Amplitude (dBm) field is empty, and vice-versa.**

## Frequency

The frequency of the first sinusoidal waveform (carrier frequency). You typically use unmodulated signals in Spectre RF analyses. The value must be a real number. Default: 0  
Units: Hz

## Phase for Sinusoid

The phase at the specified delay time. The sinusoidal waveform might start before the given delay time in order to achieve a specified phase and still remain continuous. For example, if you want to generate a cosine wave, set this parameter to 90°. The value must be a real number. Default: 0 Units: degrees

## Amplitude Modulation Parameters

The amplitude modulation (double sideband large carrier, or DSB-LC) is defined as

$$v_{AM}(t) = A (1 + m \sin(2\pi f_m t + \phi)) \sin(2\pi f_c t)$$

where

- A is the carrier amplitude (amplitude of sinusoid)
- $f_m$  is the AM modulation frequency
- $\phi$  is the AM modulation phase
- m is the AM modulation index
- $\sin(2\pi f_c t)$  is the carrier signal

The amplitude modulation parameters affect *only* the first sinusoid generated by *psin*. They have no effect on the second sinusoid.

### **AM modulation frequency**

AM modulation frequency for the sinusoidal waveform ( $f_m$  in the previous equation). The value must be a real number. Default: 0 Units: Hz

### **AM modulation phase**

AM phase of modulation for the sinusoidal waveform ( $\phi$  in the previous equation). The value must be a real number. Default: 0 Units: degrees

### **AM modulation index**

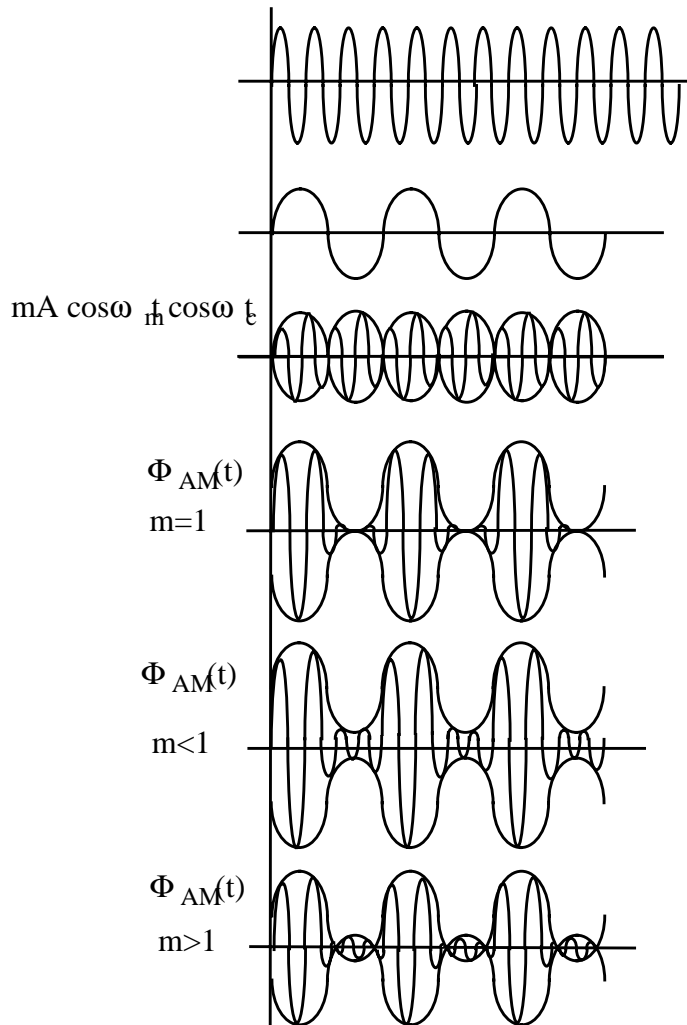
AM index of modulation for the sinusoidal waveform ( $m$  in the previous AM equation). The AM modulation index is a dimensionless scale factor used to control the ratio of the sidebands to the carrier.

$$m = (\text{peak DSB-SC amplitude}) / (\text{peak carrier amplitude})$$

The value must be a real number. Default: 0

The following figure shows the effect of varying modulation indexes for the following three cases:  $m < 1$ ,  $m = 1$ , and  $m > 1$ .  $f_c$  is the carrier frequency, and  $f_m$  is the modulation frequency.

**Figure C-2 Amplitude Modulation: Effects of Varying Modulation Indexes**



### FM Modulation Parameters

The frequency modulation for the sinusoidal case is defined as

$$v_{FM}(t) = A \sin(2\pi f_c t + \beta \sin(2\pi f_m t) + \phi)$$

where

- $A$  is the amplitude of the sinusoid
- $\beta$  is the FM modulation index
- $\sin(2\pi f_m t)$  is the modulation signal

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the psin Component

---

- $f_c$  is the carrier frequency
- $\phi$  is the phase of the sinusoid

The frequency modulation parameters affect *only* the first frequency generated by *psin*. They have no effect on the second frequency.

#### FM modulation frequency

FM modulation frequency for the sinusoidal waveform ( $f_m$  in the equation above). The value must be a real number. Default: 0 Units: Hz

#### FM modulation index

FM index of modulation for the sinusoidal waveform, the ratio of peak frequency deviation divided by the center frequency ( $\beta$  in the above equations).

$$\beta = \Delta f / f_m$$

The value must be a real number. Default: 0

#### Damping factor

Damping factor for the sinusoidal waveform. *Damping factor* specifies the time it takes to go from the envelope (full amplitude) at *time=0* to 63 percent of the full amplitude. For example, consider the following damped sinusoid:

$$v(t) = A e^{-\sigma t} \sin(2\pi f t + \phi)$$

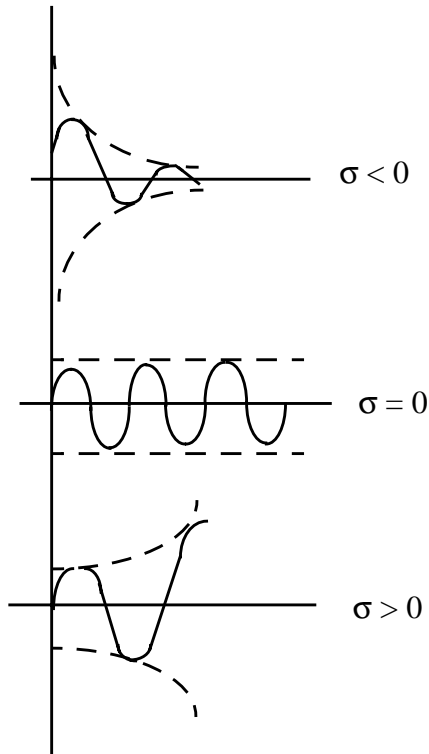
where  $\sigma$  = damping factor.

If  $\sigma = 0$ , the waveform is a pure sinusoid (steady state).

- If  $\sigma < 0$ , the waveform exhibits decaying oscillations.
- If  $\sigma > 0$ , the waveform exhibits growing oscillations.
- It takes 5s to diminish to 1 percent of the peak amplitude. The value must be a real number. Default: 0. Units: 1/seconds

The following figure shows the effect of *Damping factor* on the first sinusoid for three values of  $\sigma$ .

Figure C-3 Effect of Damping Factor on the First Sinusoid



### Amplitude 2

Peak amplitude of the second sinusoidal waveform. The value must be a real number.  
Default: 1 Units: V

Remember, when you specify the voltage on a *psin*, you are specifying the voltage *when the psin is properly terminated*, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *psin*.

### Amplitude 2 (dBm)

Amplitude of the second sinusoidal waveform in dBm (alternative to *Amplitude 2*). The value specified is the power delivered into a matched load. The value must be a real number. Units: dBm





**Set either *Amplitude2* or *Amplitude2 (dBm)*. Do not set them both!**

## Frequency 2

Frequency of the second sinusoidal waveform. The value must be a real number. Default: 0  
Units: Hz

## Phase for Sinusoid 2

The phase at the specified *Delay time* for the second sinusoid. The sinusoidal waveform might start before the given *Delay time* in order to achieve specified phase while still remaining continuous. The value must be a real number. Default: 0 Units: degrees

## Noise Parameters

Noise parameters include *Noise file name*, *Number of Noise Frequency Pairs*, and *Noise temperature*.

### Noise file name

Name of the file containing the excess spot noise data in the form of frequency-noise pairs. The value must be a string. Default: no value

### Number of noise/freq. pairs

You must specify the number of noise-frequency pairs that exist in your noise file in the form. In your file, list the noise-frequency pairs as one pair per line with a space or tab between the frequency and noise values. The values are given as a vector of real number pairs, where noise is given in  $V^2/Hz$ , and frequency is given in Hz.

*Specific to Analog Design Environment netlisting:* If you have more noise-frequency pairs than the number you specify in the *Number of noise frequency pairs* field, the number of noise-frequency pairs used is the number you specify in the field. Any additional noise-frequency pairs in the noise file are ignored. If the number of noise-frequency pairs in the noise file is smaller than the number you specified in the *number of noise frequency pairs* field, the analysis is stopped.

## Noise temperature

The *Noise temperature* of the *psin*. If not specified, the *Noise temperature* is assumed to be the actual temperature of the *psin*. When you compute the noise figure of a circuit driven at its input by a *psin*, set the noise temperature of the *psin* (Spectre parameter *noisetemp*) to 16.85C (290K). This setting matches the standard IEEE definition of noise figure. In addition, disable all other sources of noise in the *psin*, such as the Spectre parameters *noisefile* and *noisevec*. If you want a noiseless *psin*, set the *noise temperature* to absolute zero or below, and do not specify a noise file or noise vector. Default: Units: °C.

### Port Parameters

Port parameters include *Resistance*, *Port Number*, and *Multiplier*.

## Resistance

The reference resistance of the system. The value must be a real number, but not 0. Default: 50 Units: Ω

## Port number

The number of the port. The value must be a nonzero integer. Each *psin* in a schematic must have a different port number. The *Port number* is not automatically indexed when you place each *psin* on your schematic.

### Multiplier

The multiplicity factor. The value must be an integer number greater than zero. This number lets you specify a number of *psins* in parallel. For example, if you set *Resistance* to 50 and *Multiplier* to 2, you specify two *psin* ports in parallel, each with an effective reference resistance of 25 Ω. Default: 1

## Temperature Effect Parameters

### How Temperature Parameters Affect the Voltage Level (Background Information)

The value of the DC voltage can vary as a function of the temperature if you specify *tc1* and *tc2*. The variation is given by

$$V_{DC}(T) = dc * [1 + tc1 * (T - tnom) + tc2 * (T - tnom)^2]$$

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the psin Component

---

where  $T$  is the analysis temperature specified in the analysis options,  $t_{nom}$  is the nominal temperature specified in the *Choosing Analyses* form, and  $dc$  is the DC voltage.

If the analysis temperature equals the nominal temperature, the result is the voltage amplitude that you specified,  $V_{DC}(T) = dc$ .

If the nominal and analysis temperatures differ, the voltage amplitude is given by

$$V_{DC}(T) = dc * [1 + tc1 * (T - t_{nom}) + tc2 * (T - t_{nom})^2]$$

where  $T$  is the analysis temperature you specify in the analysis options and  $t_{nom}$  is the nominal temperature.  $tc1$  and  $tc2$  are the linear and quadratic temperature coefficients.

For example, if the nominal temperature is 27°C and the analysis temperature is 25°C, there is a 2° difference between the nominal and analysis temperature. The voltage amplitude is

$$V_{DC}(T) = dc * [1 + tc1 * (-2) + tc2 * (-2)^2]$$

Temperature effect parameters include *Temperature coefficient 1* (the linear temperature coefficient), *Temperature coefficient 2* (the quadratic temperature coefficient), and *Nominal temperature*.

#### Temperature coefficient 1

First order (linear) temperature coefficient of the *DC voltage* ( $tc1$ ). The value must be a real number. Default: 0 Units: °C<sup>-1</sup>

#### Temperature coefficient 2

Second order (quadratic) temperature coefficient of the *DC voltage* ( $tc2$ ). The value must be a real number. Default: 0 Units: °C<sup>-2</sup>

#### Nominal temperature

The nominal temperature for *DC voltage* ( $t_{nom}$ ). The value must be a real number. Default: Set by options specifications Units: C

#### Small-Signal Parameters

The small signal parameters are *AC magnitude*, *AC phase*, *XF magnitude*, *PAC magnitude*, *PAC magnitude (dBm)*, and *PAC phase*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the psin Component

---

Remember, when you specify the voltage on a *psin*, you are specifying the voltage *when the psin is properly terminated*, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *psin*. The same is true for the values for the *transient*, *AC*, and *PAC* signals. However, the amplitude of the sine wave in the *PAC* and *transient* analysis can alternatively be specified as the power in dBm delivered by the *psin* *when terminated with the reference resistance*

#### AC magnitude

The peak small-signal voltage. The value must be a real number. Default: 0 Units:  $\text{V}$

#### AC phase

The small-signal phase. The value must be a real number. Default: 0 Units: degrees

Typically, only one source in the circuit has *AC Magnitude* set to a value other than zero, and usually it has an *AC magnitude*=1 and *AC phase*=0. However, there are situations where more than one source has a nonzero *AC magnitude*. For example, applying a differential small-signal input could be done with two sources with the *AC magnitudes* set to 0.5 and the *AC phases* set to 0 and 180.

#### XF magnitude

The transfer function analysis magnitude. Use *XF magnitude* to compensate for gain or loss in the test fixture. The value must be a real number. Default: 1 Units:  $\text{V/V}$

#### PAC magnitude

The peak periodic AC analysis magnitude. Setting this value to unity is a convenient way of computing the transfer function from this source to the output. The value must be a real number. Default: 0 Units:  $\text{V}$

#### PAC magnitude (dBm)

The periodic AC analysis magnitude in dBm (alternative to *PAC magnitude*). The value must be a real number. Units: dBm



Caution

**Set either PAC magnitude or PAC magnitude (dBm), but do not set them both!**

## PAC phase

The periodic AC analysis phase. The value must be a real number. Default: 0 Units: degrees

Typically, only one source in the circuit has a *PAC magnitude* set to a value other than zero, and usually it has a *PAC magnitude*=1 and *PAC phase*=0. However, there are situations where more than one source has a nonzero *PAC magnitude*. For example, applying a differential small-signal input could be done with two sources with the *PAC magnitudes* set to 0.5 and the *PAC phases* set to 0 and 180."

You do not specify the *PAC* frequency in the *psin Edit Object Properties* form. Instead, you set the *PAC frequency* in the *PAC Choosing Analyses form*. For example, when making an IP3 measurement, you set the PAC frequency to a variable value in the *Choosing Analyses Form*. Then, you enter the same variable in the *PAC Amplitude* (or *PAC Amplitude dBm*) field of the *psin Edit Object Properties* form.

## Additional Notes

### Active Parameters in Analyses

In DC analyses, the only active parameters are *dc*, *m*, and the temperature coefficient parameters.

In AC analyses, the only active parameters are *m*, *mag* and *phase*.

In Transient analyses, all parameters are active except the small-signal parameters and the noise parameters.

In PAC analyses, the only active parameters are *m*, *PAC magnitude* (amplitude or dBm), and *PAC phase*.

*XF magnitude* is active in *XF* and *PXF* analyses only.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using the psin Component

---

---

## The RF Library

---

### The Contents of the rfLib

The elements contained in the RF Library, *rfLib*, are organized into the following categories:

Categories in <i>rfLib</i>	Description of Category and Link
<i>Original_RFAHDL_lib</i>	The original RFAHDL library <a href="#">“Models for Transistor-Level RF Circuit Design”</a> on page 745
<i>bot_upBB</i>	Bottom-Up Baseband elements <a href="#">“Bottom-Up Design Elements”</a> on page 787
<i>measurement</i>	Measurement elements <a href="#">“Measurement Elements”</a> on page 768
<i>testbenches</i>	Testbench elements <a href="#">“Testbenches Elements”</a> on page 786
<i>top_dwnBB</i>	Top-Down Baseband elements <a href="#">“Models for Top-Down RF System Design”</a> on page 788
<i>top_dwnPB</i>	Top-Down Passband elements <a href="#">“Models for Top-Down RF System Design”</a> on page 788
<i>Everything</i>	Lists all elements in <i>rfLib</i>
<i>Uncategorized</i>	Files created by testbench elements <a href="#">“Uncategorized Elements”</a> on page 787

The *rfLib* contains elements to support the design of both RF circuits and RF systems.

### Elements for Transistor-Level RF Circuit Design

Elements in the category *Original\_RFAHDL\_lib* exist specifically to support transistor-level RF circuit designers.

The *Original\_RFAHDL\_lib* category contains the RF AHDL library. The elements in this library are detailed behavioral models but they are not baseband equivalent. Models in *Original\_RFAHDL\_lib* are documented in [“Models for Transistor-Level RF Circuit Design”](#) on page 745.

## Elements for Top-Down System-Level RF Design

Elements in the categories *top\_dwnBB* and *top\_dwnPB* exist to support system-level RF designers.

The *top\_dwnBB* category contains the Top-Down Baseband models of common architectural function blocks. The default view of these models is the baseband view (called *veriloga*) but most models in this category also have a differential passband view (called *veriloga\_PB*). The only exceptions are the *BB\_loss* and *VGA\_BB* models which are meant only for baseband analysis and have no passband view.

The *top\_dwnPB* category contains the Top-Down Passband models—single-ended passband versions of the baseband models.

The elements in *top\_dwnBB* capture only information that can be modeled at baseband. The elements in *top\_dwnPB* are passband versions of the baseband models which allow the RF system designer to switch between baseband and passband views during the design process.

The *top\_dwnBB* and *top\_dwnPB* models are documented in [“Models for Top-Down RF System Design”](#) on page 788.

## Elements for Use With Both Design Methodologies

Elements in the *measurement* and *testbenches* categories are not part of either RF architecture. They can be used by both RF system designers and RF circuit designers.

The *measurement* category contains models of the instrumentation blocks and baseband signal generators used to facilitate measurements and diagnostics.

The *testbenches* category contains the test circuits used to define model specifications. Where possible, the element names are in terms of standard RF measurements.

The models in the *measurement* category are documented in [“Measurement Elements”](#) on page 768. The circuits in the *testbenches* category are documented in [Chapter 8, “Methods for Top-Down RF System Design.”](#)



## Elements for Bottom Up Transmitter Design

The *bot\_upBB* category contains the Bottom-Up Baseband models—behavioral baseband J-models for transmitters. These J-models are documented in “[Bottom-Up Design Elements](#)” on page 787.

## Models for Transistor-Level RF Circuit Design

If you are a transistor-level RF circuit designer who creates top-down designs, you must model functional RF blocks at the behavioral level. You can use the high-level models from *Original\_RFAHDL\_lib* as building-blocks for complex RF systems or executable specifications. You can also use the models from *bot\_upBB*. They are described in “[Bottom-Up Design Elements](#)” on page 787.

When distributed in source code format, these models also help you write libraries that reflect your own design needs.

The Verilog-A<sup>®</sup> language lets you use the RF library models in Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) without an explicit equivalent circuit. This approach uses the high-level description language capability of Spectre RF to create high-level models that capture the essential information of RF functional blocks. You can then insert these models into regular RF circuits for simulation. The essential parameters are translated into coefficients and equations that describe the relations between the voltages and currents at the connecting nodes.

### Original\_RFAHDL\_lib Elements

You can characterize most RF circuits with three sets of parameters

- Linear Parameters
- Nonlinear Parameters
- Noise Parameters.

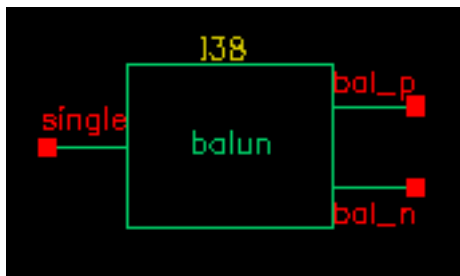
The RF AHDL library, *Original\_RFAHDL\_lib*, contains models for top-down design. It contains all of the elements of the original *rfLib*. The models in *Original\_RFAHDL\_lib* are designed specifically to work with Spectre RF as these models do not have hidden states.

The *Original\_RFAHDL\_lib* contains the elements discussed in the sections that follow.

- [Balun](#)
- [balun\\_com](#)

- Filters
- Low Noise Amplifier
- Mixer
- Power Amplifier
- Oscillator
- Quadrature Signal Generator
- Phase Shifter

## Balun



The balun (balancing transformer) is used in circuits that require single/differential signal transformation. Although a passive network (including the transformer) is used to achieve balun, this implementation employs a three-port network. There are three ports (or nodes), because the reference nodes are always at the global ground: *single*, *bal\_p*, and *bal\_n*.

The three ports are

- single* Single end
- bal\_p* In-phase end of the balanced output
- bal\_n* Out-of-phase end of the balanced output

When the ports are numbered as *single*(1), *bal\_p*(2), and *bal\_n*(3), the S-parameter for the three-port network is

$$S = \begin{bmatrix} 0 & t & -t \\ t & 0 & 0 \\ -t & 0 & 0 \end{bmatrix}$$

where

$$t = \frac{10^{-\text{loss}/10}}{\sqrt{2}}$$

when loss is specified in dB.

This module can also be used in common mode cancellation applications.

The module is declared as follows

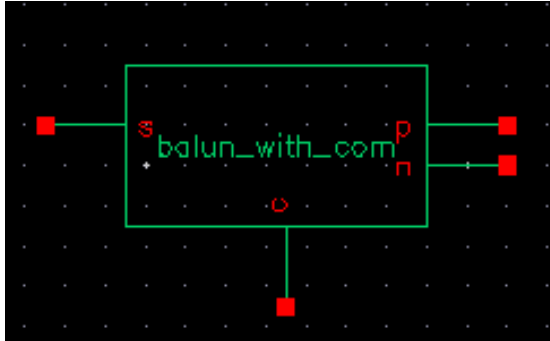
```
module balun(single, bal_p, bal_n);
  inout single, bal_p, bal_n;
  electrical single, bal_p, bal_n;
  parameter real rin = 50 from (0:inf);
  parameter real rout = 50 from (0:inf);
  parameter real loss = 0 from [0:inf];
```

Parameters include the input impedance (for single end), the output impedance (for balanced end to ground), and the insertion loss (from single end to balanced end and from balanced end to single end).

The parameters are

<code>rin</code>	Input impedance [ $\Omega$ ]
<code>rout</code>	Output impedance [ $\Omega$ ]
<code>loss</code>	Insertion loss [dB]

## Balun\_com



The balun\_com has, in addition to the three ports of the balun, an external reference node that can be used for DC bias set up in the balanced end. The balun\_com is equivalent to the balun when the voltage of the reference node *c* is set to 0.

The four ports of the balun\_com are

<i>s</i>	Single end
<i>p</i>	In-phase end of the balanced output
<i>n</i>	Out-of-phase end of the balanced output
<i>c</i>	Common (reference) node for <i>p</i> and <i>n</i>

The module is declared as follows

```
module balun_com(s, p, n, c);
  inout s, p, n, c;
  electrical s, p, n, c;
  parameter real rin = 50 from (0:inf);
  parameter real rout = 50 from (0:inf);
  parameter real loss = 0 from [0:inf];
```

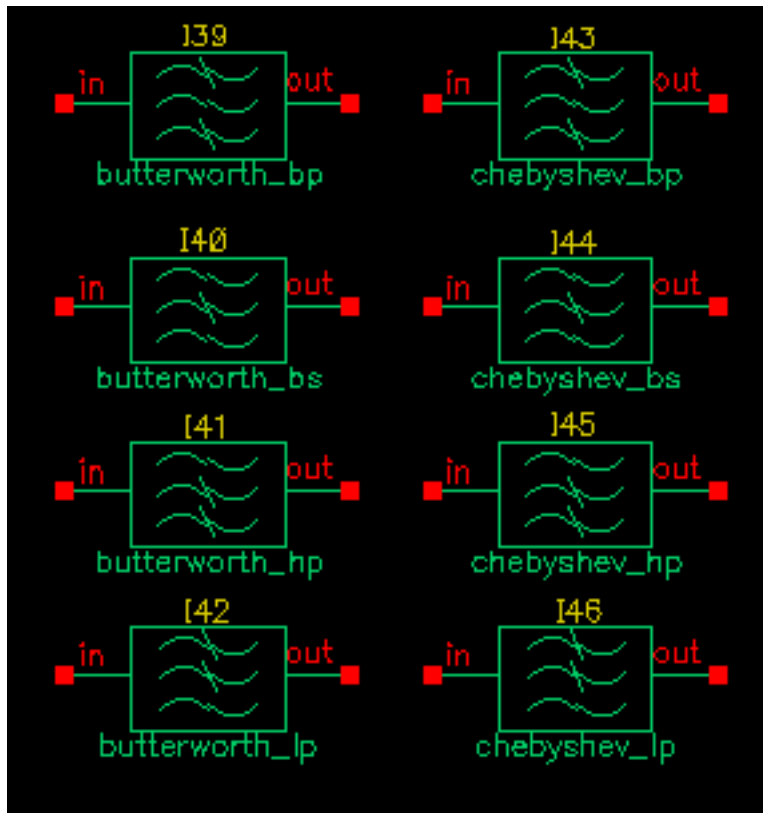
Parameters include the input impedance (for single end), the output impedance (for balanced end to ground), and the insertion loss (from single end to balanced end and from balanced end to single end).

The parameters are

<i>rin</i>	Input impedance [ $\Omega$ ]
<i>rout</i>	Output impedance [ $\Omega$ ]

loss      Insertion loss [dB]

## Filters



Filter properties are specified in the frequency domain, but it is not easy for Spectre RF to process frequency-domain data. Spectre RF simulation requires a large signal, time-domain model to simulate filter behavior.

As part of the RF AHDL library, filters are implemented using a network synthesis technique which consists of the following two steps:

1. Calculate the normalized low-pass filter prototype, which consists of serial inductors and parallel capacitors
2. Perform frequency transformation and scaling to synthesize the frequency responses of the filter type

The synthesized model contains many inductors and capacitors. They are implemented using the integral and differential functions of the Verilog-A language. Insertion loss is added using

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

the S-parameter network technique. This network essentially dampens the signal flow by the specified insertion loss value.

Low-pass, high-pass, bandpass, and bandstop filters are implemented and each of these can be a Butterworth or Chebyshev type filter (for a total of eight filters) as follows

butterworth_bp	chebyshev_bp
butterworth_bs	chebyshev_bs
butterworth_hp	chebyshev_hp
butterworth_lp	chebyshev_lp
butterworth_bp_laplace	chebyshev_bp_laplace
butterworth_bs_laplace	chebyshev_bs_laplace
butterworth_hp_laplace	chebyshev_hp_laplace
butterworth_lp_laplace	chebyshev_lp_laplace

In the current implementation of the Verilog-A language, the order and internal states of the filter cannot be dynamically allocated. You must use the `'define` derivative in the Verilog-A source code to specify the order. Use S-parameters to test the filters because S-parameters capture the input/output impedance matching.

For example, the Butterworth bandpass filter, `butterworth_bp`, has the following module declaration:

```
module butterworth_bp(t1, t2);
    inout t1, t2;
    electrical in, out;
    parameter real r1 = 50 from (0:inf);
    parameter real r2 = 50 from (0:inf);
    parameter real f0 = 1e9 from (0:inf);
    parameter real bw = 0.10 from (0:0.5);
    parameter real fc = 1e9 from (0:inf);
    parameter real loss = 0 from [0:inf];
```

where `t1` and `t2` are the input and output nodes, respectively.

The parameters are

<code>r1</code>	Input impedance [ $\Omega$ ]
<code>r2</code>	Output impedance [ $\Omega$ ]

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

<code>f<sub>c</sub></code>	Corner frequency (3 dB point) for low-pass and high-pass filter [Hz]
<code>f<sub>0</sub></code>	Center frequency for bandpass or bandstop filter [Hz]
<code>bw</code>	Relative frequency for bandpass or bandstop filter [Hz]
<code>loss</code>	Insertion loss [dB]

Figure is the simple schematic used to test the filter. Two ports are used to obtain the S-parameters.

#### Schematic for Testing Filter Models

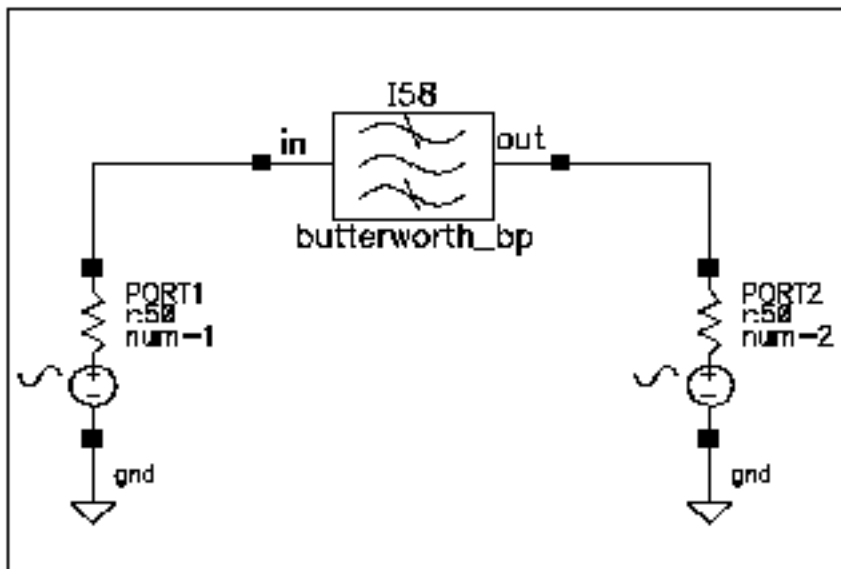
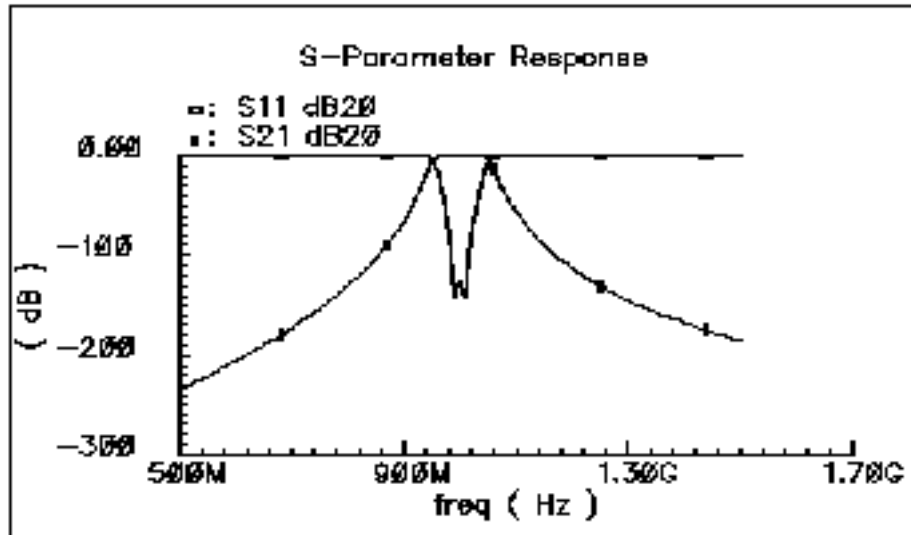
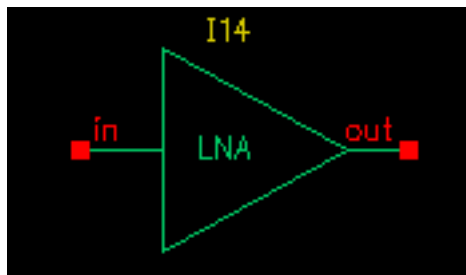


Figure [D-1](#) shows the calculated S-parameters of this Butterworth bandpass filter, which has a center frequency of 1 GHz and a relative bandwidth of 10 percent. The order of this specific filter is 10.

Figure D-1 S-Parameters of a Butterworth Filter



## Low-Noise Amplifier



Low-noise amplifiers (LNAs) are commonly used in receiver design to amplify the signal with a low noise figure. A typical LNA has the following three sets of parameters:

- Linear model
- Nonlinear model
- Noise model.
- The module is declared as follows:

```
module lna(in, out);  
  inout in, out;  
  electrical in, out;  
  parameter real nf = 2 from [0:inf);
```



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

```
parameter real ip3 = -10;
parameter real gain = 15 from [0:inf);
parameter real isolation = 200 from (0:inf);
parameter real rin = 50 from (0:inf);
parameter real cin = 0 from [0:100];
parameter real rout = 50 from (0:inf);
parameter real cout = 0 from [0:100];
parameter real gammain = -150 from (-inf:0);
parameter real mismatch = 1 from [-1:1] exclude (-1:1);
parameter real gammaout=-150 from (-inf:0);
```

The parameters are

nf	Noise figure [dB]
ip3	Input referenced IP3 [dBm]
gain	S21 (power gain) [dB]
isolation	S12 [dB]
rin	Reference impedance of the input port [ $\Omega$ ]
rout	Reference impedance of the output port [ $\Omega$ ]
gammain	Input return loss [dB]
mismatch	Mismatch sign of input. 1: input impedance > reference impedance -1: otherwise
gammaout	Output return loss [dB]
cin	Parasitic input capacitance [pF]
cout	Parasitic parallel output capacitance [pF]

Internally, a set of linear equations is constructed to satisfy the S-parameters. Furthermore, nonlinearity, expressed by a third-order polynomial function, is added to the gain (or S21) to describe the IP3. Excess white noise is added at the input port to describe the noise figure.

IP3 is the measure of the corruption of signals due to the third-order intermodulation of two nearby tones as shown in Figure [D-2](#). You measure this parameter using a two-tone test. Avoid the measurement of IP3 by a single tone test.

Figure D-2 Intermodulation of Two Nearby Signals

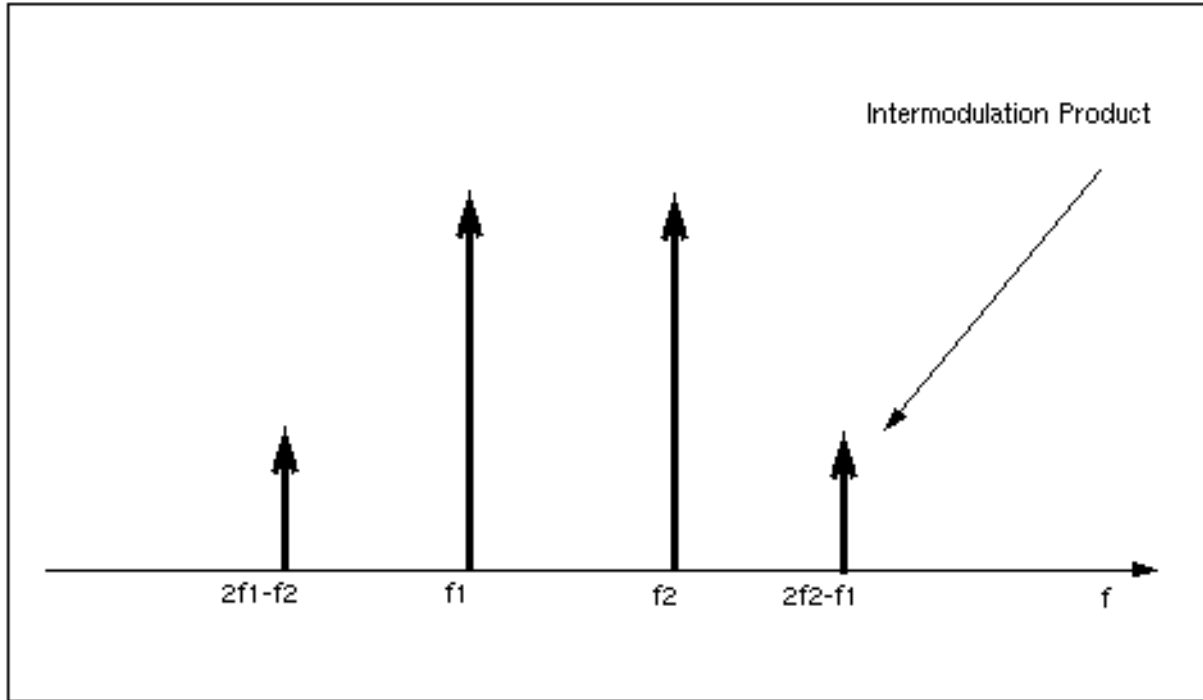
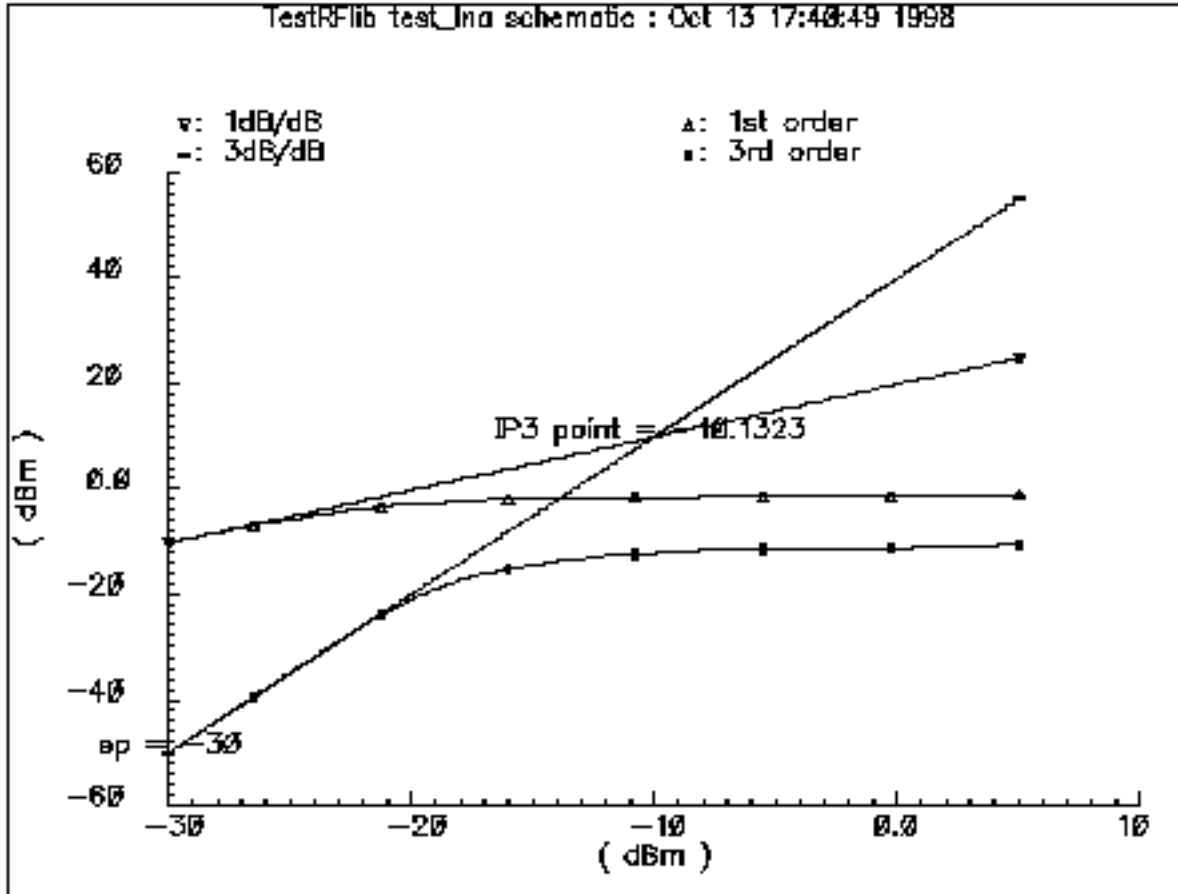
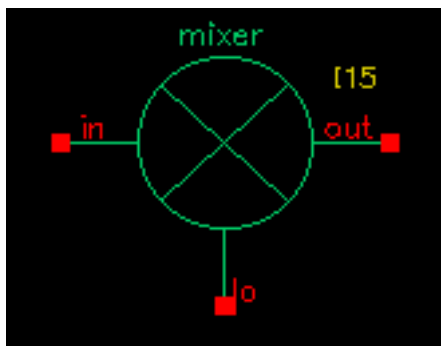


Figure D-2 shows the captured IP3 when the requested value of IP3 is  $-10\text{dBm}$ .

### IP3 from Spectre RF Simulation



### Mixer



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

Mixers are important for frequency translation in RF circuits. A typical mixer has the following three sets of parameters

- Time-varying linear model
- Nonlinear model
- Noise model

This RF library model describes the typical behavior of integrated mixers. The LO switches the input signal on and off. Input LO power beyond the specified limit is effectively clipped off.

Declare the module as follows

```
module mixer(in, lo, out);
    electrical in, lo, out;
    parameter real gain = 10 from [-50:50];
    parameter real plo = 10 from [-100:100];
    parameter real rin = 50 from (0:inf);
    parameter real rout = 200 from (0:inf);
    parameter real rlo = 50 from (0:inf);
    parameter real ip2 = 5;
    parameter real ip3 = 5;
    parameter real nf = 2 from [0:inf];
    parameter real isolation_LO2IN = 20 from (0:inf);
    parameter real isolation_LO2OUT = 20 from (0:inf);
    parameter real isolation_IN2OUT = 20 from (0:inf);
```

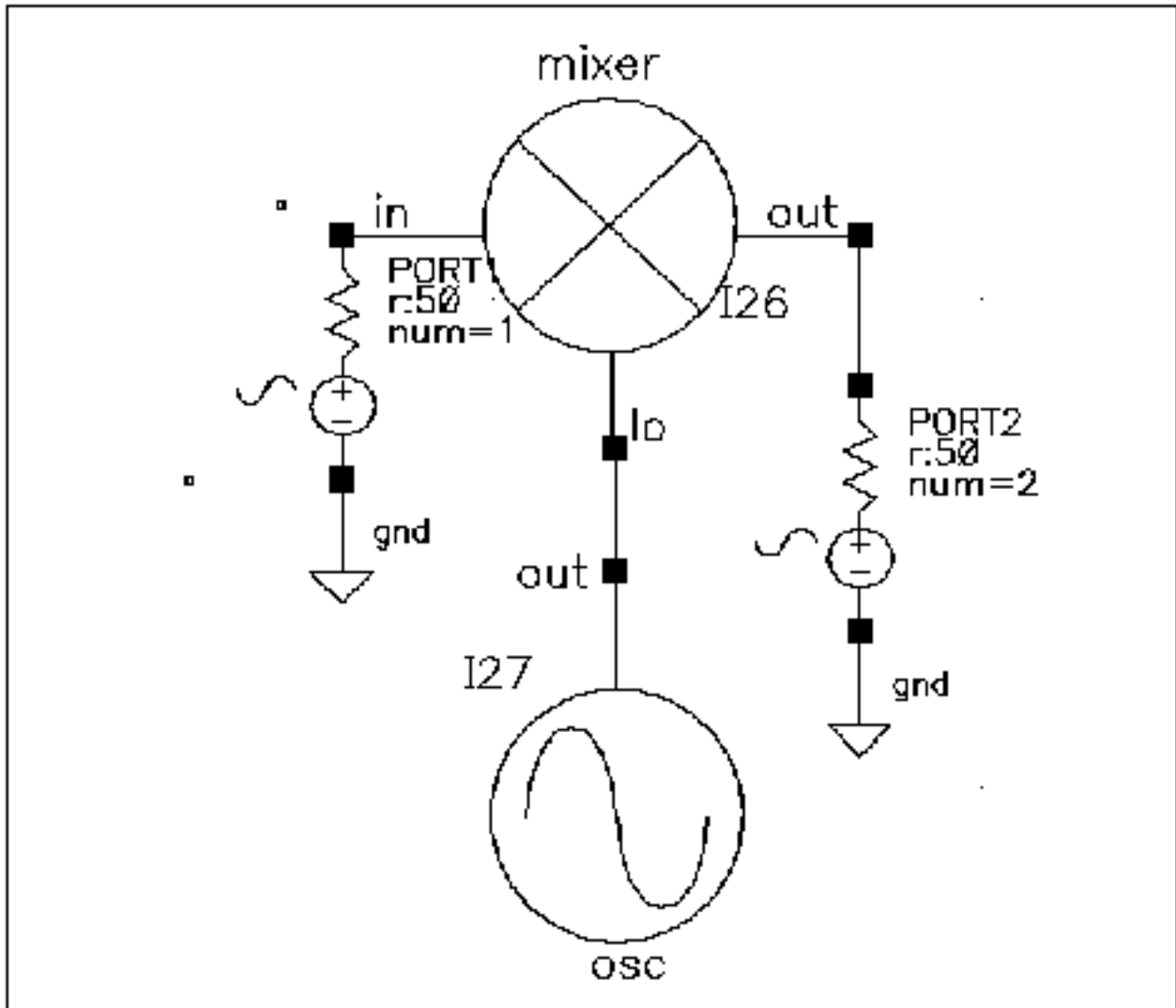
The parameters are

gain	Gain from IN to OUT [dB]
plo	Power of the LO input [dBm]
rin	Input impedance for IN [ $\Omega$ ]
rout	Output impedance for OUT [ $\Omega$ ]
rlo	Input impedance for LO [ $\Omega$ ]
ip2	Input referenced IP2 [dBm]
ip3	Input referenced IP3 [dBm]
nf	Noise figure (DSB) [dB]
isolation_LO2IN	Isolation from LO to IN [dB]
isolation_LO2OUT	Isolation from LO to OUT [dB]

isolation\_IN2OUT      Isolation from IN to OUT [dB]

Figure D-2 is the simple schematic that tests the mixer.

### Schematic for Testing the Mixer Model

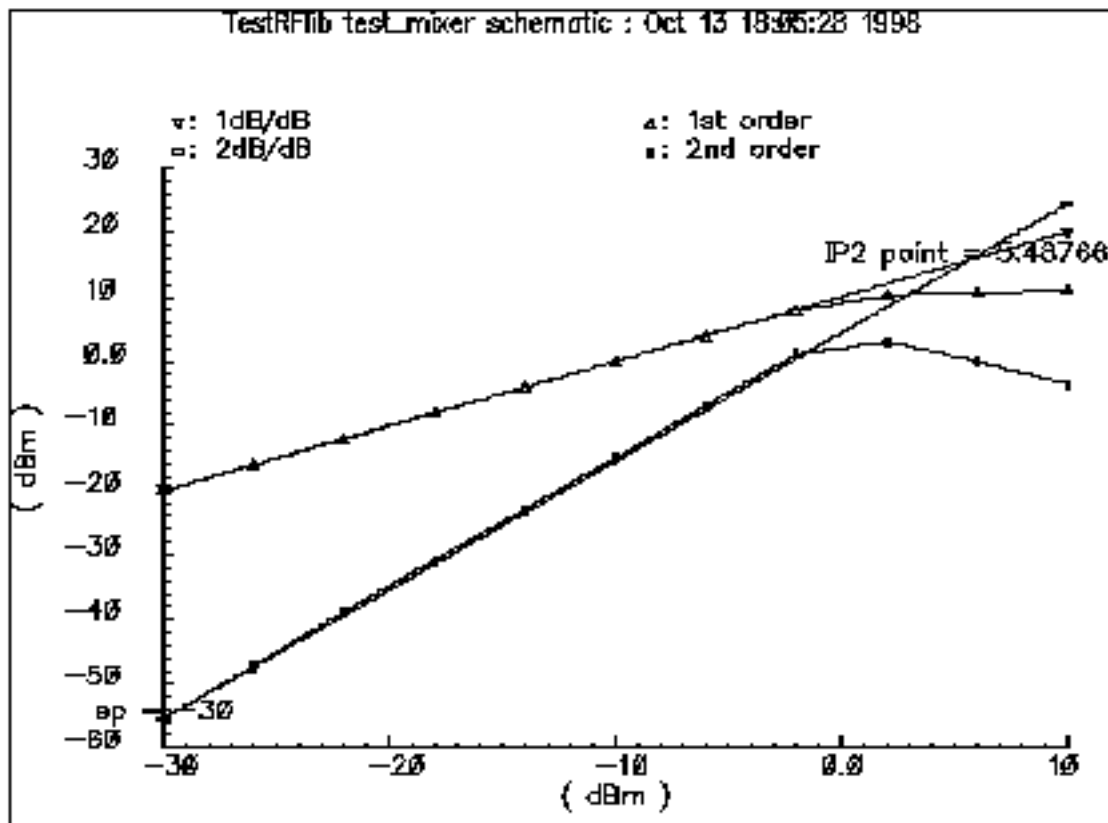


The maximum power of the fundamental frequency of the local oscillator,  $p_{lo}$ , can be used in the mixing process. Therefore, the gain, defined as the output power of the mixed product versus the input power of the RF signal, depends on the power level of the LO. The gain levels off, however, to the specified maximum value as the LO signal becomes larger.

You can measure both IP3 and IP2 with Spectre RF. You must select frequencies carefully when you measure IP3 to measure harmonic distortion (HD) and IP2. Testing IP3 requires two tones to measure the intermodulation distortion (IMD), while testing IP2 requires only one tone.

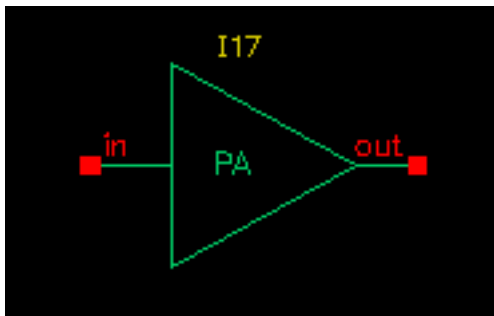
Assume the RF input frequencies are  $f_1$  and  $f_2$ , and the LO frequency is  $f_{i0}$ . If the input power level at  $f_1$  equals that at  $f_2$ , the IP3 is the intercept point of the extrapolated line of output power at frequency  $|f_{i0} - (2f_2 - f_1)|$  versus the extrapolated line of the linear output signal at  $|f_{i0} - f_1|$ . Input-referenced IP3, therefore, can be read as the X-axis value at the intercept point. The IP2, for the purpose of measuring the half-IF effects, is defined as the intercept point of the extrapolated line of output power at frequency  $|2(f_{i0} - f_1)|$  versus the linear output signal. Figure D-2 shows that the intercept point of the 1 dB/dB and 2 dB/dB lines is at the X-axis reading of 4.78 dBm, while the requested IP2 value is 5 dBm. The order of the intercept point is based only on the order of the RF signals. The order of LO signal is not counted in the definition of the intercept point. In the implementation of this model, the orders of LO for IP3 and IP2 are 1 and 2 respectively.

### IP2 Measurement



Internally, a set of equations is built to satisfy a three-port S-parameter. A third-order polynomial describes the nonlinearity of IP3. The LO signal is further multiplied by itself to derive the second-order harmonic, which is then used to produce the IP2 effect. Excessive white noise is added in the RF input port to satisfy the noise figure. Remember, however, that the noise figure is double-sideband. If the noise at the image frequency is not filtered out, the measured noise figure is 3dB larger than the DSB noise figure.

## Power Amplifier



Power amplifiers (PAs) are used in RF transmitters to achieve output of a higher power level. The PA model differs from the LNA model in that it has greater power delivery capabilities with less stress on matching capabilities.

The Verilog-A module is declared as follows:

```
module pa(in, out);
    inout in, out;
    electrical in, out;
    parameter real nf = 2 from [0:inf);
    parameter real gain = 20 from [0:inf);
    parameter real rin = 50 from (0:inf);
    parameter real rout = 50 from (0:inf);
    parameter real pldb = 30;
    parameter real psat = 35;
    parameter real ip2 = 40;
```

The parameters are

nf	Noise figure [dB]
gain	S21 [dB]
rin	Input impedance [ $\Omega$ ]
rout	Output impedance [ $\Omega$ ]

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

<code>psat</code>	Maximum output power [dBm]
<code>p1db</code>	Output-referenced 1dB compression [dBm]
<code>ip2</code>	Input-referenced IP2 [dBm]

The power amplifier model has the following three parts:

- the linear model
- the nonlinear model
- the noise model

Internally, for simplicity, the reverse isolation is assumed to be ideal. A set of linear equations is constructed to satisfy these S-parameters. Nonlinear effects are added to the gain to describe the nonlinearity. The output power of the power amplifier compresses to 1 dB less than the output of an ideal linear amplifier at the 1 dB compression point. Further increase of the input power makes the output approach the saturation power only at the fundamental operating frequency. IP2 describes the second order effects of the amplifier, so use only one tone in the test. Excess white noise is added at the input port to describe the noise figure.

The implementation of `psat` assumes a pure sinusoidal waveform. To maintain a restrained output power, the output waveform is clipped from a sinusoidal to a square wave form. Figure [D-2](#) shows the input and output waveforms of the power amplifier. Because of the output waveform clipping, the input sinusoidal wave should have a DC component of zero.



### Input and Output Waveforms of the Power Amplifier

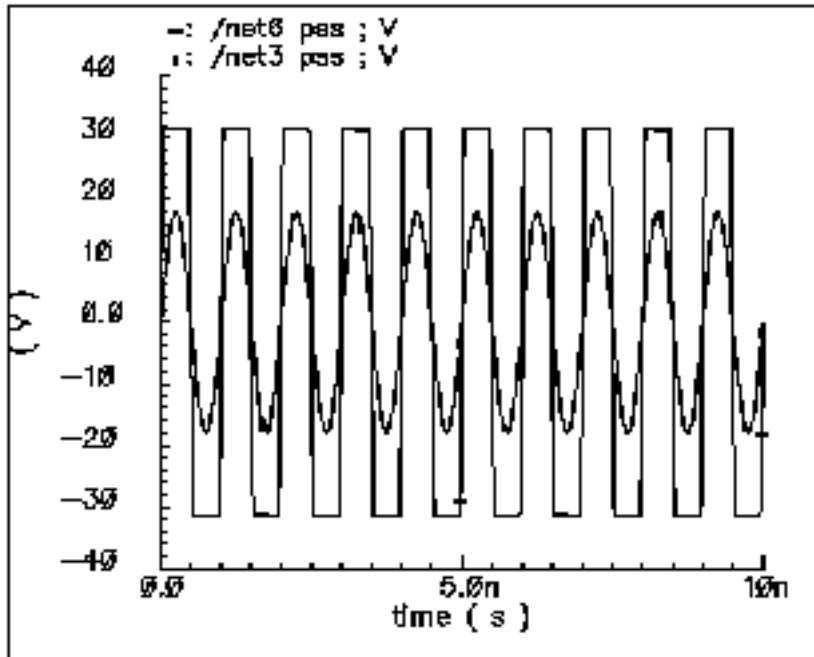
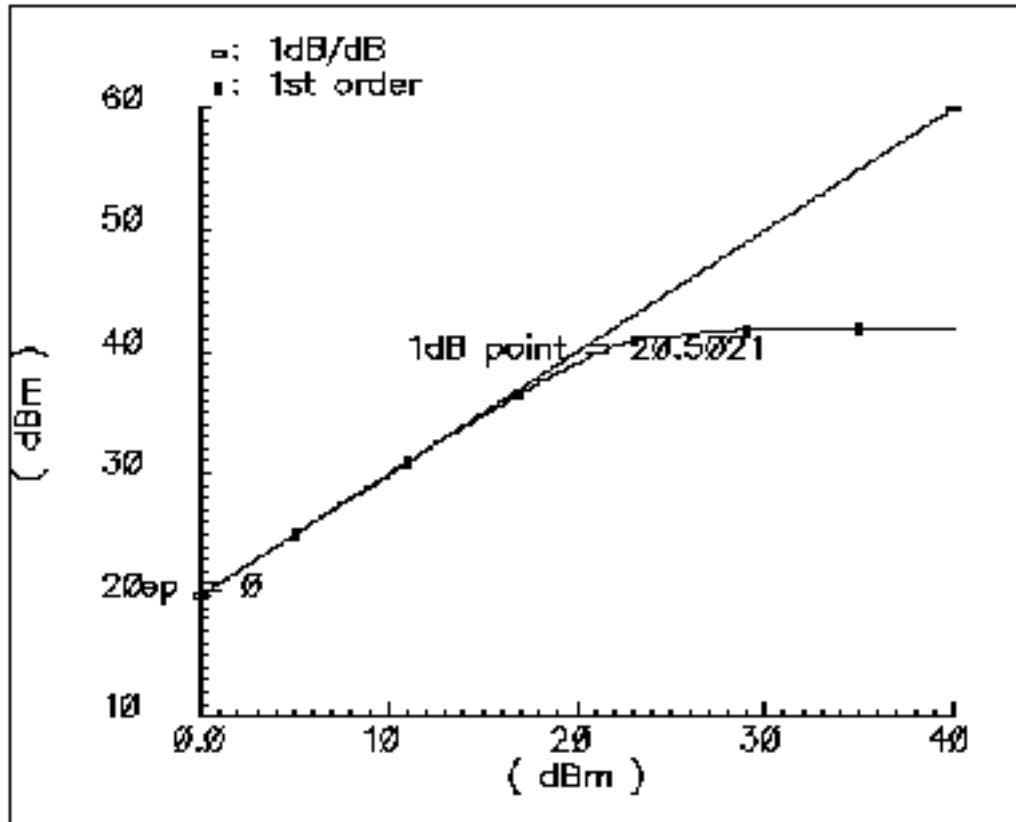


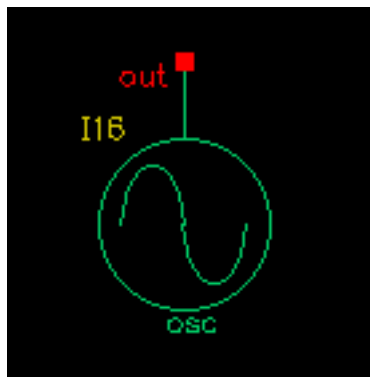
Figure D-3 shows the 1 dB compression point and the saturation power. This difference is caused by the 50  $\Omega$  load impedance. The specified output referenced 1 dB compression point is 40 dBm, which Spectre RF captures as 39.6.

If  $p_{sat}$  is much larger than  $p_{1db}$ , your  $p_{sat}$  might not be satisfied.

Figure D-3 1dB Compression Point and Saturation Power



## Oscillator



Oscillator models describe the essential information for a typical oscillator, more precisely, a local RF power source.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

The definition of the model in the Verilog-A language is as follows:

```
module osc(out);
    electrical out;
    inout out;
    parameter real power = 10;
    parameter real f = 1e9 from (0:inf);
    parameter real rout = 50 from (0:inf);
    parameter real floor = -60 from (-inf:0);
    parameter real f1 = 1000 from (0:1e6);
    parameter real n1 = -40 from (bottom:0);
    parameter real fc = 0 from [0:f1];
```

The parameters are

power	Output power when matched [dBm]
freq	Output frequency [Hz]
rout	Output impedance [W]
bottom	Noise floor [dBc/Hz]
f1	Frequency point for n1 [Hz]
n1	Phase noise at f1 [dBc/Hz]
fc	Corner frequency of white phase and flicker phase [Hz]

This model is not an autonomous model. Rather, it simply generates a sinusoidal wave with the specified impedance, power level, and phase noise characteristics.

When the load is matched to the internal impedance, the load dissipates the specified output power. You can specify the noise floor of the output signal. Furthermore, by adding one point (frequency, phase noise), you can specify  $1/f^2$  frequency noise (corresponding to the phase noise induced by white noise). If  $f_c$ , the corner frequency of white phase and flicker phase noise, is bigger than 0,  $1/f^3$  frequency noise (flicker-noise-induced phase noise) is further specified. Otherwise,  $1/f^3$  noise is not included.

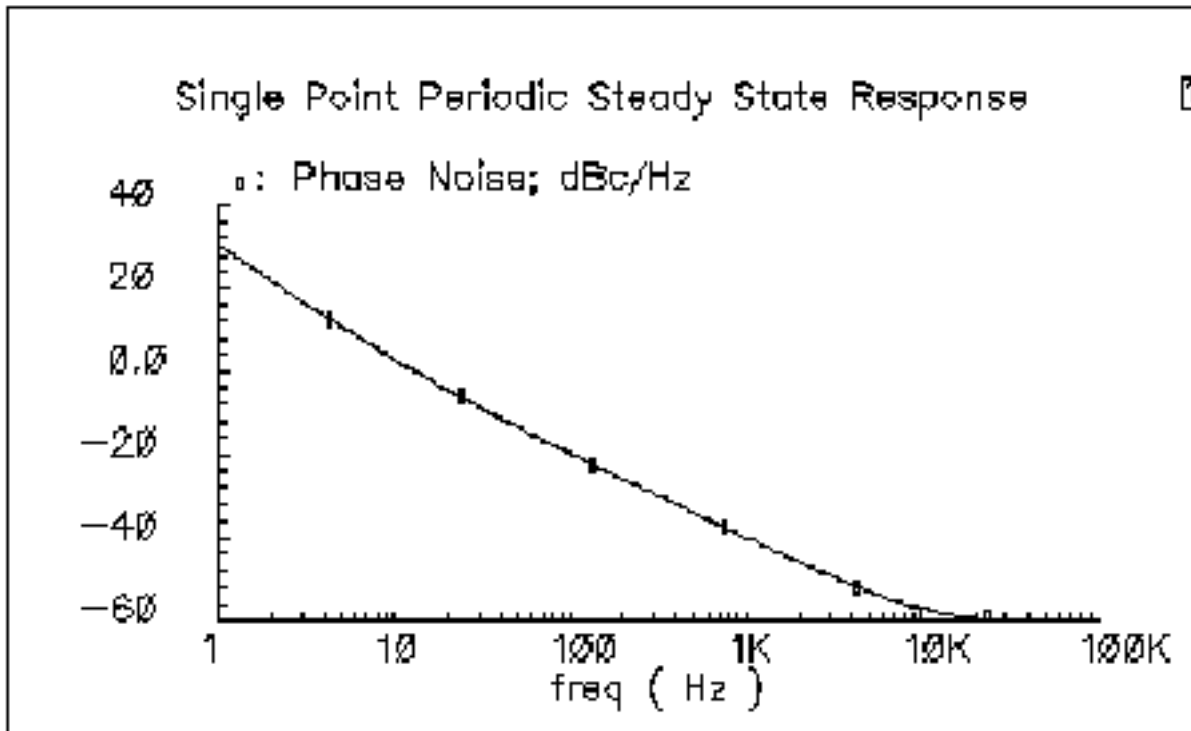
The phase noise values that are symmetric around the carrier are correlated. The noise floor, however, is not correlated.

Figure [D-4](#) shows the phase noise of the oscillator model. In Figure [D-4](#), the specified parameters are

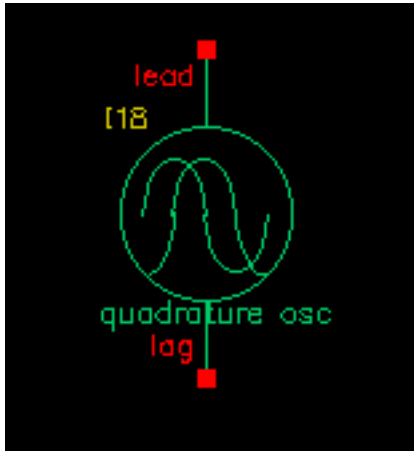
```
noise floor -60 dBc/Hz
```

$f_1$  1 K  
 $n_1$  -40 dBc/Hz  
 $f_c$  100

Figure D-4 Phase Noise for the Oscillator



## Quadrature Signal Generator



The quadrature signal generator model is included because, in quadrature receiver design, a phase shifter is ordinarily used to generate the quadrature signal from one signal source such as the VCO. However, a phase shifter is hard to implement in a wide band model.

A quadrature signal consists of two signals with a 90-degree phase difference but with identical noise and amplitude.

The Verilog-A module is declared as follows

```
module quadrature(lead, lag);
    electrical lead, lag;
    inout out_cos, out_sin;
    parameter real power = 10;
    parameter real f = 1e9 from (0:inf);
    parameter real rout = 50 from (0:inf);
    parameter real floor = -60 from (-inf:0);
    parameter real f1 = 1000 from (0:1e6);
    parameter real n1 = -40 from (bottom:0);
    parameter real fc = 0 from [0:f1];
```

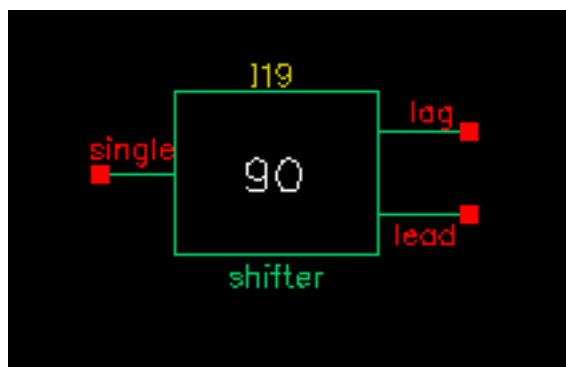
The parameters are

power	Output power when matched [dBm]
freq	Output frequency [Hz]
rout	Output impedance [W]
phase_shift	

bottom	Noise floor [dBc/Hz]
f1	Frequency point for n1 [Hz]
n1	Phase noise at f1 [dBc/Hz]
fc	Corner frequency of white phase and flicker phase [Hz]

The difference between the quadrature signal generator model and the oscillator model is that the oscillator has only one output node but the quadrature signal generator has two output nodes, `lead` and `lag`. In the quadrature signal generator model, when the power levels, output impedances, and noise sources are identical, the two outputs, `lead` and `lag`, have a 90-degree phase difference.

### Phase Shifter



In digital RF system designs, quadrature signal processing involves the phase splitting of high-frequency signals. The most common use of such components is to generate two signals that have a 90-degree phase difference based on one signal source (such as the RF signal or oscillator output). Another common use for a phase shifter is to combine two signals after adding a 90-degree phase difference, as in image-rejection receiver designs.

The Verilog-A module is declared as follows

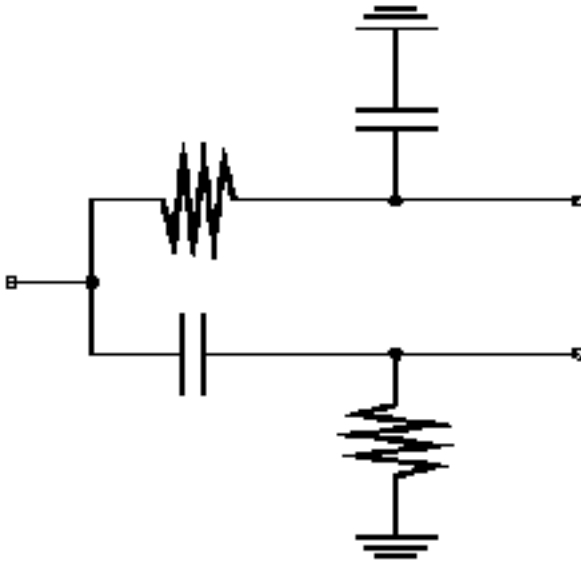
```
module shifter(single, lag, lead);
    inout single, lag, lead;
    electrical single, lag, lead;
    parameter real freq = 1e9 from (0:inf);
    parameter real r      = 50 from (0:inf);
endmodule
```

The parameters are

freq	Frequency of operation [Hz]
r	Resistance [ $\Omega$ ] (see Figure D-4)

Internally, the phase shifter is implemented using the RC-CR circuit as shown in Figure D-4. While the phase difference is also 90-degrees when the `lead` and `lag` have the same output impedance, only at the operating frequency do the magnitudes remain the same. This circuit network also generates white noise.

### Phase Shifter



There are two buffered versions of the shifter:

- The `shifter_combiner` combines two signals so that they add if one leads the other by 90 degrees and so that they cancel if it lags by 90 degrees.
- The `shifter_splitter` splits a signal into two signals 90 degrees out of phase with each other.
- You specify the input and output impedances. These networks are noiseless.

## Measurement Elements

The *Measurement* category contains elements used to facilitate measurements and diagnostics. It includes the instrumentation blocks and the baseband signal generators. These models are not part of any specific RF design architecture.

This section describes the *Measurement* elements and explains how to change the FIR filters inside the baseband signal generators.

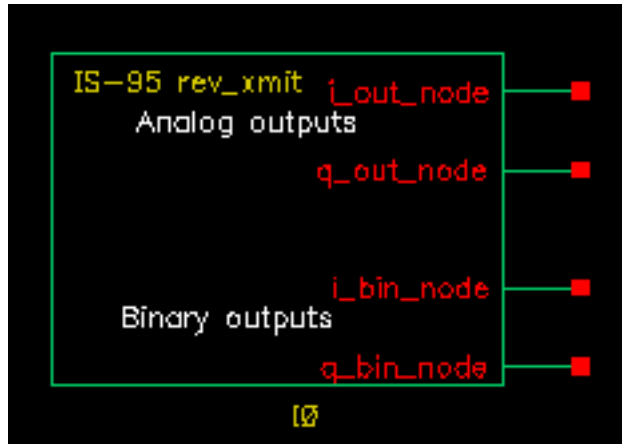
**Note:** All of the baseband signal sources generate digitally filtered signals. The baseband sources do not work with Spectre RF because the digital filters have hidden states.

The *Measurement* category contains the following elements, discussed in the sections that follow.

- [CDMA Signal Source](#)
- [GSM Signal Source](#)
- [\$\pi/4\$ -DQPSK Signal Source](#)
- [Eye-Diagram Generator](#)
- [Rectangular-to-Polar Coordinate Transformation](#)
- [Polar-to-Rectangular Coordinate Transformation](#)
- [Ideal Transformer](#)
- [Instrumentation Block](#)
- [Offset Instrumentation Block](#)
- [Instrumentation Terminator](#)
- [Baseband Driver](#)
- [Phase Generator](#)



## CDMA Signal Source (CDMA\_reverse\_xmit)



The CDMA signal source (`CDMA_reverse_xmit`) generates a reverse-link (handset-to-base-station) IS-95 signal with the following characteristics

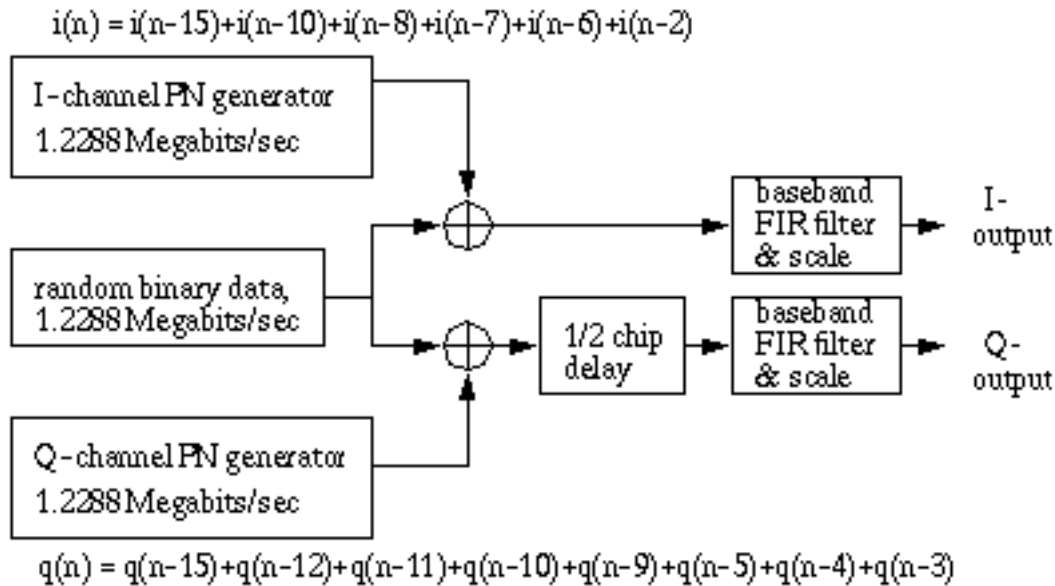
modulation	Offset QPSK
symbol rate	1.2288 megasymbols per second
sample rate	4.9152 megasamples per second

Two separate 16-bit pseudo-noise generators generate the *I* and *Q* spreading sequences operating at the sample rate.

The CDMA source

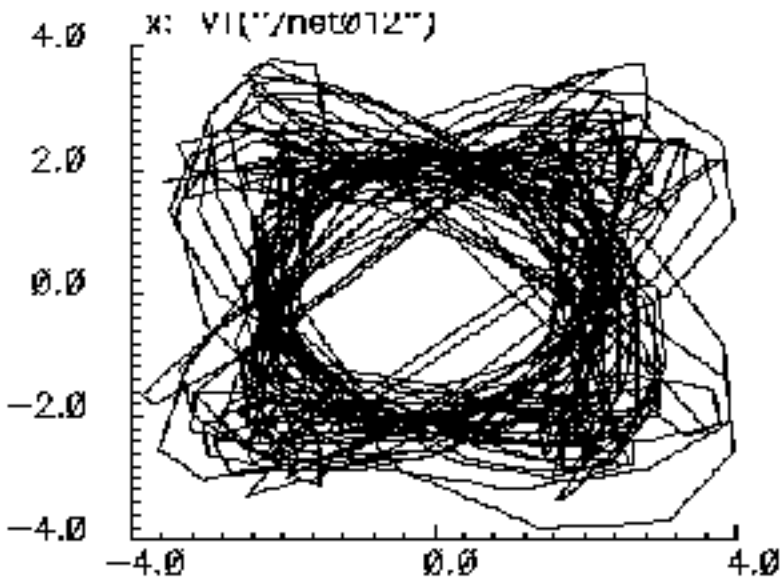
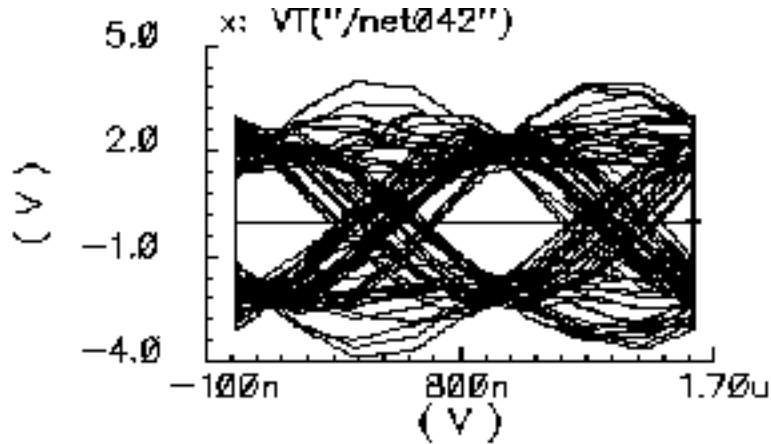
- Generates a random bit at the symbol rate
- Oversamples it by a factor of 4
- Spreads the bit with the *I* and *Q* spreading sequences
- Filters each sequence with a 48-tap FIR filter. The filter coefficients are the impulse response of a raised cosine filter.
- The CDMA signal source generates a reverse-link (handset-to-base-station) IS-95 signal. The modulation is offset QPSK with a symbol rate of 1.2288 Mega-symbols per second and a sample rate of 4.9152 Megasamples per second. Two separate 16-bit pseudo-noise generators generate the *I* and *Q* spreading sequences operating at the sample rate.
- Figure [D-5](#) shows a block diagram of the signal generator.

Figure D-5 CDMA Baseband Test Signal Generator



The eye-diagram generator (`eye_diagram_generator`) created the eye-diagram and trajectory. Figure D-5 shows the eye-diagram of one of the outputs and the trajectory of both outputs.

### Eye Diagram and CDMA Trajectory



### Instance Parameters

The *amplitude* parameter sets the amplitude of the unfiltered signals. An amplitude of 1 means that each FIR filter is driven by 1 volt impulses. If you change the internal variable `IMPULSE_PULSE` to 2, the filters are driven by 1 volt pulses of four samples duration.

The *seed* parameter changes the seed for the random number generator.

## Outputs

The CDMA signal generator creates four output signals:

<i>i_bin_node</i>	The <i>I</i> unfiltered binary output.
<i>i_out_node</i>	The filtered <i>I</i> output.
<i>q_bin_node</i>	The <i>Q</i> unfiltered binary output.
<i>q_out_node</i>	The <i>Q</i> filtered output.

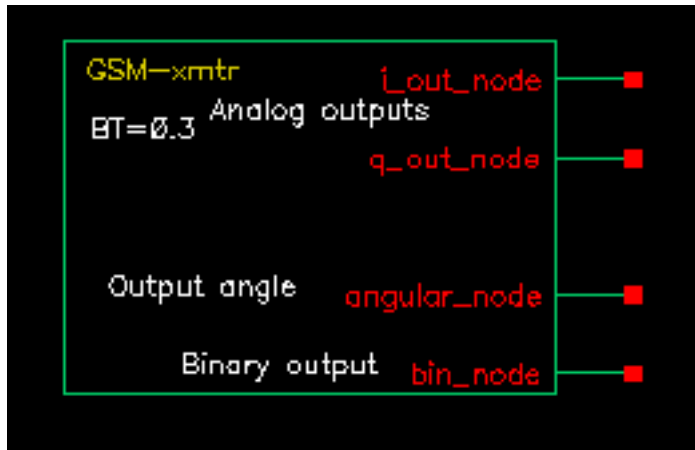
## Changing the FIR Filter

You cannot change the FIR filter, such as the tap length and tap coefficients, directly from the instance. However, you can do so using the Modelwriter as described in [“Modifying the Baseband Signal Generators Using the Modelwriter”](#) on page 782.

## Output Transitions

The filtered outputs slew linearly from one value to the next because the rise and fall times in the transition statements equal one period. To make the outputs take abrupt steps, copy the module to your library and change the rise and fall times in the last transition statements.

## GSM Signal Source (GSM\_xmtr)



The GSM source generates a signal conforming to the GSM standard. The modulation is GMSK and the data is generated in frames of 3 fixed start bits, 142 random data bits, 3 fixed stop bits, and 8.25 fixed guard bits. (The embedded deterministic pattern and quarter of a bit is necessary to produce the correct spectrum.) The bit rate is 270833.333 bits per second and the sample rate is four times that.

The FIR filter is a Gaussian filter implemented with 32 taps.

Figure [D-6](#) shows a block diagram of the signal source.

**Figure D-6 GSM Baseband Signal Generator**

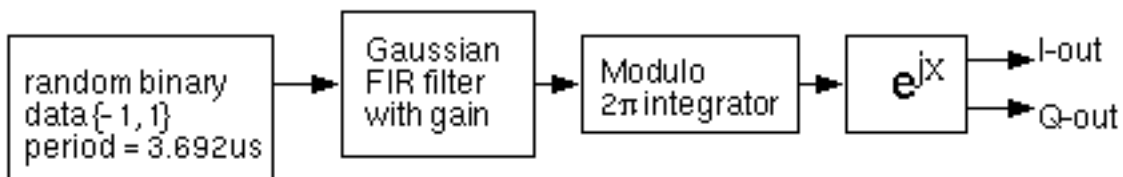
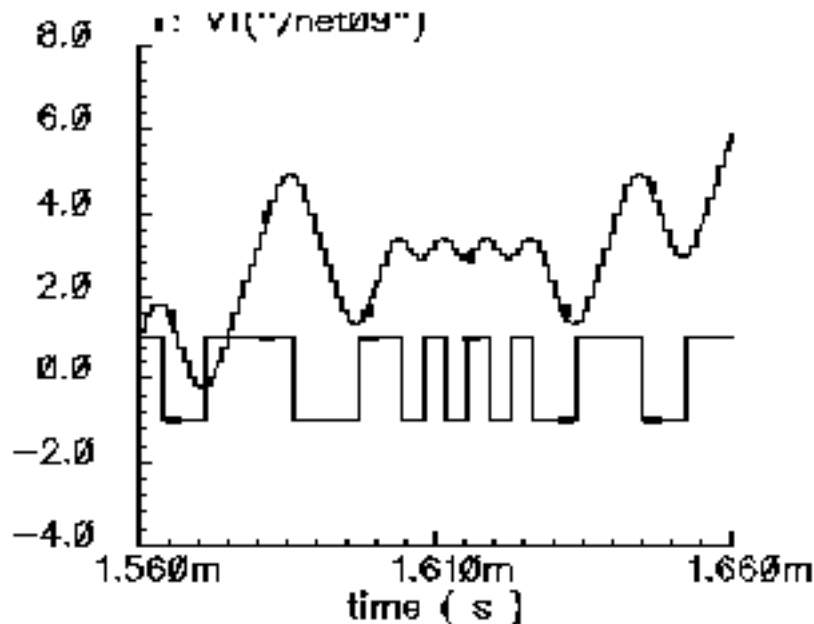


Figure [D-7](#) shows the binary data stream and the corresponding angle.

Figure D-7 GSM Binary Data and Resulting Phase



### Instance Parameters

The *amplitude* parameter sets the amplitude of the unfiltered signals. An amplitude of 1 means that each FIR filter is driven by 1-volt impulses. If you change the internal variable `IMPULSE_PULSE` to 2, the filters are driven by 1-volt pulses of four samples duration.

The *seed* parameter changes the seed for the random number generator.

### Outputs

The generator creates four output signals

<i>angular_node</i>	The output signal.
<i>i_out_node</i>	The phase, multiplied by the amplitude.
<i>bin_node</i>	The bit stream being transmitted.
<i>q_out_node</i>	The phase multiplied by the amplitude.

## **Changing the FIR Filter**

You cannot directly change the FIR filter, such as the tap length and tap coefficients, from the instance. However, you can make changes using the Modelwriter as described in [“Modifying the Baseband Signal Generators Using the Modelwriter”](#) on page 782.

## **Output Transitions**

The filtered outputs slew linearly from one value to the next because the rise and fall times in the transition statements equal one period. To make the outputs take abrupt steps, copy the module to your library and change the rise and fall times in the last transition statements.

## Pi/4-DQPSK Signal Source (pi\_over4\_dqpsk)

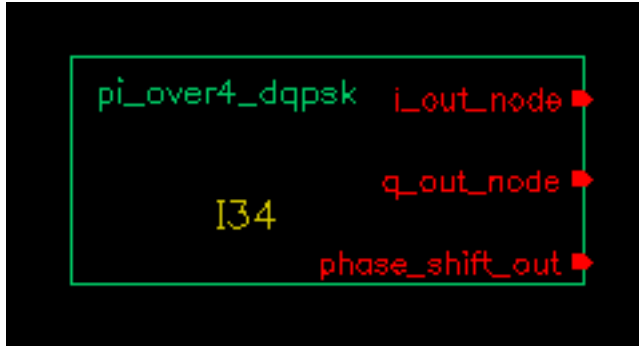


Figure D-8 shows the block diagram for this source.

**Figure D-8**  $\Pi/4$ -DQPSK baseband signal generator

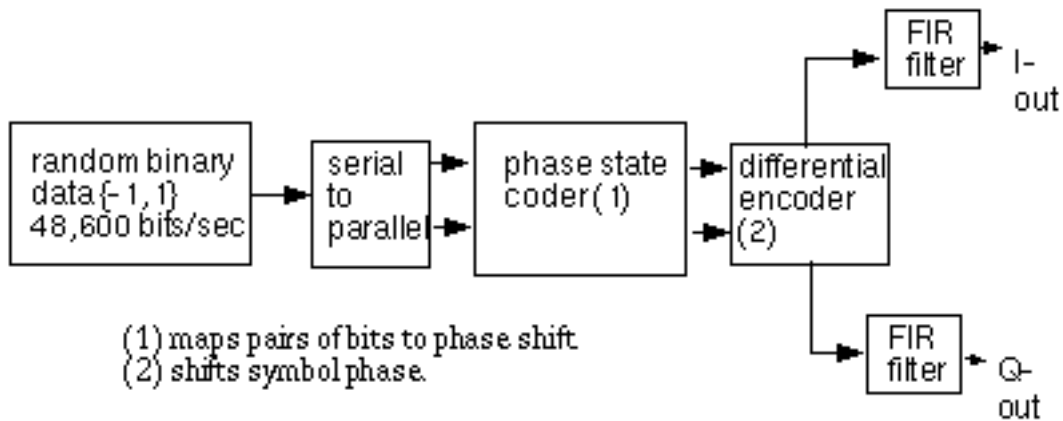


Table D-8 shows how the phase shift is generated.

### Phase Shift

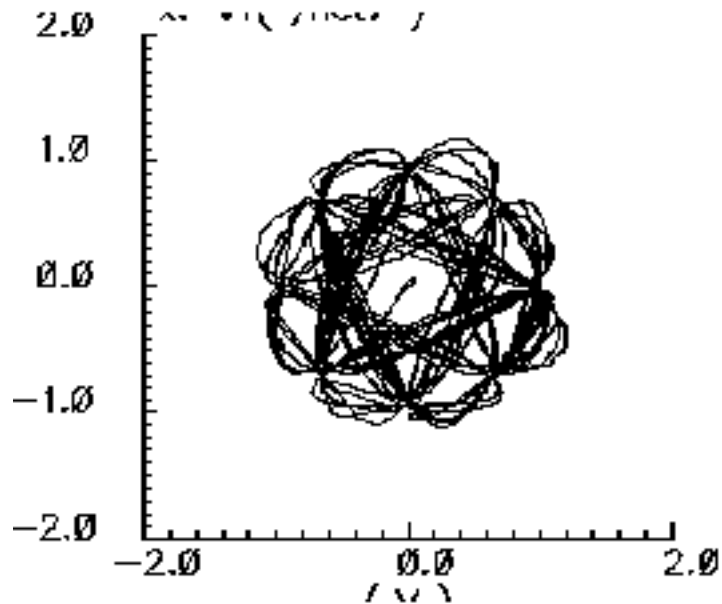
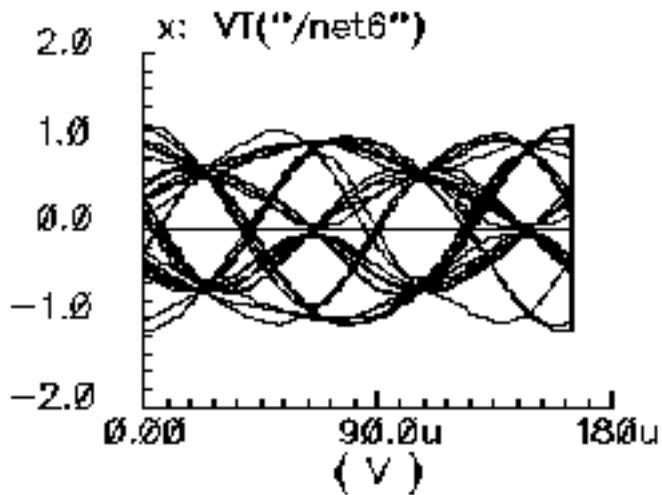
1st bit	2nd bit	Phase shift
0	0	$\pi/4$
0	1	$3\pi/4$
1	0	$-\pi/4$
1	1	$-3\pi/4$



The symbol rate is 24300 symbols per second and the sample rate is 8 times that. The FIR filter is a raised cosine filter implemented with 64-taps.

The eye-diagram generator (`eye_diagram_generator`) created the eye-diagram and trajectory. Figure D-8 shows the eye-diagram and trajectory for this generator.

### Eye Diagram and Pi/4 Trajectory



The *amplitude* parameter lets you set the amplitude of the unfiltered signals. An amplitude of “1” means that each FIR filter is driven by 1-volt impulses. If you change the internal variable `IMPULSE_PULSE` to 2, the filters are driven by 1-volt pulses of four samples duration.

The *seed* parameter lets you change the random number generator seed.

## Outputs

The generator creates three output signals.

<i>i_out_node</i>	The phase, multiplied by the amplitude
<i>q_out_node</i>	The phase, multiplied by the amplitude
<i>phase_shift_out</i>	The phase shift from one symbol to the next

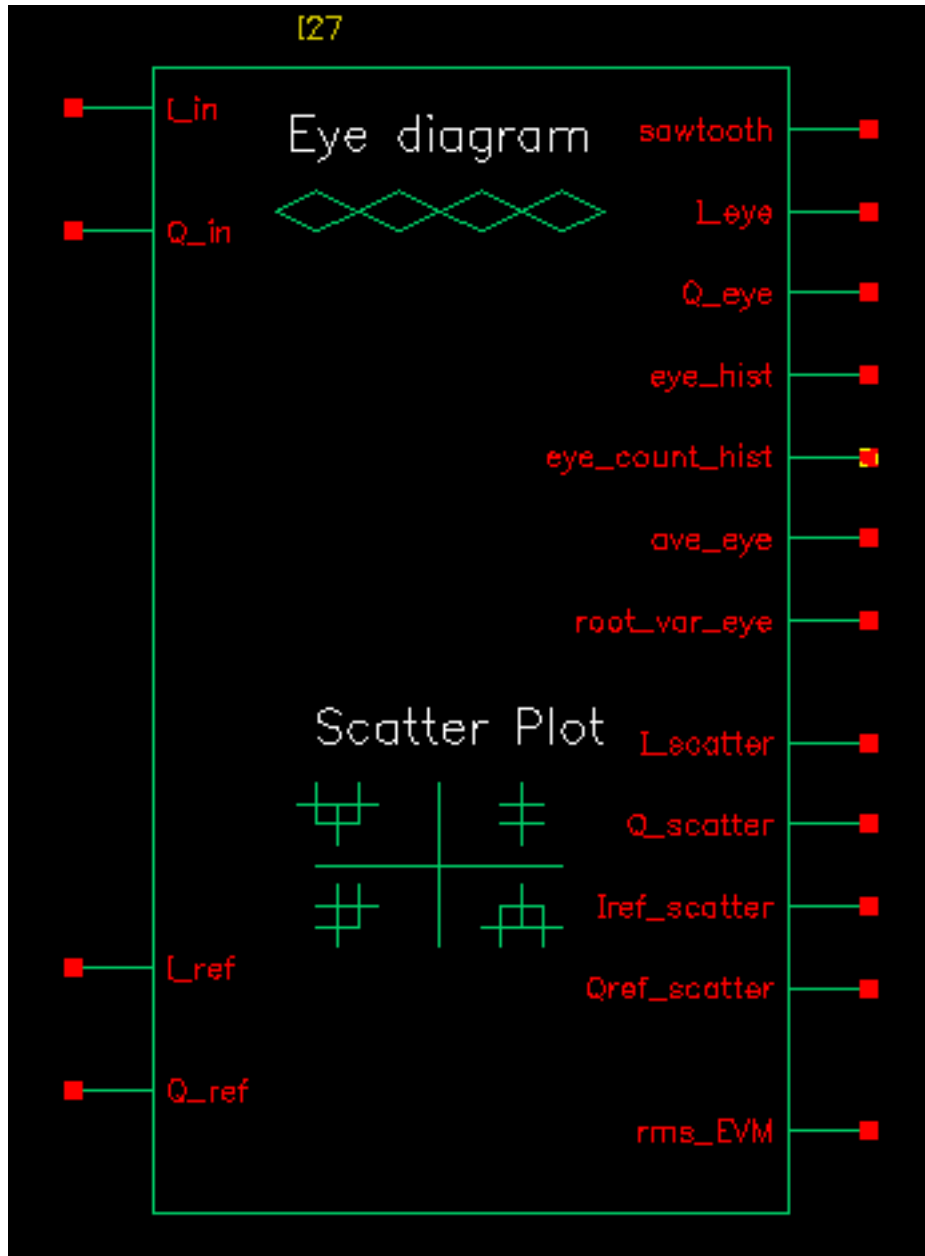
## Changing the FIR filter

You cannot change the FIR filter, such as the tap length and tap coefficients, directly from the instance. However, you can do so using the Modelwriter as described in [“Modifying the Baseband Signal Generators Using the Modelwriter”](#) on page 782.

### Output transitions

The filtered outputs slew linearly from one value to the next because the rise and fall times in the transition statements equal one period. To make the outputs take abrupt steps, copy the module to your library and change the rise and fall times in the last transition statements.

### Eye-Diagram Generator (eye\_diagram\_generator)



The eye-diagram generator creates eye-diagrams and trajectories for the baseband signal generators.

Input

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

The input to the eye-diagram generator is the *I* or *Q* output of one of the baseband signal generators.

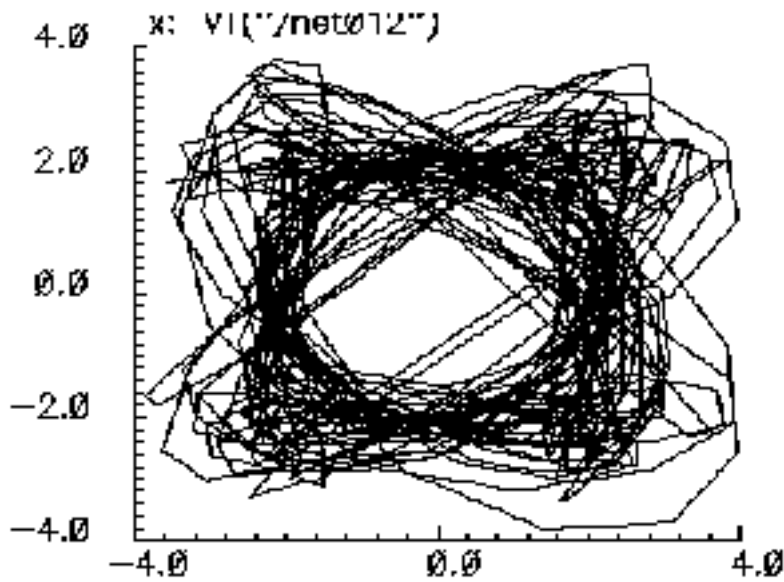
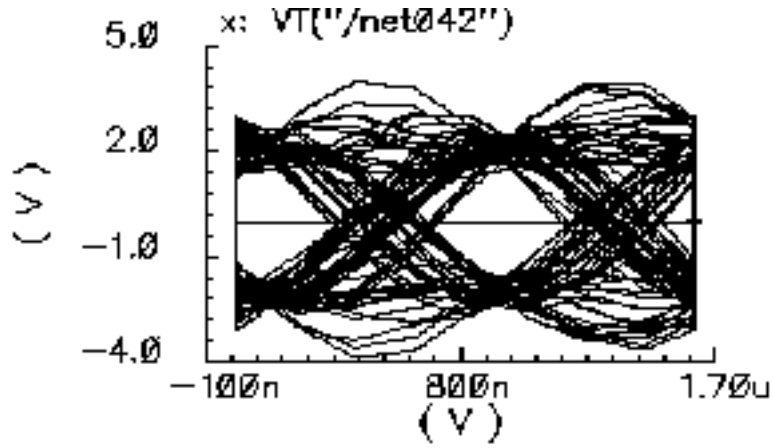
#### Output

The eye-diagram generator has two outputs labeled “y-axis” and “x-axis”. The eye-diagram is generated by plotting the “y-axis” output against the “x-axis” output.

The eye-diagram generator does not work with Envelope analysis to generate similar plots. This is because the Envelope harmonic time analysis is generated by a post-processing step and the eye-diagram generator works during simulation.

Figure [D-8](#) shows an eye-diagram of one of the outputs and the trajectory of both outputs for the CDMA baseband signal generator.

### Example Eye Diagram and CDMA Trajectory

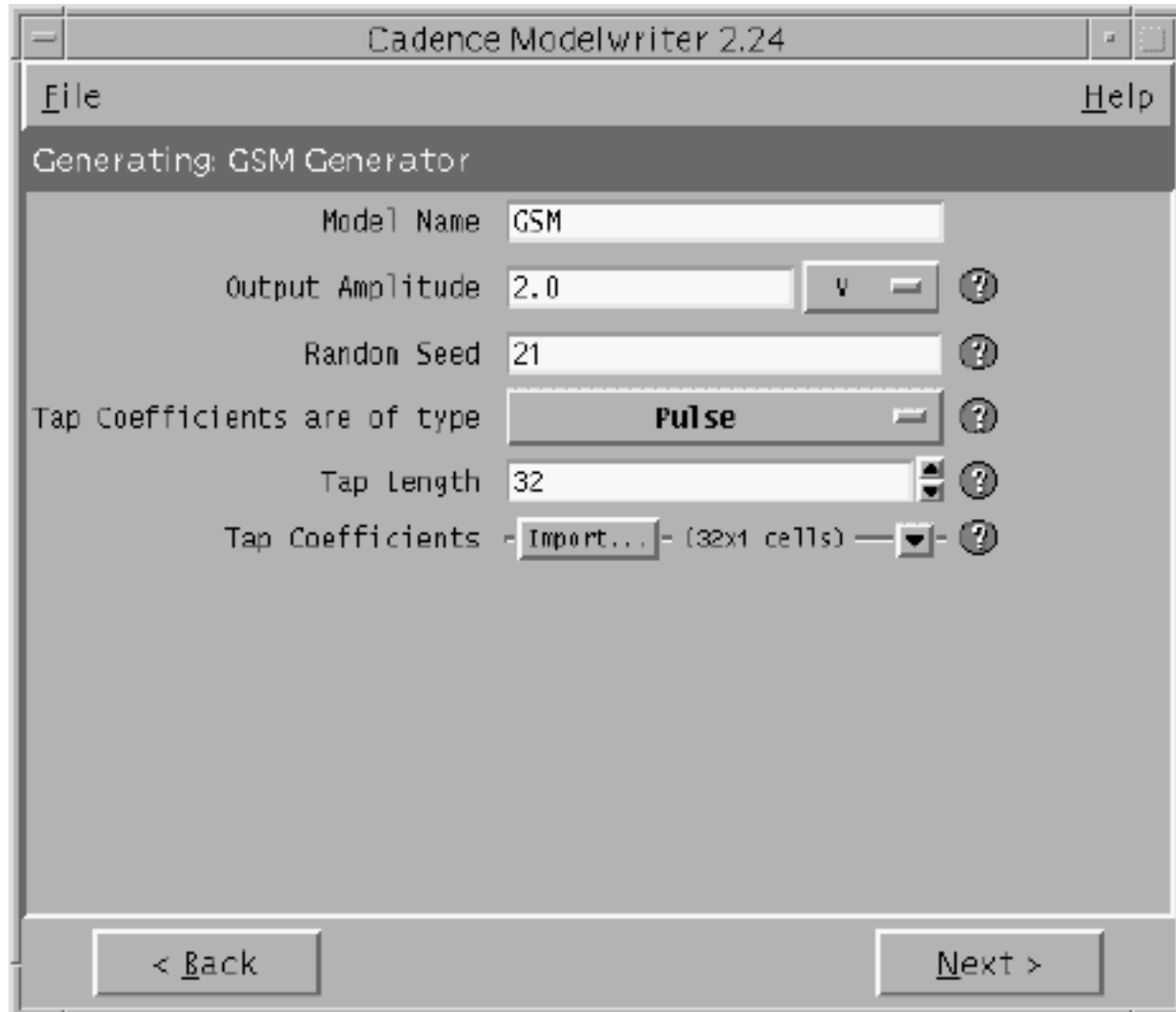


## **Modifying the Baseband Signal Generators Using the Modelwriter**

The baseband signal generators use FIR (finite impulse response) filters to shape their output pulses. Shaped output pulses serve several purposes:

- They help keep the transmitted signal inside the specified band.
- They work with their receive counterparts to maximize the signal-to-noise ratio.
- Together with their receiver counterparts, they satisfy the Nyquist sampling criterion for an intersymbol-interference-free channel.
- The Modelwriter gives you a convenient user-interface for creating variations on the baseband signal generators in the library. The most likely variation is in the FIR filter. This section explains how to use the Modelwriter to create a new GSM generator with different FIR filters.
- Bring up the Modelwriter and do the following:
  - a. Double click the Telecom folder.
  - b. Select the GSM generator.
  - c. Click the *Next* button in the lower right hand corner of the Modelwriter window.
  - d. You should now see the picture in Figure [D-8](#) in the window.

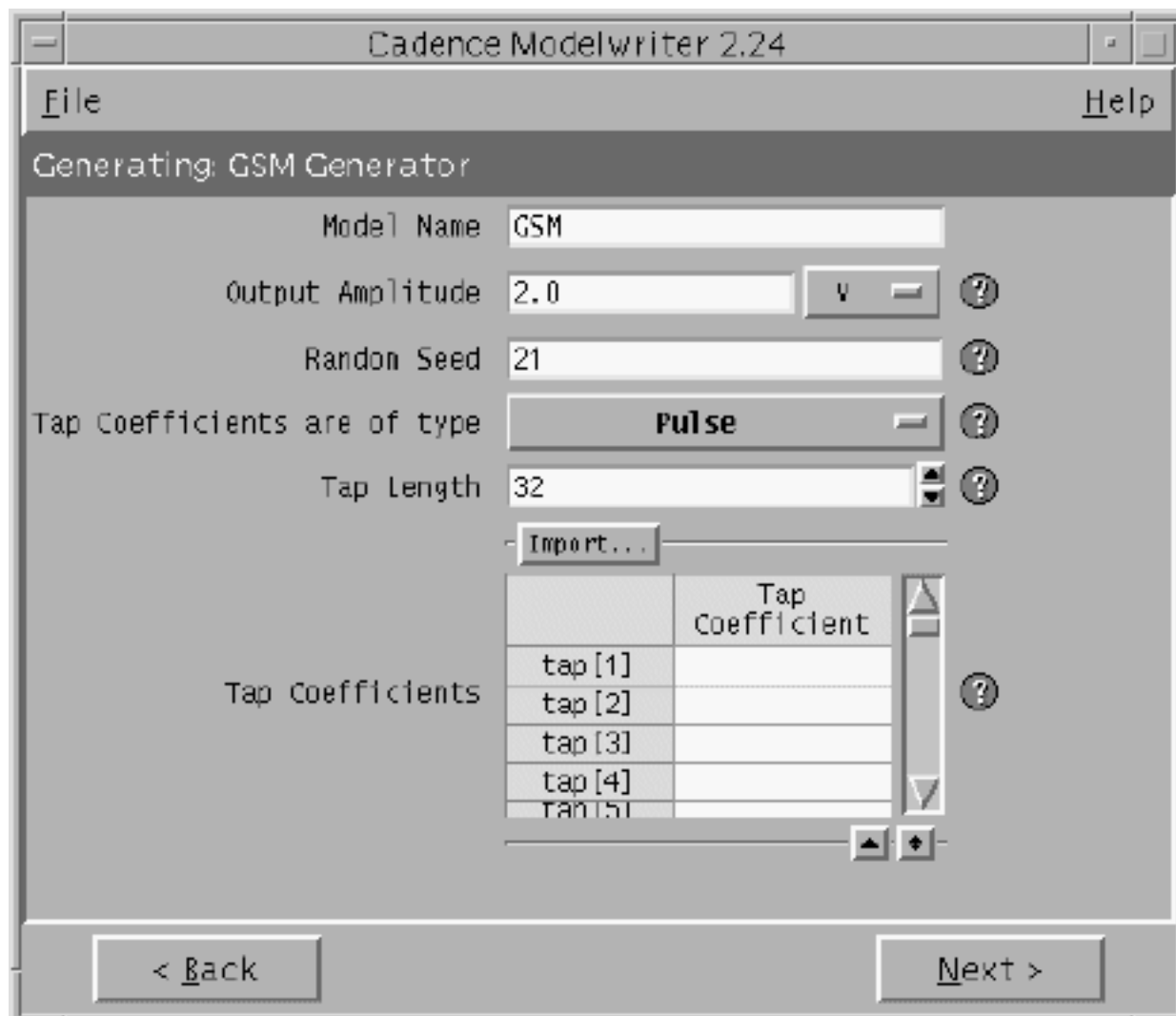
## GSM Generator



1. Specify how you plan to drive the filter by selecting the type of tap coefficients.
2. FIR filters can be driven by pulses or impulses.
3. Specify the length of the FIR filter in the *Tap Length* field.
4. You can specify the tap coefficients in two ways
  - From a file
  - By direct manual entry.
  - To read the coefficients from a file do the following:

- e. Select the *Import* button.
- f. Enter the path to the file.
- g. Click *Open*.
- h. The coefficients appear in the window as shown in Figure D-9. (The tap[1] coefficient multiplies the filter state with the least amount of delay, the filter state closest to the input.)

Figure D-9 Tap coefficients

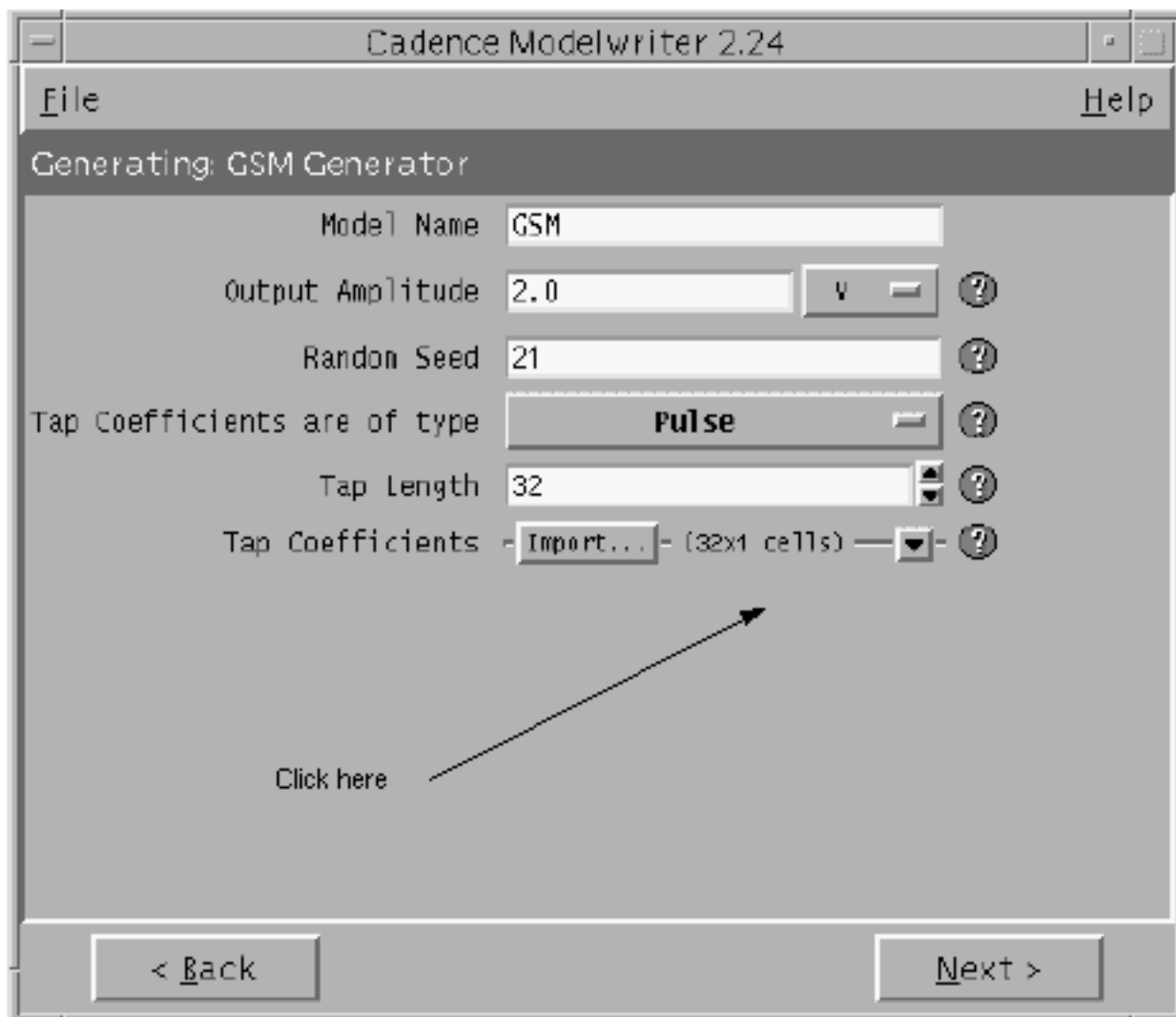


To enter the coefficients manually do the following:



- a. Click the lower rightmost button flagged in the form.  
See Figure D-9.
- b. Enter the coefficients,
- c. Click *next* to view the model,
- d. To write the model to a file, click *Save Generated Code....*

### Manual entry of the tap coefficients



## Testbenches Elements

The *testbenches* category contains the test circuits used to define model specifications. Where possible, the element names are in terms of standard RF measurements. The most precise way to describe a measurement is with a test circuit, set up instructions, and sample measurements. The circuits in the testbench category serve this purpose. Table [D-1](#) lists the *testbenches* elements and includes a link to where each one is used.

**Table D-1 Elements in the Testbenches Category of rfLib**

<b>Element Name</b>	<b>Example Where the Element is Used</b>
<i>AM_PM_test_ckt</i>	<a href="#">“AM/PM Conversion Parameters”</a> on page 808
<i>BB_ind_cap_test</i>	<a href="#">“RLC Test Circuits”</a> on page 845
<i>PB_BB_filter_comparison</i>	<a href="#">“Comparison of Baseband and Passband Models”</a> on page 873
<i>PB_ind_cap_test</i>	<a href="#">“RLC Test Circuits”</a> on page 845
<i>ava_pwr_gain</i>	<a href="#">“Available Power Gain Parameter”</a> on page 794
<i>demod_ip3</i>	<a href="#">“IQ Demodulator”</a> on page 828
<i>dwn_cnvrt_test</i>	<a href="#">“RF-to-IF and IF-to-RF Mixers”</a> on page 836
<i>mixer_ip3</i>	<a href="#">“IP3 Parameter”</a> on page 801
<i>mod_1dbcp</i>	<a href="#">“Available Power Gain and 1dB Compression Point”</a> on page 823
<i>mod_demod_test</i>	<a href="#">“IQ Demodulator”</a> on page 828
<i>noise_figure</i>	<a href="#">“Noise Figure Parameter”</a> on page 799
<i>one_db_cp</i>	<a href="#">“Output 1dB Compression Point Parameter”</a> on page 798
<i>quad_and_phase_error_demo</i>	<a href="#">“Quadrature Error and Phase Error”</a> on page 825
<i>shifter_combiner_test</i>	<a href="#">“Phase Shifter Combiner”</a> on page 865
<i>shifter_splitter_test</i>	<a href="#">“Phase Shifter Splitter”</a> on page 861
<i>up_cnvrt_test</i>	<a href="#">“Testing the up_cnvrt Mixer”</a> on page 840
<i>view_switching</i>	

## Uncategorized Elements

Elements in the *uncategorized* category are data files created by elements in the *testbenches* category. The following list shows typical files you might find here.

- cdma\_2ms\_idata
- cdma\_2ms\_qdata
- dqpsk\_20ms\_idata
- dqpsk\_20ms\_qdata
- gsm\_5ms\_idata
- gsm\_5ms\_qdata

See “[Testbenches Elements](#)” on page 786 for more information.

## Bottom-Up Design Elements

The *bot\_upBB* library contains Verilog-A versions of the J-model which have two primary uses

- To facilitate estimation of Adjacent Channel Power Ratio (ACPR)
- To import transmitter impairments into the Alta™ SPW environment

The *bot\_upBB* library contains the following 5 verilog-A versions of the J-model

- 1st\_order\_j\_model
- 3rd\_order\_j\_model
- 5th\_order\_j\_model
- 7th\_order\_j\_model
- 9th\_order\_j\_model

These Verilog-A models read the same extracted J-model data files as the Alta SPW J-models.

For more information about using extracted J-model data files to estimate ACPR (adjacent channel power ratio) and about extracting J-models, see [Chapter 6, “Creating and Using Transmitter J-Models.”](#)

## Models for Top-Down RF System Design

If you are a system-level RF designer charged with designing RF systems from the specifications provided by DSP design teams, you can make use of the models described in this section. You can use the models described here for designing the top down analog RF subsystems that fit into larger DSP systems—from specifications provided by the DSP system designers. In particular, these models form a canonical set of top-down behavioral baseband models for exploring RF architectures in ADE. For methodology information and examples using these models, see [Chapter 8, “Methods for Top-Down RF System Design.”](#)

The baseband and passband models come from the following *rfLib* categories

- Category *top\_dwnBB* contains models of common RF function blocks.
- The default view of each model is the baseband view (called *veriloga*).
- Each model also has a differential passband view (called *veriloga\_PB*).
- Each model has a *symbol* view used in schematics.
- The only exceptions are the *BB\_loss* and *VGA\_BB* models which are meant only for baseband analysis and have no passband view.
- Category *top\_dwnPB* contains single-ended passband versions of the baseband models.
- Category *measurement* contains the instrumentation block and baseband signal generator models used to make RF measurements. These elements are not part of an RF architecture. They simply facilitate RF measurements and diagnostics.
- Category *testbenches* contains the test circuits used in this chapter to define the model specifications in the *rfLib*. Where possible, the models are specified in terms of standard RF measurements. The most precise way to describe a measurement is with a test circuit, set up instructions, and sample measurements. The circuits in the testbenches category serve this purpose

The *top\_dwnBB* models provide RF designers with a fast method to map RF system specifications into detailed RF designs. The baseband models facilitate fast evaluation of candidate RF architectures specified with DSP metrics. The passband views of the baseband models provide a behavioral system testbench for checking detailed designs of individual RF system components.

Baseband models are behavioral models and all behavioral models sacrifice some accuracy for increased simulation speed. Such sacrifices are usually acceptable in architectural studies because many implementation-dependent details do not affect high level decisions. The

modeling approach taken in top-down design is to simulate only those effects that drive the decisions at hand.

Baseband modeling in no way replaces passband modeling. Some effects missed by equivalent baseband models can affect high level decisions. However, the application of baseband models early followed by passband models later minimizes the number of slow simulations needed at low levels of design abstraction. Baseband models help you to quickly weed out designs that would surely fail tests simulated with passband models.

The success of a modeling approach to top-down design hinges on knowing two things

- How the models fit into the design flow
- Exactly what each modeling parameter means. This section defines the parameters that specify the models

## Baseband and Passband Models

The baseband and passband elements contained in the RF Library, *rfLib*, are organized into the following categories

**Table D-2 Categories of Baseband and Passband Elements in the rfLib**

Category in <i>rfLib</i>	Description of Content
<code>top_dwnBB</code>	<p>The Baseband Library.</p> <p>This category contains models of common architectural function blocks. The default view of the models in this category is the baseband view called <i>veriloga</i>, but all models in this category have a differential passband view called <i>veriloga_PB</i>.</p> <p>The only exceptions are the <i>BB_loss</i> and <i>VGA_BB</i> models which are meant only for baseband analysis and have no passband view.</p>
<code>top_dwnPB</code>	<p>The Passband Library.</p> <p>This category contains single-ended passband versions of the baseband models</p>
<code>measurement</code>	<p>Contains models used to make measurements. Includes the instrumentation blocks and the baseband signal generators. These elements are not part of an RF architecture. They simply facilitate measurements and diagnostics.</p>

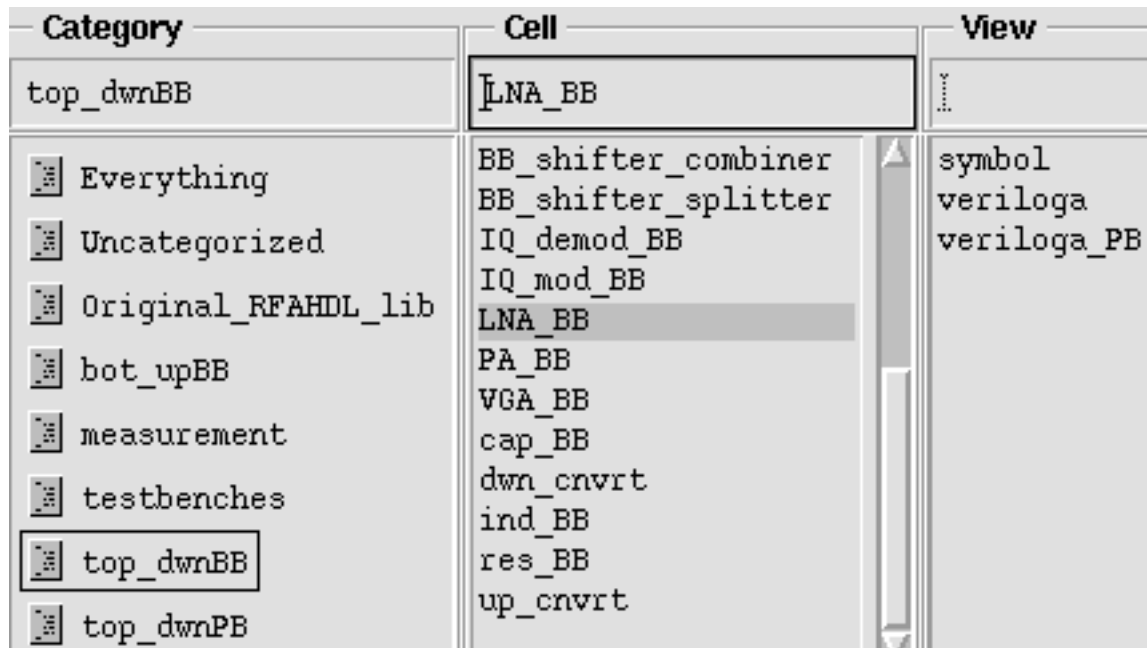
**Table D-2 Categories of Baseband and Passband Elements in the rfLib**

Category in <i>rfLib</i>	Description of Content
testbenches	Contains the test circuits used to define the model specifications in the <i>rfLib</i> . Where possible, the models are specified in terms of standard RF measurements. The most precise way to describe a measurement is with a test circuit, set up instructions, and sample measurements. The circuits in the testbenches category serve this purpose

Except for AM/PM conversion, all model specifications are defined by measurements taken from the passband models in the *top\_dwnPB* category.

Most baseband models in *top\_dwnBB* have both a baseband view (*veriloga*) and a differential passband view (*veriloga\_PB*). Figure D-2 shows the Library Manager with *LNA\_BB*, the low noise amplifier baseband cell selected.

**Library Manager Showing Views of *LNA\_BB* in *top\_dwnBB***



*LNA\_BB* has three views:

symbol      The schematic symbol for *LNA\_BB*.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

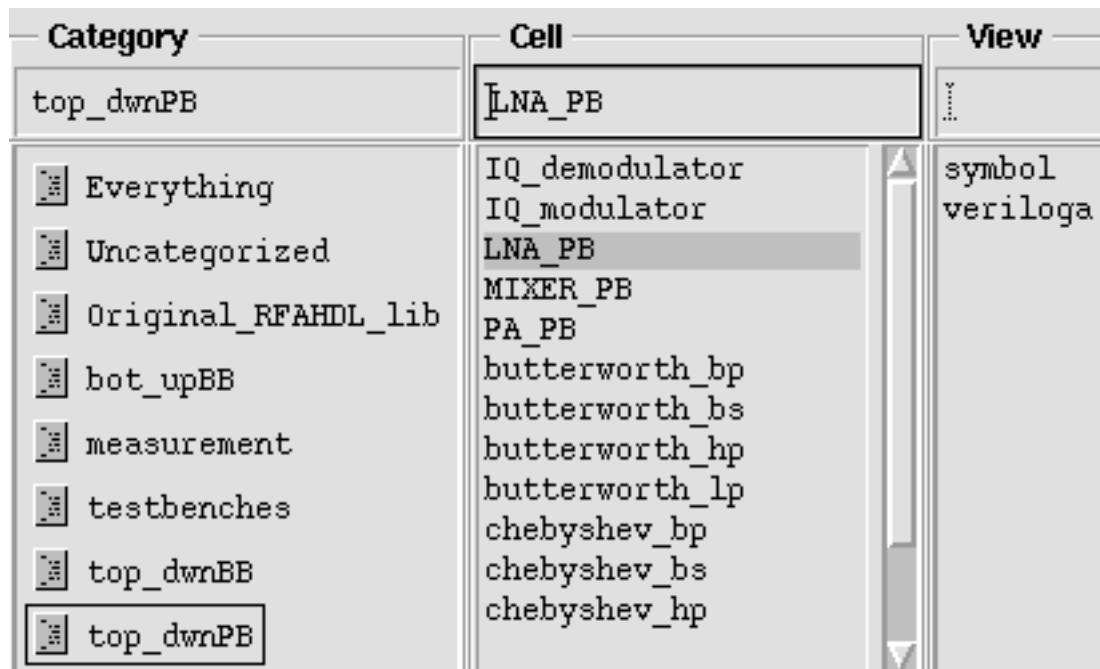
`veriloga`      The baseband LNA element with third-order effects.

`veriloga_PB`    The passband differential view of the same LNA element.

Those baseband models that do not have a differential passband view (*veriloga\_PB*) are meant only for baseband use.

The models in the *top\_dwnPB* category of *rfLib* are singled ended versions of the differential passband views. The single-ended passband models describe the measurements but the measurements also apply to the differential passband versions. Figure D-10 shows the Library Manager with *LNA\_PB*, the low noise amplifier baseband cell, selected.

**Figure D-10 Library Manager Showing Views of LNA\_PB in top\_dwnPB**



*LNA\_PB* has two views:

`symbol`      The schematic symbol for *LNA\_PB*.

`veriloga`      The passband LNA element with third-order effects.

## Assumptions About Behavioral Models

All behavioral models require assumptions and the top-down models at hand are no exception. The assumptions are summarized below:

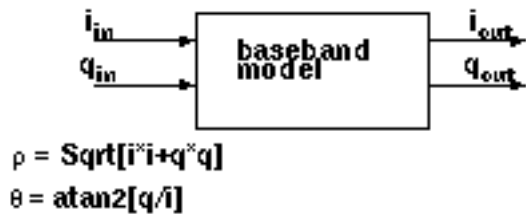
- Non-linear RF components are memoryless.
- Noise in the models is white, additive Gaussian noise which is split equally between the two output pins. In any model, *I* and *Q* noise sources are uncorrelated.
- Except for loading effects, signal flow is unilateral.
- Terminal impedances are purely resistive.
- The following is a single assumption stated three ways:
  - Only third-order non-linearities matter—even-order non-linearities do not matter.
  - Between any two baseband models, carrier harmonics are negligible.
  - Baseband models need not simulate IP2 or DC offsets.
  - The basic problem stated here in three ways is that *even* non-linearities do not produce output power at the carrier's fundamental frequency. Baseband models model only effects such as IP3 because third order distortion affects the fundamental. Second order distortion only affects the fundamental at the output of a subsequent, cascaded, non-linear device. You *can* build a baseband model of the two cascaded, non-linear devices in order to simulate how second order distortion in the first device affects the output of the second device at the fundamental. You *cannot* cascade individual baseband models of the two devices and observe any second-order effects.
- You can model all mixers, including those in modulators and demodulators, as a static non-linearity followed by an ideal multiplier.
- All local oscillators are sinusoidal.
- In addition, two compatibility assumptions apply.
- All baseband models, *except* the baseband signal generators and the instrumentation blocks, are compatible with Spectre RF.
- All models are written in Verilog-A.

## Inputs and Outputs for Baseband Models

Except where noted, all baseband models have inputs and outputs as shown in Figure [D-10](#).



## Inputs and Outputs on Baseband Models



**Note:** Be careful not to confuse baseband models with two-port S-parameter models.

When a device loads or drives adjacent devices with finite resistances, the terminals on the symbol are wider apart than when they load/drive adjacent models with ideal resistances (zero or infinite resistance).

## Some Common Model Parameters

Model parameters specify the baseband and passband models in the *top\_dwnBB* and *top\_dwnPB* categories in *rfLib*.

The AM/PM conversion parameters specify the baseband models only.

All other parameters specify both the passband and baseband models of a function block.

The more common parameters are defined in terms of passband test circuits described in this section. Except for the AM/PM conversion parameters, baseband parameters are not described with test circuits because, given the same parameters, the baseband models simulate the same peak voltages and currents as simulated by the passband models. Given the identical parameter values, the baseband and passband models simulate the same peak voltages and currents, but not the same power levels.

Parameters described here include

- Available Power Gain Parameter
- Input and Output Resistance Parameters
- Output 1dB Compression Point Parameter
- Noise Figure Parameter
- IP3 Parameter
- AM/PM Conversion Parameters

- ❑ AM/PM sharpness
- ❑ |radians| @ 1 dB cp
- ❑ |radians| @ big input
- ❑ {1, 0, 1} for {cw, none, ccw}

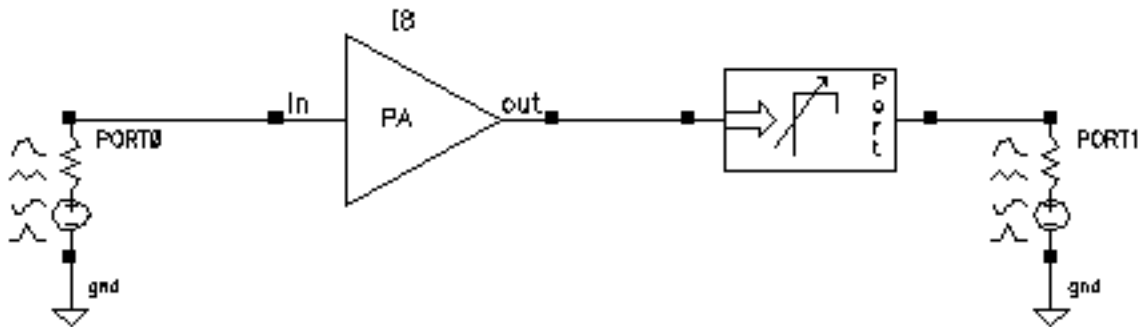
### Available Power Gain Parameter

When an amplifier's load is equal to its output resistance, available power gain equals the following

$$10 \times \log\left(\frac{\text{outputpower}}{\text{inputpower}}\right)$$

The test circuit in Figure D-10 is listed as *ava\_pwr\_gain* in the *testbenches* category in *rfLib*.

### The *ava\_pwr\_gain* Circuit



### Computing Constant Power Contours

The *ava\_pwr\_gain* test circuit is set up to compute constant power contours. As you would expect, maximum power transfer occurs when the load and output impedances are matched. The port adapter inserts reactive elements into the signal path to load the amplifier with the specified reflection coefficient.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

Figure D-11 on page 796 shows a Smith Chart that displays how the load power varies with the load reflection coefficient.

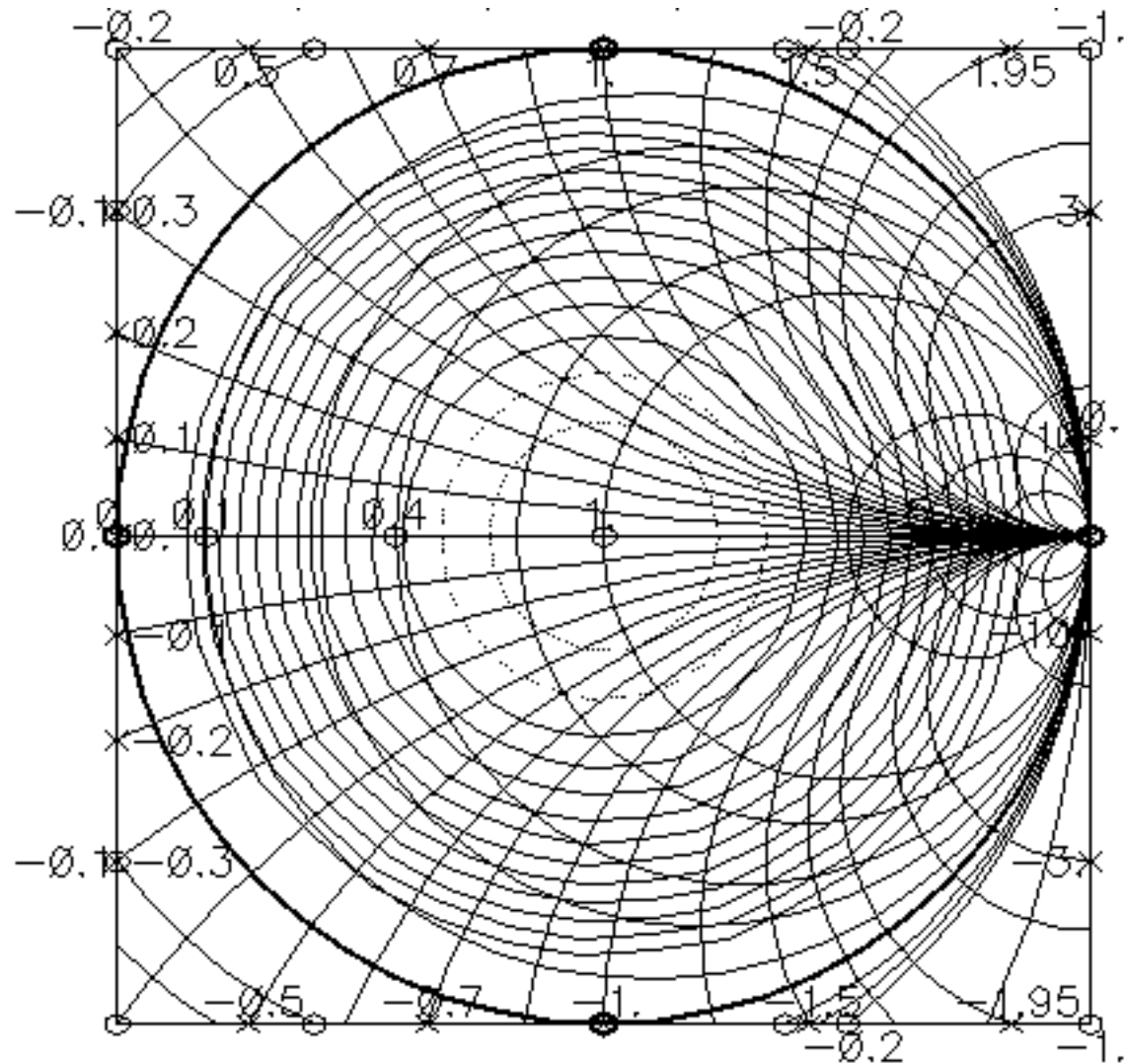
The load pull contours were computed by

- Sweeping the  $\rho\rho$  parameter in a PSS analysis ( $\rho\rho$  is the *phase* of the reflection coefficient)
- Sweeping the  $\rho\rho$  parameter with the Parametric Tool ( $\rho\rho$  is the *magnitude* of the reflection coefficient)

The *load reflection coefficient* is defined with reference to the amplifier output resistance, 300 Ohms in this case. The amplifier input resistance is 20 Ohms. The input source resistance is 50 Ohms. The amplifier 1 dB compression point is set high enough to make the amplifier linear. The available power gain parameter is 20 dB.

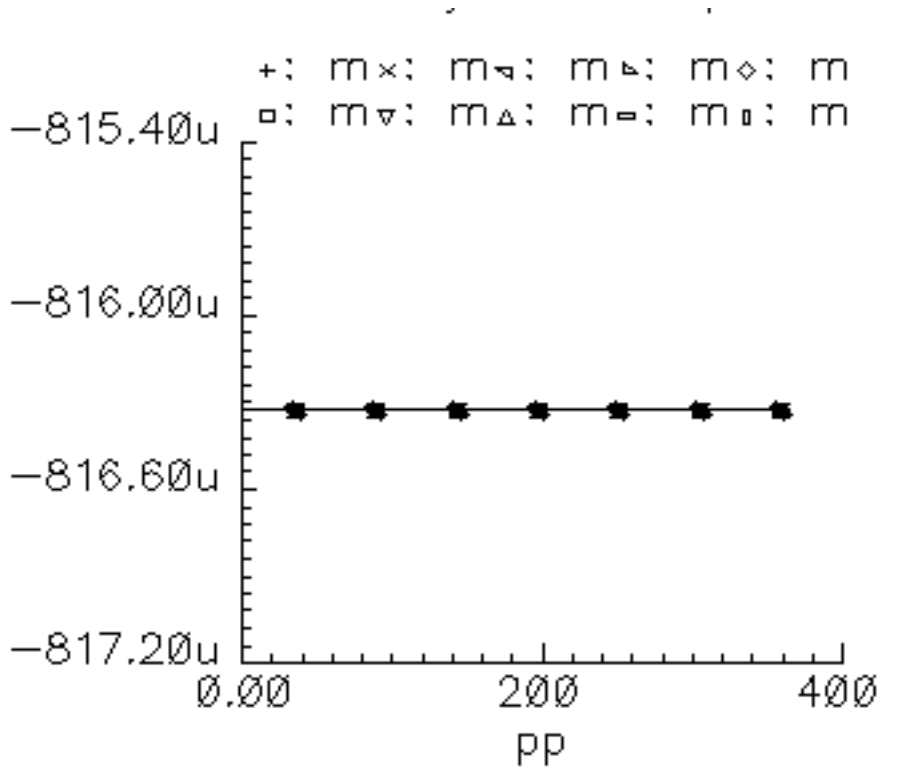
In order to generate the load pull contours you must save *both* the current flowing into the port adapter (*port*) as well as the current flowing into *Port0*.

Figure D-11 Smith Chart



When you place the cursor on the smallest contour on the Smith Chart, you can see that the amplifier delivers a maximum power of 81.63 mW to an optimum load of 300 Ohms (reflection coefficient = 0). When you plot the magnitude of the power coming from the input port against the sweep variable ( $\Gamma_P$ , phase of the reflection coefficient) you find that input power equals 816.3 uW, independent of load, as shown in Figure D-11. The ratio of maximum output to input power equals 100, or dB, as specified.

**Input power**



Note that the voltage gain in this test circuit does not equal 10 because the amplifier's input and output resistances are different. You can verify that the ratio of the output to input voltage is as follows

$$10 \sqrt{\frac{R_{out}}{R_{in}}}$$

where,  $R_{out}$  is the amplifier output resistance and  $R_{in}$  is the amplifier input resistance. This assumes the amplifier is not driven into non-linear operation.

## Input and Output Resistance Parameters

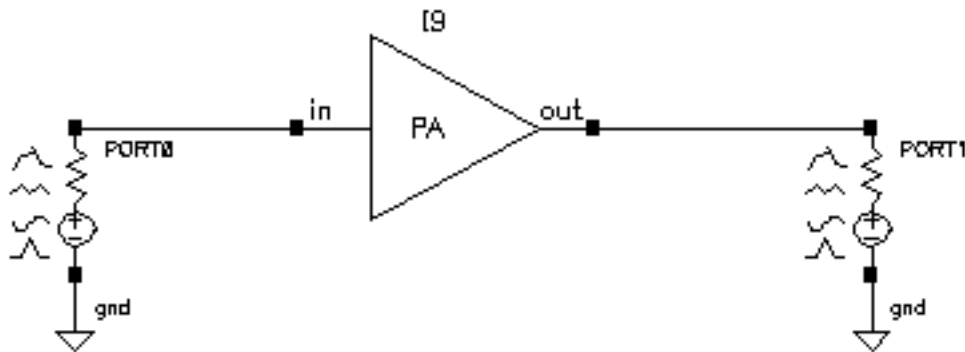
The input and output resistances specify the current drawn by the associated terminals as a linear function of terminal voltage. There is no test circuit for terminal resistances since the definition is so simple.

## Output 1dB Compression Point Parameter

The 1 dB compression point specifies a saturation non-linearity. It is the output power in dBm where the output power falls 1 dB below the power extrapolated linearly from the amplifier's linear region of operation.

The test circuit in Figure [D-11](#) is listed as *one\_db\_cp* in the *testbenches* category in *rfLib*.

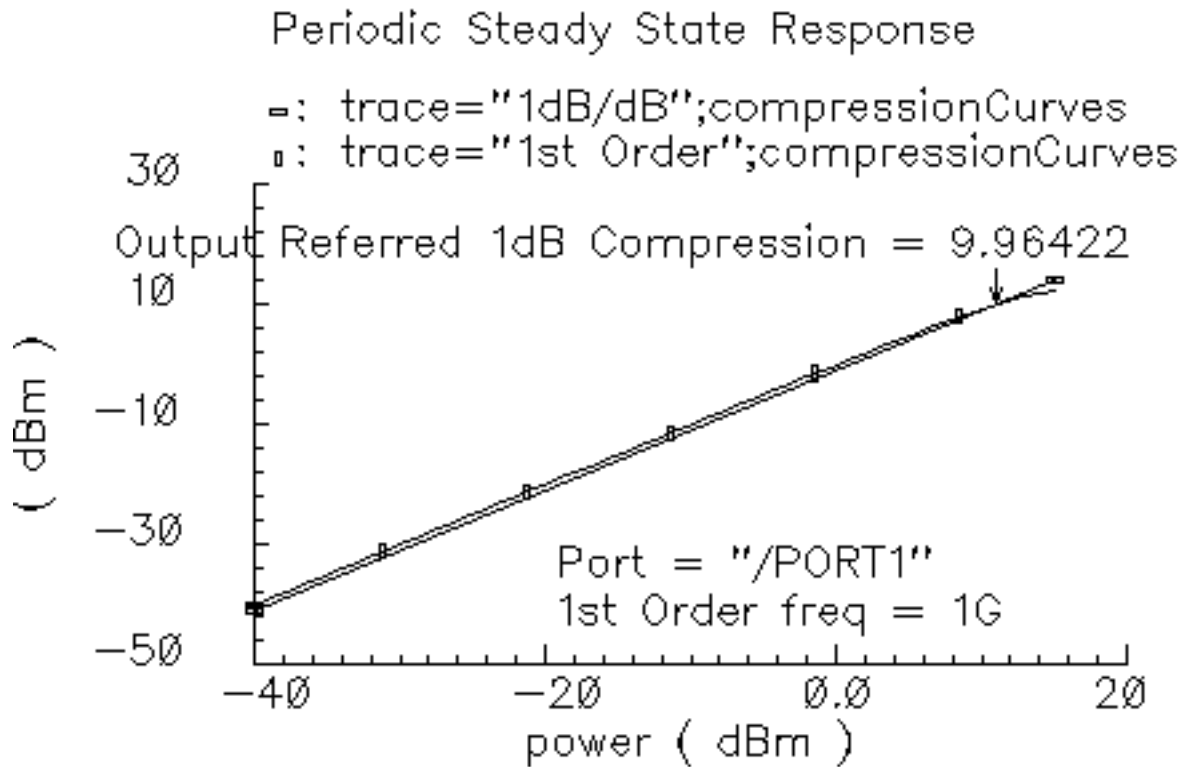
### The *one\_db\_cp* Circuit



In the *one\_db\_cp* test circuit, *power* is the dBm of power delivered by the leftmost port. The available power gain is 0 dB. The 1dB compression point is 10 dBm. The input and output resistances are 50 Ohms and so are the port resistances.

To measure the 1dB compression point, perform a swept PSS analysis. Sweep *power* from -40 dBm to 15 dBm in 50 linear steps. The output referred 1dB compression point is computed for the 1<sup>st</sup> harmonic with an *Extrapolation Point (dBm)* of -40. Click the rightmost port device to display the output as illustrated in Figure [D-11](#).

## Resulting 1dB Compression Point

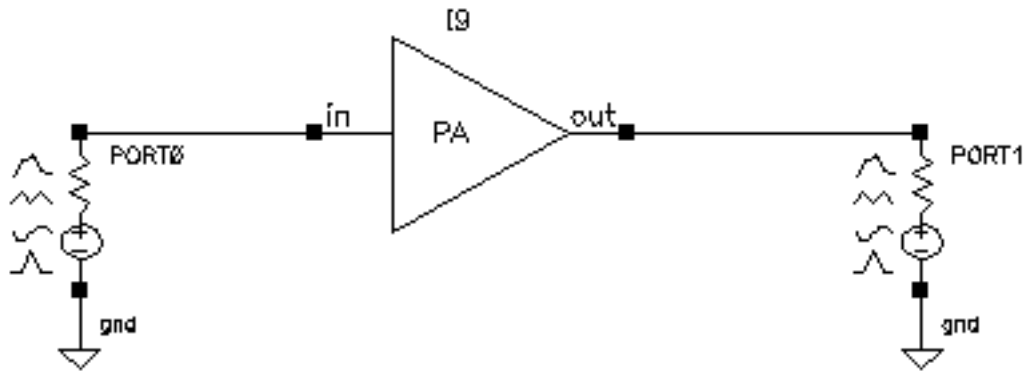


The specified output referred compression point is 10 dBm. The measured value is 9.964 dBm, which is fairly close to the specified value. The measured 1db compression point is as specified only when the driving source resistance matches the amplifier input resistance and the load port resistance matches the amplifier's output resistance. In all compression point and IPN calculations, input power is computed from the maximum power the input Port can deliver, not from an actual power measurement. If you mismatch either terminal you do not measure the specified compression point.

## Noise Figure Parameter

Noise figure is calculated as the input signal-to-noise ratio divided by the output signal-to-noise ratio. The test circuit for defining the noise figure parameter is shown in Figure [D-12](#). The circuit is listed as *noise\_figure* in the *testbenches* category of the *rfLib*. It is similar to the *one\_db\_cp* test circuit.

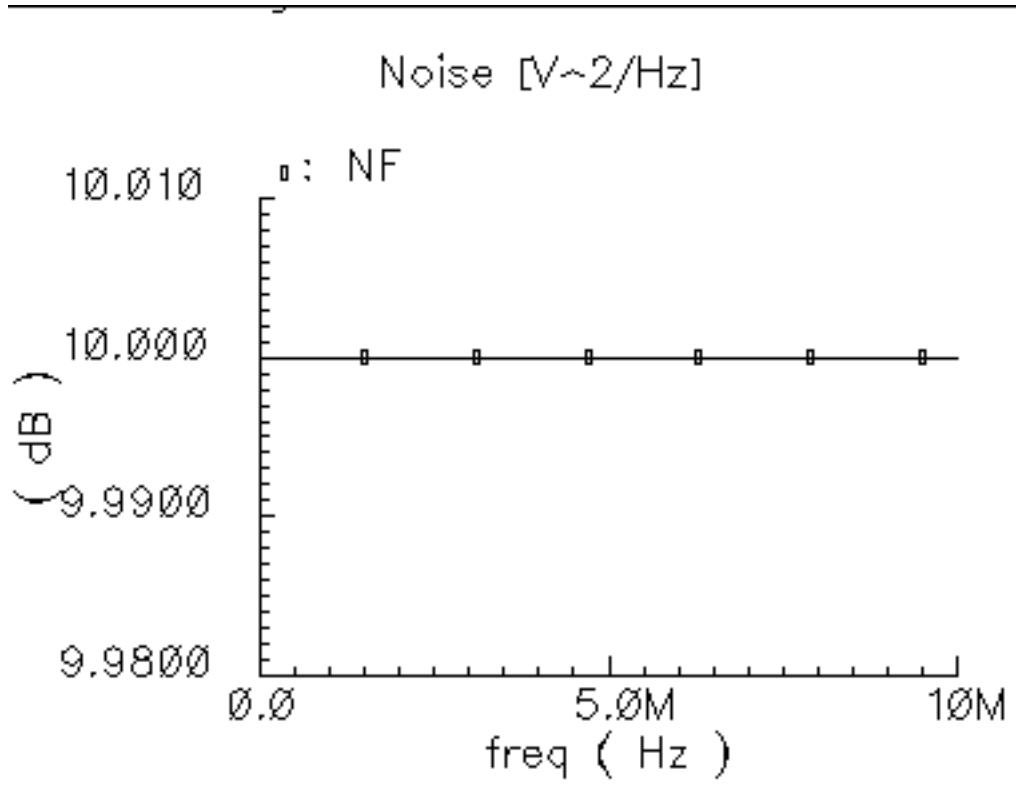
Figure D-12 The noise\_figure Circuit



The specified noise figure is 10 dB. A Spectre RF Noise analysis produces the noise figure shown in Figure D-12. To measure the specified noise figure, the driving port resistance must match the amplifier's input resistance. The port at the output does not have to match the amplifier's output resistance but the port impedance should be resistive. The input probe is the leftmost port, the output port is the rightmost port. Since the model is static, you can compute noise figure over any frequency interval.



## Noise Figure Results

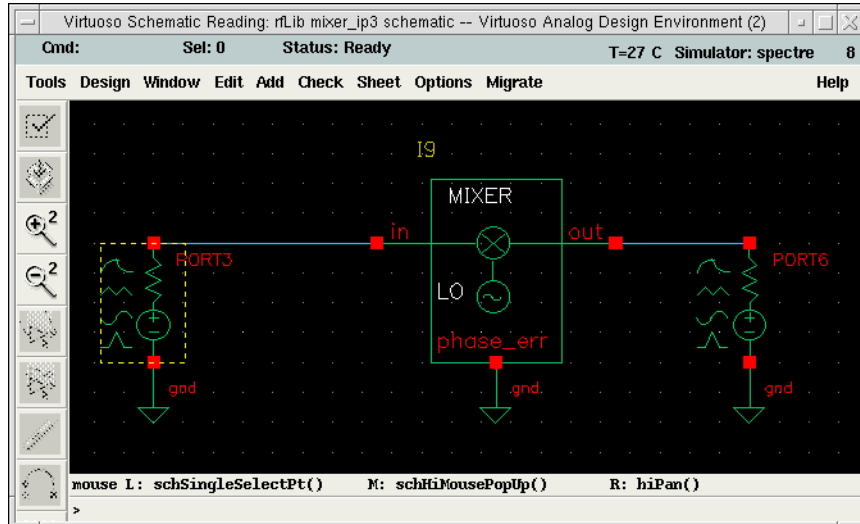


## IP3 Parameter

IP3 is measured with a two-tone test. One tone is the fundamental PSS frequency while the other is the frequency in a single point PAC analysis. IP3 is defined as the input power level in dBm where the extrapolated power in one of the third order intermodulation terms equals the extrapolated power in the fundamental term. As with the 1dB compression point measurement, input and output terminals must be matched to the source and load respectively.

The IP3 specification is demonstrated step by step on the mixer model because the mixer IP3 measurement can be confusing. Figure [D-13](#) shows the test circuit. The circuit is listed as *mixer\_ip3* in the *testbenches* category of the *rfLib*.

Figure D-13 The mixer\_ip3 Test Circuit



### Measuring IP3 for a Mixer

1. Open the schematic for the circuit and bring up ADE.
2. In the Virtuoso Analog Design Environment window, choose *Analyses — Choose*.  
The Choosing Analyses form appears.
3. Set up a PSS analysis.
  - a. Select *pss*.  
The title *Periodic Steady State Analysis* appears along with the fields required for specifying a PSS analysis.  
A 920 MHz tone already appears in the form.
  - b. Add a *Fundamental Tone* called *eee* (the name is arbitrary) with a *Value* of 1 GHz.
  - c. Select *Beat Frequency*.
  - d. Click *Auto Calculate*.  
The result is 40M Hz.
  - e. For the *Number of harmonics*, type 2.
  - f. Select *Sweep*.
  - g. For the *Variable Name*, use the `power` variable.

**h.** In the Sweep Range pane, select *Start\_Stop*.

**i.** In the *Start* field, type -60.

**j.** In the *Stop* field, type 0.

**k.** Select *Linear*.

**l.** Select *Number of Steps*.

**m.** In the *Number of Steps* field, type 10.

**4.** Set up a PAC analysis.

**a.** Select *pac*.

The title *Periodic AC Analysis* appears along with the fields required for specifying a PAC analysis.

**b.** Set *SweepType* to *absolute*.

**c.** Select *Single-Point*.

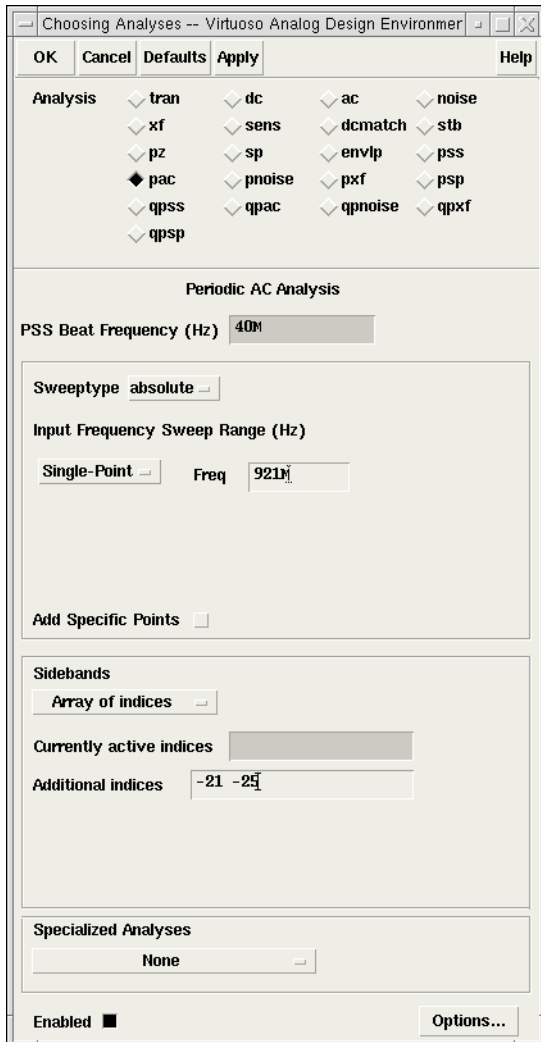
**d.** In the *Freq* field, type 921 M.

**e.** In the *sidebands* pane, select *Array of indices*.

**f.** In the *Additional indices* field, type -21 and -25.

After these steps, the Choosing Analyses form looks like this.

**Figure D-14 PAC Setup**



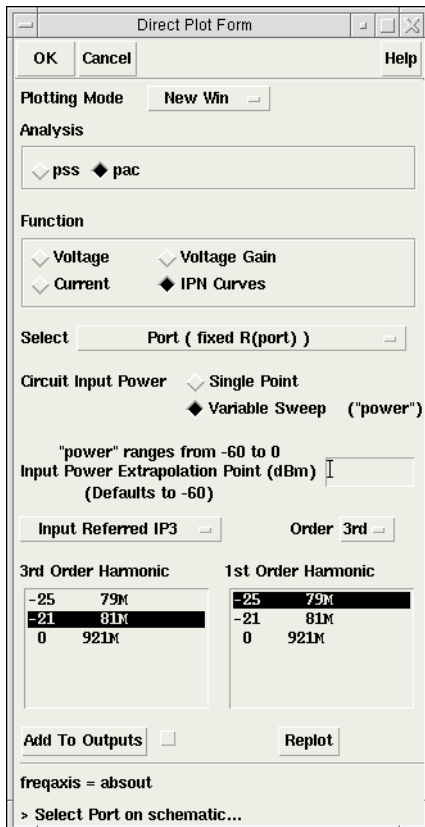
*Why select the -21 and -25 sidebands?* Recall from the assumptions, the non-linearity occurs before the frequency translation. The input tones to the non-linearity are the large 920 Mhz tone and the small signal 921 MHz tone. In an IP3 measurement only one tone must be large, the other can be small. PAC analysis performs small signal perturbations on the PSS solution. One perturbation term exiting the non-linearity appears at 921 MHz, right where it started. One of the third order intermodulation perturbation terms exiting the non-linearity appears at  $2 \cdot 920 - 921 = 919$  MHz. The ideal mixer, driven by a pure 1 GHz local oscillator, translates the 921 MHz tone to  $921 - 1000 = -79$  MHz while translating the 919 MHz tone to  $1000 - 919 = 81$  Mhz. A single point 921 MHz PAC analysis produces tones displaced from harmonics of the fundamental by 921 MHz. The PAC sidebands specify which harmonics to use. You save the 79 MHz tone by saving the -25<sup>th</sup> sideband because the fundamental frequency is 40 Mhz and

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

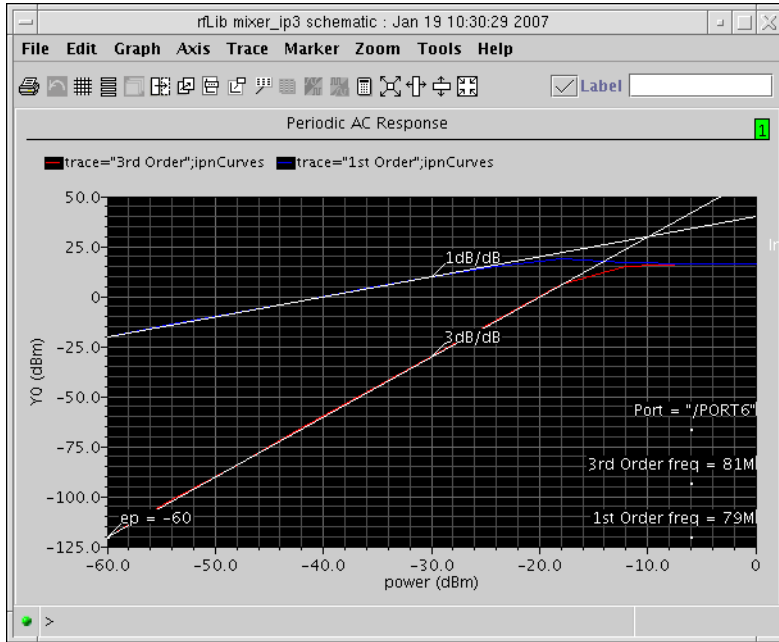
$921 - 40 \times 25 = -79$  MHz. You save the 81 MHz tone by saving the -21 sideband because  $921 - 40 \times 21 = 81$  MHz. Figure [Figure D-14](#) on page 804 shows the PAC setup.

5. Run the analysis.
6. Plot the PAC results. To do this, set up the Direct Plot form like this.



7. In the Composer window, click the output Port. The results appear as shown in [Figure D-15](#) on page 806.

**Figure D-15 IP3 Results**



The measured IP3 is, -10 dBm, as specified. The measured IP3 is as specified only if the input port resistance matches the input resistance of the device-under-test. Other input resistances produce a measured IP3 different than the one specified.

### ***Measuring IP3 for an LNA***

You can measure IP3 of an LNA by replacing the mixer with an LNA and ensuring the input terminal remains matched. In this example, remove the 1 GHz *Fundamental Tone* from the PSS analysis. The *Beat Frequency* should now be 920 MHz. In the PAC set up, change the additional indices from -21 and -25 to -1 and -2.

After the analysis completes, set up the PSS Results form as shown in Figure [D-16](#). As before, the input referred IP3 is 10 dBm, as specified. Figure [D-17](#) shows the LNA IP3 results.

Figure D-16 Direct Plot Form for the LNA

OK Cancel Help

Plot Mode  Append  Replace

Analysis

pss  pac

Function

Voltage  Current

IPN Curves

Select

Circuit Input Power  Single Point

Variable Sweep ("power")

"power" ranges from -100 to 0

Extrapolation Point (dBm)

(Defaults to -100)

3rd Order Sideband

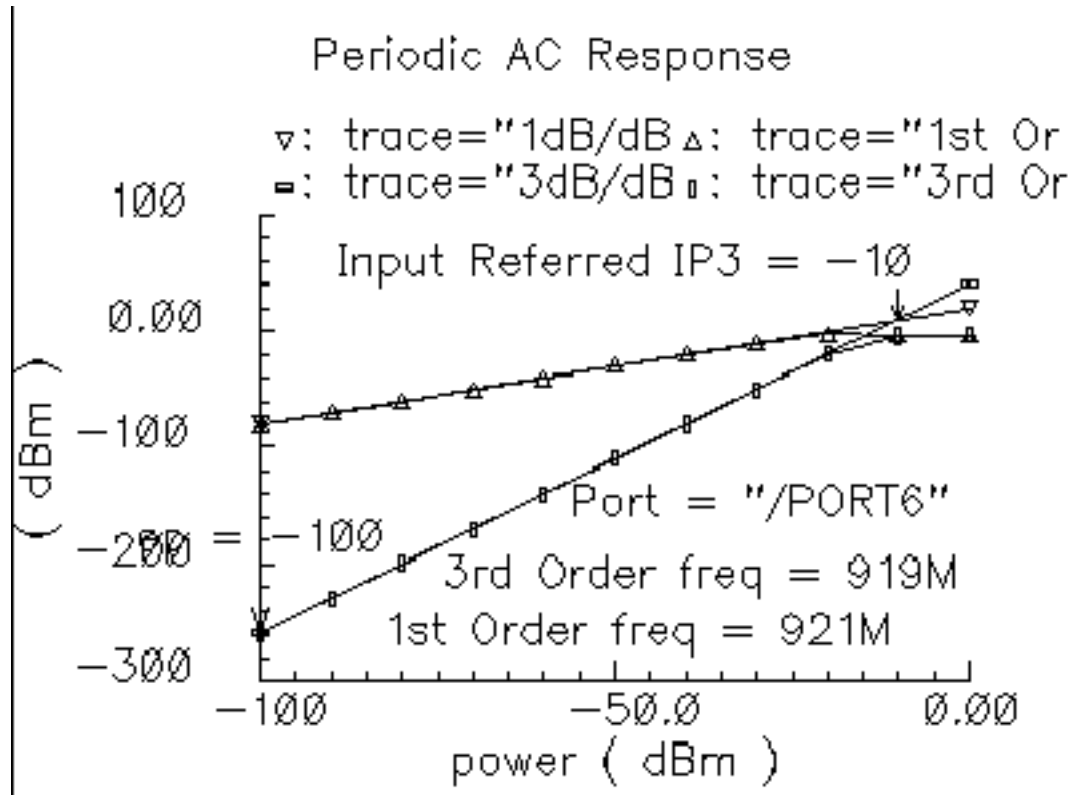
-2	919M
-1	1M
0	921M

1st Order Sideband

-2	919M
-1	1M
0	921M

> Select Port on schematic...

Figure D-17 Results for the LNA



### AM/PM Conversion Parameters

Only the baseband models include the four parameters for AM/PM conversion.

Table D-3 AM/PM Conversion Parameters for Baseband Models

AM/PM Parameter	Definition
AM/PM Sharpness	Defines how steep the output phase shift changes are with respect to input power.
radians @1dB cp	Defines the absolute value of the output phase shift at the 1dB compression point for power amplifiers. This is the phase shift at an arbitrary output power level for some models.
radians @big input	Defines the absolute value of the output phase shift as input power goes to infinity (if it could go to infinity)

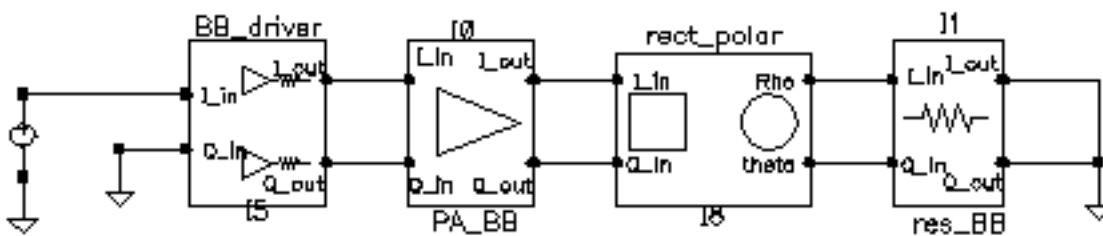


**Table D-3 AM/PM Conversion Parameters for Baseband Models**

AM/PM Parameter	Definition
{1, 0, -1} for {cw,none,ccw}	Defines the direction of the phase shift. 1 for clockwise, 0 for no phase shift, -1 for counter clockwise.

The test circuit in Figure D-3 is listed as *am\_pm\_test\_ckt* in the *testbenches* category in *rfLib*.

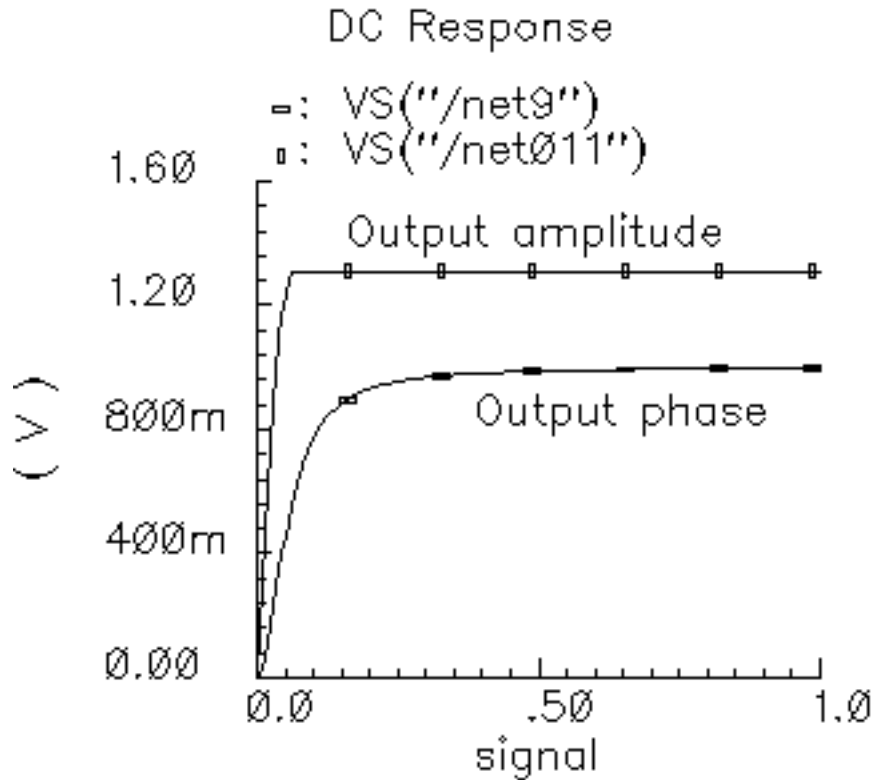
### The *am\_pm\_test\_ckt* Circuit



In the *am\_pm\_test\_ckt* test circuit

- The first block (*BB\_driver*) scales the control voltage generated by the leftmost element so that the output equals the specified dBm when the control voltage equals 1 volt. This is done so you can specify maximum dBm but still sweep linearly from zero signal.
- The second block (*PA\_BB*) is a power amplifier.
- The third block (*rect\_polar*) transforms the rectangular description of the baseband signal into polar coordinates so you can observe the phase shift and output signal level directly.
- Figure D-18 shows the output amplitude and phase as functions of the input signal level. Generate these with a swept DC analysis. Sweep the *signal* variable from 0 to 1 in 200 linear steps and display the *rect\_polar* outputs.

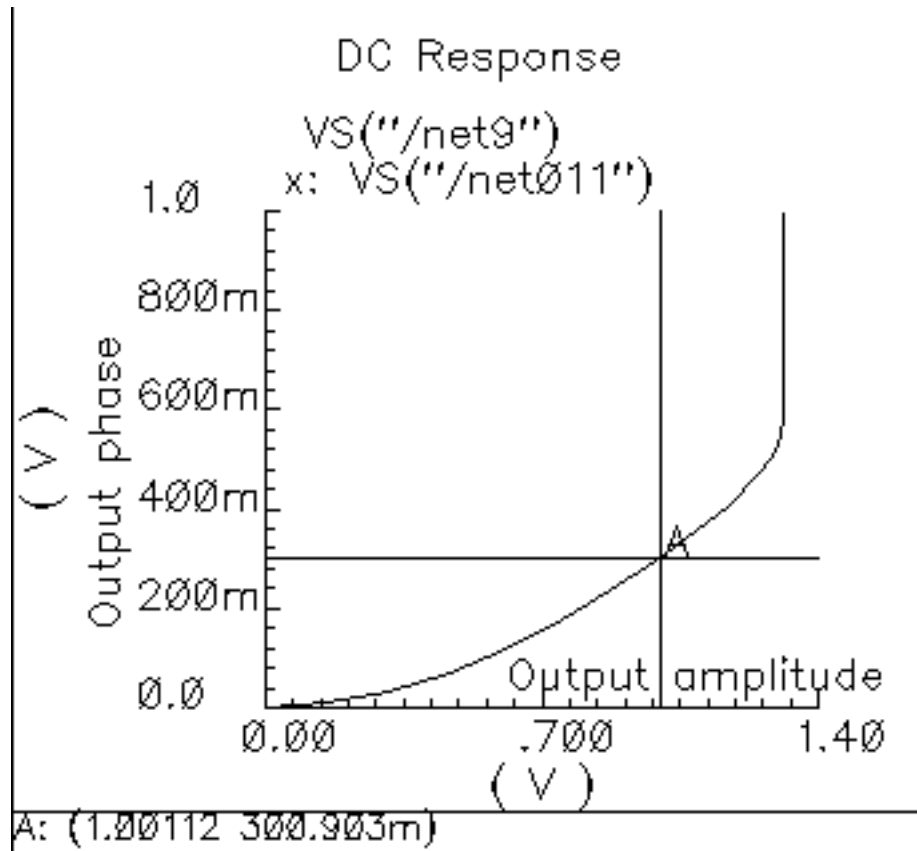
Figure D-18 Output Amplitude and Phase



By changing the x-axis to be the output amplitude trace, you can confirm that the phase shift at the output referred 1 dB compression point of 10dBm (or 1 volt peak across a 50 ohm load) equals 0.3 radians, as specified. Figure [D-19](#) shows the plot.

Note that the measured power across the load is as specified only when the load matches the amplifier output resistance. If you mismatch the load you do not measure the specified phase shift at the specified output power level.

Figure D-19 Output Phase Versus Output Amplitude



In the next three figures, output phase is plotted against input signal level. Each plot shows the effect of one of the AM/PM conversion parameters. You can generate the plots by applying the Parametric Tool to the existing analysis.

Figure [D-20](#) shows the effect of the  $|radians|@1\text{ db cp}$  parameter. Sweep  $rad\_cp$  from 10 m to 100 m in 5 linear steps.

Figure D-20 Output Modified by the |radians|@1 db cp Parameter

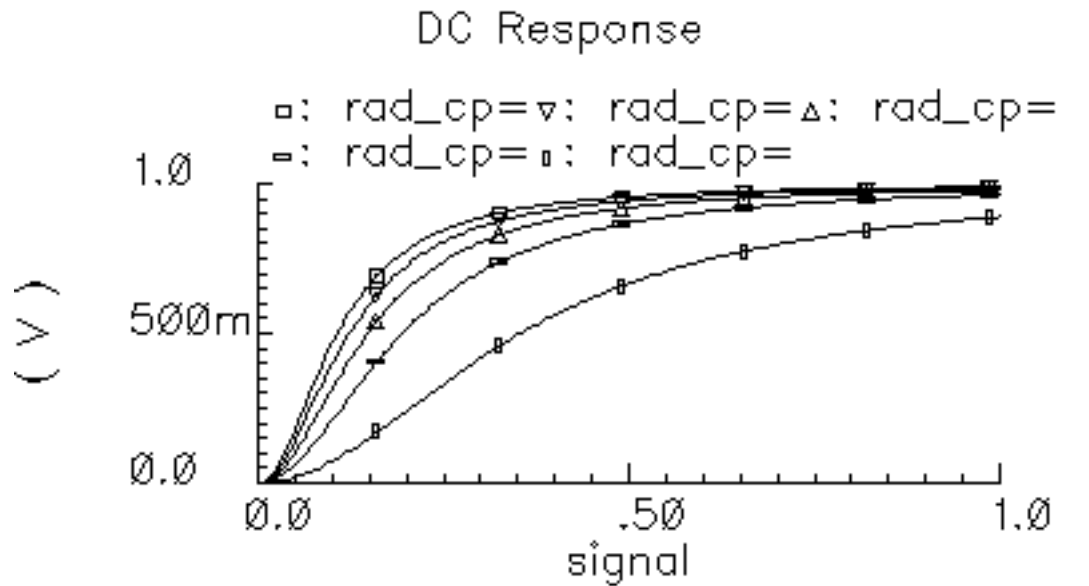


Figure D-21 shows the effect of the *am/pm sharpness* parameter. Sweep sharpness from 1 to 6 in 5 linear steps.

Figure D-21 Output Modified by the Sharpness Parameter

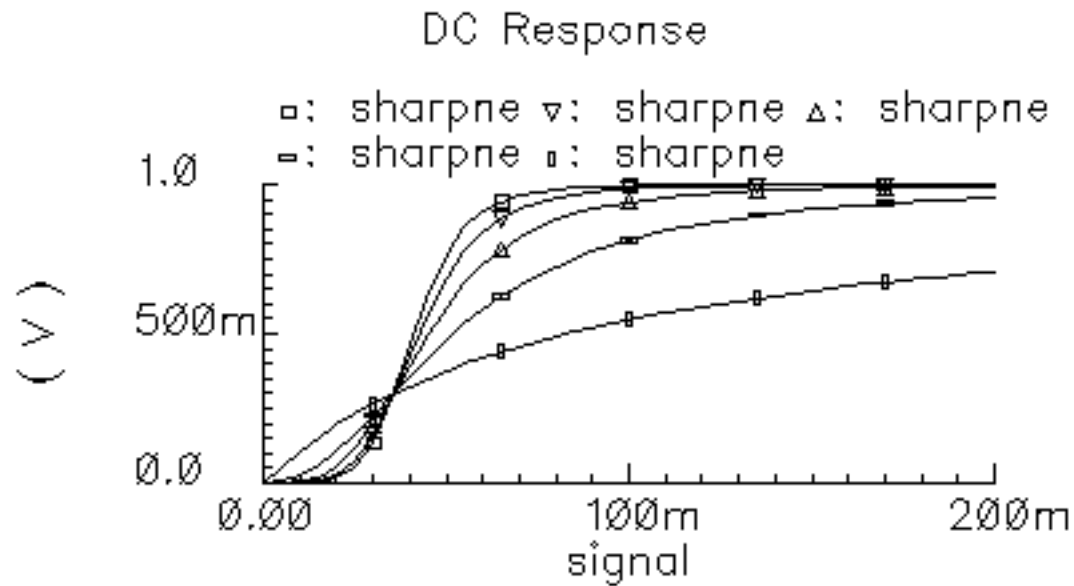
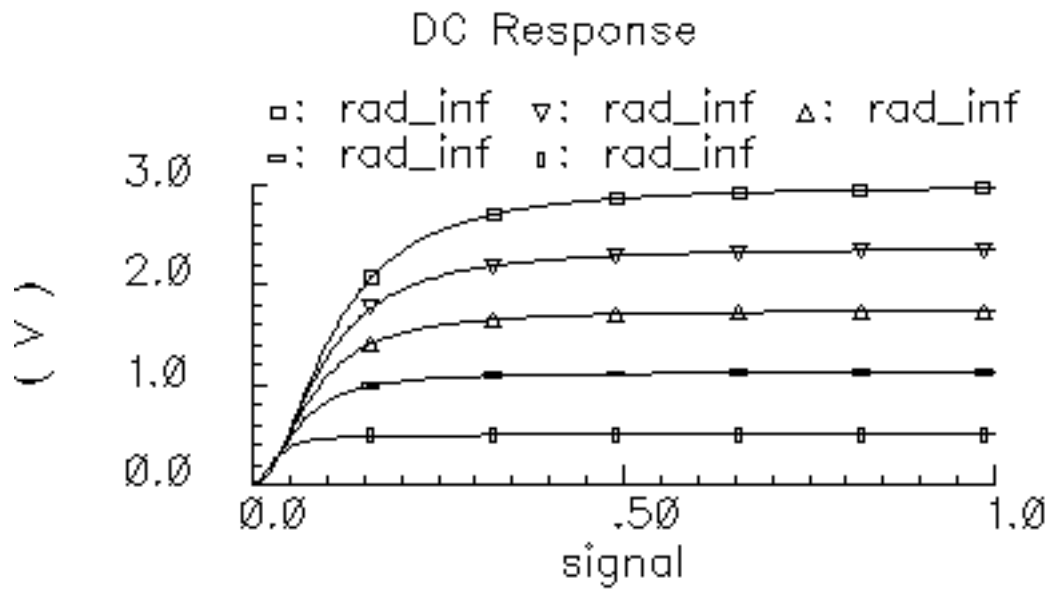


Figure D-22 shows the effect of the *rad\_inf* parameter. Sweep *rad\_inf* from 0.5 to 3 in 5 linear steps.

**Figure D-22 Output Modified by the *rad\_inf* Parameter**



## Library Description

The new models are listed in the table and described in the following sections.

<b>Top-Down Model</b>	<b>Model Name in <i>rfLib</i></b>
Power Amplifier	<i>PA_BB</i>
Low Noise Amplifier	<i>LNA_BB</i>
I/Q Modulator	<i>IQ_mod_BB</i>
I/Q Demodulator	<i>IQ_demod_BB</i>
Upconverting Mixer	<i>up_cnvrt</i>
Downconverting Mixer	<i>dwn_cnvrt</i>
Inductor	<i>ind_BB</i>
Capacitor	<i>cap_BB</i>
Resistor	<i>res_BB</i>
Linear Time-Invariant Filters	
Butterworth Bandpass Filter	<i>BB_butterworth_bp</i>
Butterworth Bandstop Filter	<i>BB_butterworth_bs</i>
Butterworth Highpass Filter	<i>BB_butterworth_hp</i>
Butterworth Lowpass Filter	<i>BB_butterworth_lp</i>
Chebyshev Bandpass Filter	<i>BB_chebyshev_bp</i>
Chebyshev Bandstop Filter	<i>BB_chebyshev_bs</i>
Chebyshev Highpass Filter	<i>BB_chebyshev_hp</i>
Chebyshev Lowpass Filter	<i>BB_chebyshev_lp</i>
Butterworth Bandpass Filter (laplace)	<i>BB_butterworth_bp_laplace</i>
Butterworth Bandstop Filter (laplace)	<i>BB_butterworth_bs_laplace</i>
Butterworth Highpass Filter (laplace)	<i>BB_butterworth_hp_laplace</i>
Butterworth Lowpass Filter (laplace)	<i>BB_butterworth_lp_laplace</i>
Chebyshev Bandpass Filter (laplace)	<i>BB_chebyshev_bp_laplace</i>
Chebyshev Bandstop Filter (laplace)	<i>BB_chebyshev_bs_laplace</i>

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## The RF Library

---

<b>Top-Down Model</b>	<b>Model Name in <i>rfLib</i></b>
Chebyshev Highpass Filter (laplace)	<i>BB_chebyshev_hp_laplace</i>
Chebyshev Lowpass Filter (laplace)	<i>BB_chebyshev_lp_laplace</i>
Baseband Shifter Combiner	<i>BB_shifter_combiner</i>
Baseband Shifter Splitter	<i>BB_shifter_splitter</i>
Rectangular-to-Polar Coordinate Transformation	<i>rect_polar</i>
Polar-to-Rectangular Coordinate Transformation	<i>polar_rect</i>
Ideal Transformer	<i>BB_xfmr</i>
Loss	<i>BB_loss</i>
Instrumentation Block	<i>comms_instr</i>
Offset Instrumentation Block	<i>offset_comms_instr</i>
Instrumentation Terminator	<i>instr_term</i>
Baseband Driver	<i>BB_driver</i>
Phase Generator	<i>phase_generator</i>

---

### Notes on Models Involving Frequency Translation

Passband models involving frequency translation have internal oscillators. When running a PSS or transient analyses, be sure to make the `maxstep` value small enough to prevent aliasing.

The Envelope and QPSS analyses do not require a non-default `maxstep` value because those analyses require external sources at the same frequencies that exist inside the Verilog-A modules. This requirement exists because there is currently no way to name sources inside Verilog-A modules.

The local oscillators were absorbed into the mixer/modulator/demodulator models to maintain a close relationship between the baseband and passband models. With the ability to use an external source, it would be possible to drive a mixer with non-sinusoidal oscillator signals. Such signals introduce harmonics that the baseband models cannot simulate. The first passband models encountered in the top-down flow, called level 1 passband models, are meant to introduce as few new effects as possible. The idea is to simplify diagnostics by introducing new effects sequentially. The next level of detail in the behavioral passband

## **Virtuoso Spectre Circuit Simulator RF Analysis User Guide**

### **The RF Library**

---

models, the level 2 models, are left to the user for now. Level 3 behavioral models are unnecessary because they do not offer a significant advantage over device-level models.



## Top Down Baseband and Passband Models: top\_dwnBB and top\_dwn PB

### Power Amplifier Model

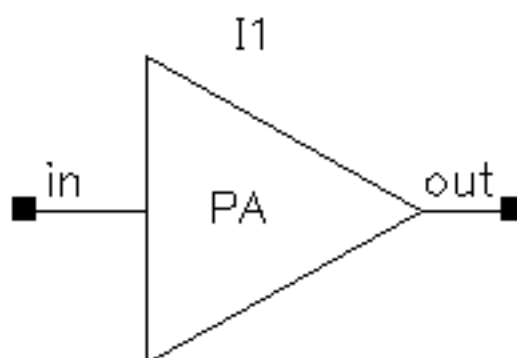
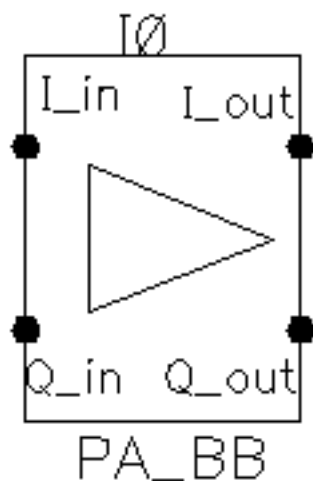
(baseband = PA\_BB, passband = PA\_BB)

Figure D-23 Baseband and Passband Power Amplifier Models

Single-ended  
baseband symbol  
(verilog)

Differential  
passband symbol  
(veriloga\_PB)

Single-ended  
passband symbol  
(veriloga)



The following parameters specify the power amplifier model.

Both Passband and Baseband models.

- available power gain
- input resistance

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

- output resistance
- output 1db cp in dBm
- noise figure

Baseband model only.

- am/pm sharpness
- |radians|@1db cp
- |radians|@ big input
- {1,0,-1} for {cw,none,ccw}

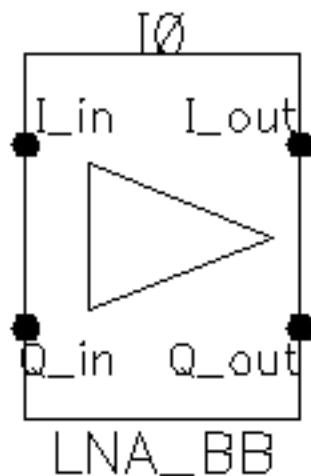
## Low Noise Amplifier Baseband Model

(baseband = LNA\_BB, passband = LNA\_PB)

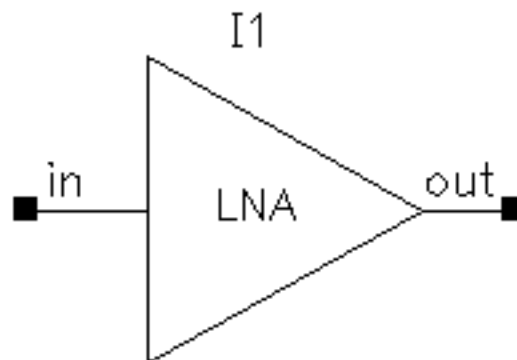
**Figure D-24 Baseband and Passband Power Amplifier Models**

Single-ended  
baseband symbol  
(verilog)

Differential  
passband symbol  
(veriloga\_PB)



Single-ended  
passband symbol  
(veriloga)



The following parameters specify the low noise amplifier model.

Both Passband and Baseband models.

- available power gain
- input resistance
- output resistance
- input referred IP3[dBm]

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

- noise figure

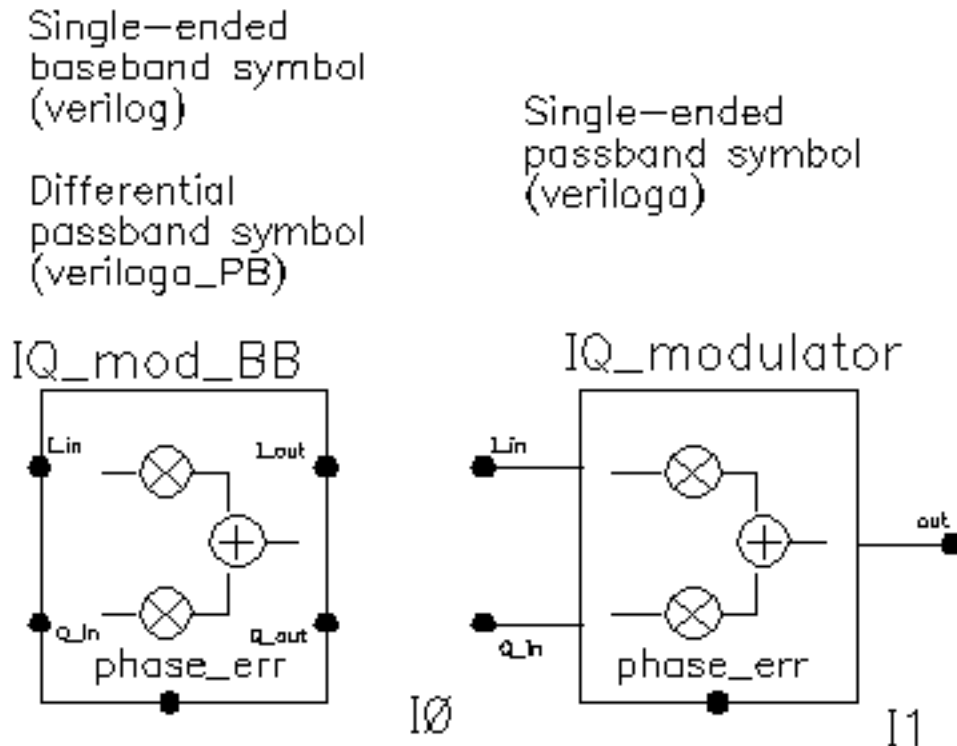
Baseband model only.

- am/pm sharpness
- cmp[dBm] = output power level where the next parameter is defined
- |radians|@cmp = output phase shift at output power of cmp.
- |radians|@ big input
- {1,0,-1} for {cw,none,ccw}

## **IQ Modulator Models**

(baseband = IQ\_mod\_BB, passband = IQ\_modulator)

**Figure D-25 Baseband and Passband IQ Modulator Models**



The `IQ_modulator` converts baseband signals to RF or IF. Figure D-26 summarizes exactly what the passband IQ modulator model does. The only difference between the baseband and passband models is carrier suppression. The non-linear functions,  $g_i$  and  $g_q$ , are specified by their available power gain and 1dB compression points just as in the power amplifier. The functions  $\gamma_i$  and  $\gamma_q$  characterize AM/PM effects in each mixer and are specified by the same parameters that specify power amplifier AM/PM conversion. Since noise is always added at the input, and the input is at baseband in this case, the noise sources are not doubled as they are in the power amplifier or LNA models. Noise figure is defined with reference to one input. Noise is injected at both inputs but the noise injected at just one input alone produces the specified noise figure. Thus, the noise figure parameter should be interpreted as noise figure per input. This model also includes a parameter called `quadrature_error` which specifies how far away the two local oscillator signals are from being exactly in quadrature.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

`Phase error` is the voltage on the phase error pin. The phase error pin has a fixed noiseless resistive input impedance of 50 ohms. The phase error pin can be used to introduce a dynamic phase error or phase noise. Phase noise can be fed into the phase error pin from a phase-domain PLL model or from a Port. Noise in Port models can be specified either by the internal resistance or by a data file that tabulates a power spectral density. The phase error pin can also be driven by a ramp or circular integrator output to model a frequency error between the incoming carrier and local oscillator.

The following parameters specify the IQ modulator. The available power gain and one dB compression point are explained first. The effects of the `phase_error` pin and the quadrature error parameter are discussed at the end of this section.

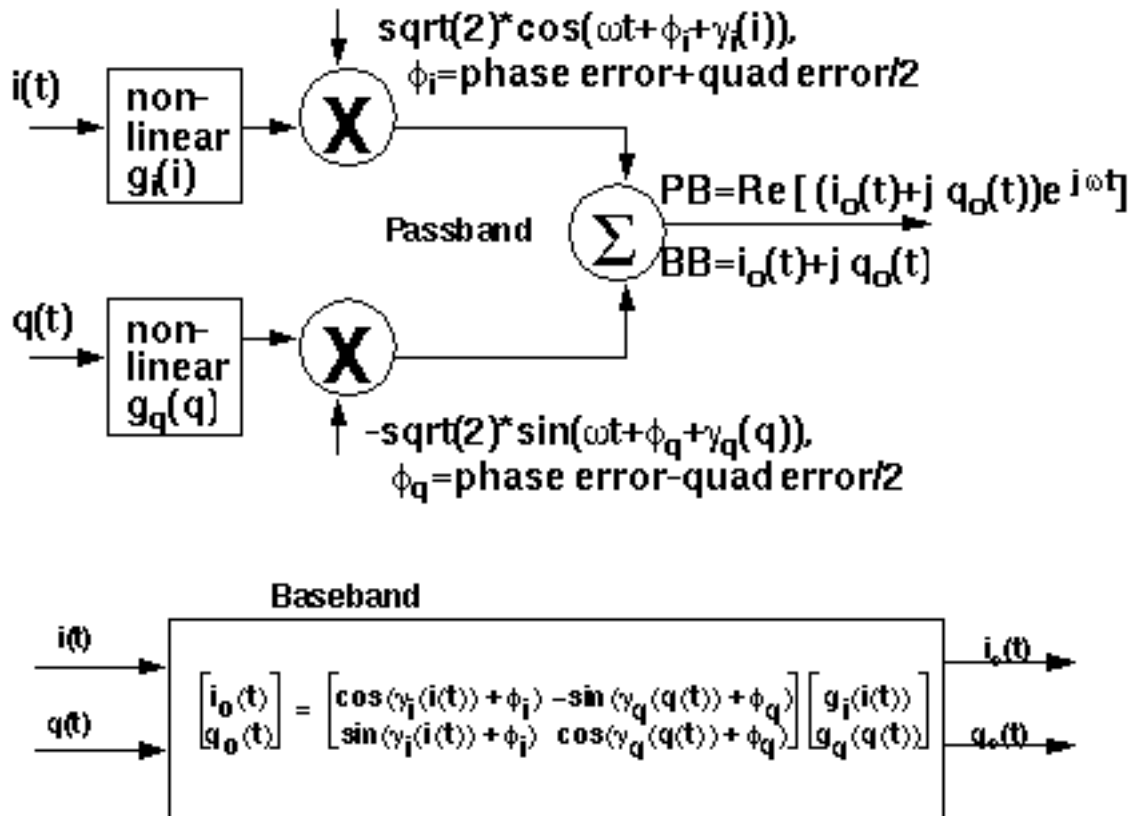
Both Passband and Baseband models.

- available I-mixer gain[dB]
- available Q-mixer gain[dB]
- input resistance
- output resistance
- I-output 1dB CP[dBm]
- Q-output 1dB CP[dBm]
- noise figure [dB]
- quadrature error

Baseband models only.

- I-sharpness factor
- Q-sharpness factor
- Q-radians@Q\_cmp
- I-radians@I\_cmp
- I-radians@big I-input
- Q-radians@big-input
- {1,0,-1} for {cw,none,ccw}
- {1,0,-1} for {cw,none,ccw}

Figure D-26 IQ Modulator Calculations



### Available Power Gain and 1dB Compression Point

Available power gain of the IQ-modulator is best explained with an example. Recall the circuit called `mod_1dbcp` listed in the `testbenches` category of the `rfLib`. The schematic contains two disjoint circuits. One shows how not to measure gain and compression point, the other shows the proper measurement.

Set up a PSS analysis. Both test circuits run in the same simulation. The beat frequency is 100 MHz. Save the first and 11<sup>th</sup> harmonics. In the options, set `maxstep` to 50 ps. Sweep the variable `power` linearly in 50 steps from -40 to 15.

1. After the analysis completes, plot the output referred 1dB compression point of the top circuit using -40 dBm as the Extrapolation point. First select the 11<sup>th</sup> harmonic (1.1 GHz) and click the output port in the top test circuit, the bad test circuit. Note that the linear gain is 3 dB lower than specified, as is the output referred 1dB compression point. The gain was specified as zero dB and the 1dB compression point was 10 dBm. The error arises from the fact that the input signal power splits between upper (1.1 GHz) and lower

(900 MHz) sidebands but the ADE measurement only looks at one output sideband. The bottom test circuit resolves the ambiguity by defining the gain of the IQ-modulator as the gain from the baseband input to an ideally-demodulated baseband output. The bottom test circuit follows the IQ-modulator with an ideal IQ-demodulator. The gain of the demodulator is zero dB and the 1dB compression point is high enough to render the demodulator distortionless.

2. Repeat the steps for plotting the 1dB compression point but this time chose the first harmonic and select the output port that loads the bottom circuit. Select the first (100 MHz) harmonic and plot the 1dB compression point again. Now you should see a 1dB compression point plot that reflects the specified parameters of the IQ-modulator. The gain is now also correct, which can be computed from the ratio of the output to input power well below the compression point. Figure [D-28](#) shows such a plot.

Figure D-27 1db Compression Point Test Circuit

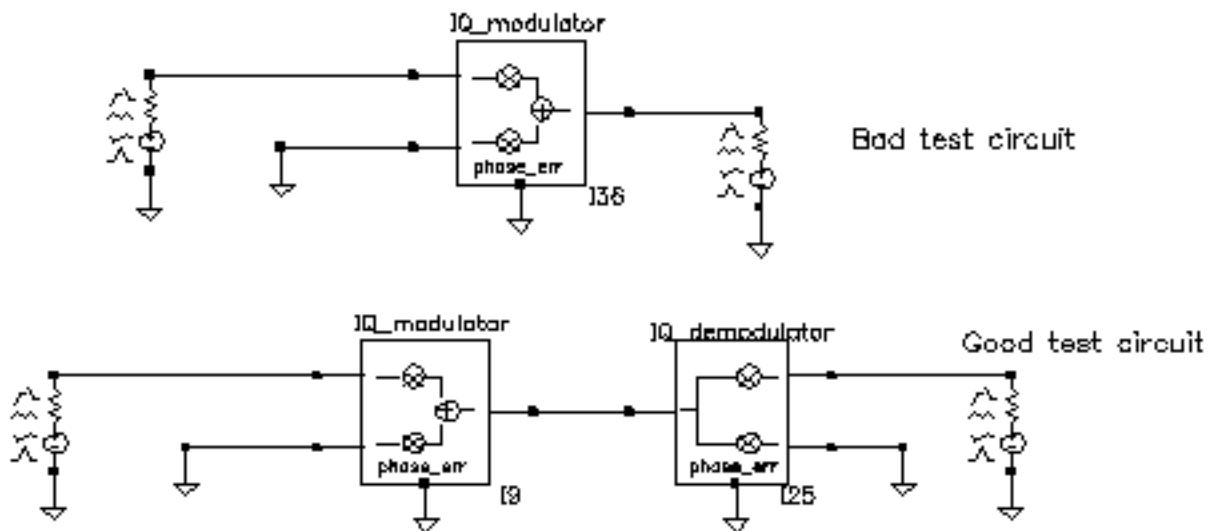
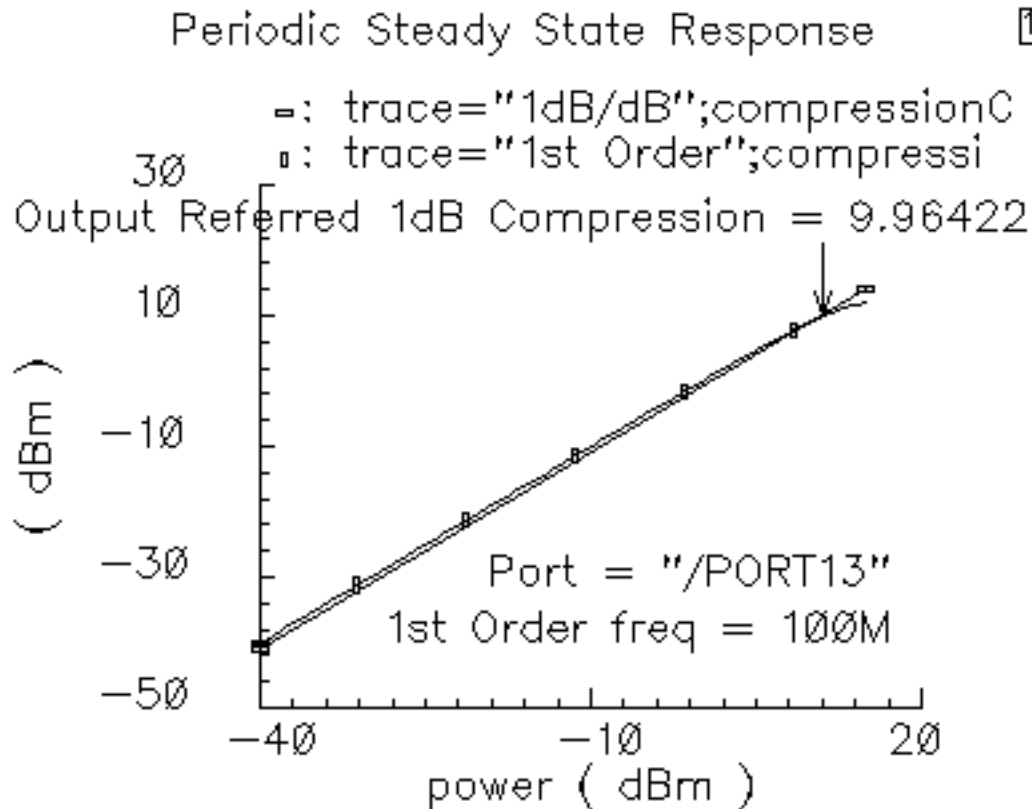




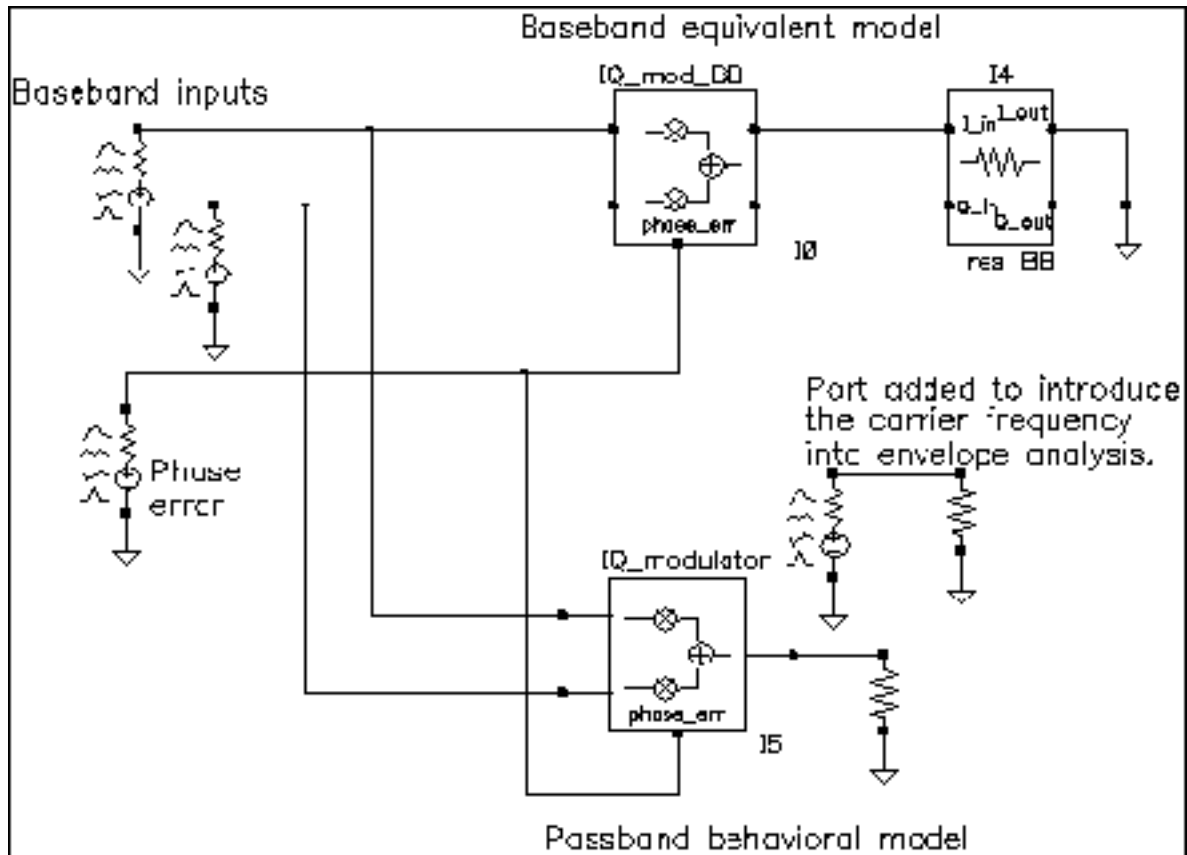
Figure D-28 1db Compression Point Plot



### Quadrature Error and Phase Error

Quadrature error describes how far away from 90 degrees the two local oscillators are from each other. Ideally, they are exactly 90 degrees, or  $\pi/2$  radians, apart in phase. In practice, parasitics and asymmetric delays can drive the phase shift away from  $\pi/2$ . Figure D-28 show a baseband test circuit and its passband equivalent. The schematic is listed in the rfLib *testbenches* category under the name *quad\_and\_phase\_error\_demo*. Both circuits are driven from a common set of baseband sources. The test circuit serves two purposes, it shows the correspondence between baseband and passband models and it demonstrates how quadrature error and phase error affect the baseband trajectory. The baseband input signal is a complex tone, which makes a circular input baseband trajectory. If there were no quadrature error, the baseband representation of the modulator output would also be a circle. With quadrature error, the output trajectory is an ellipse. If the `phase_err` pins are driven by a ramp, the ellipse precesses. The ramp represents a small but fixed difference between carrier and local oscillator frequencies.

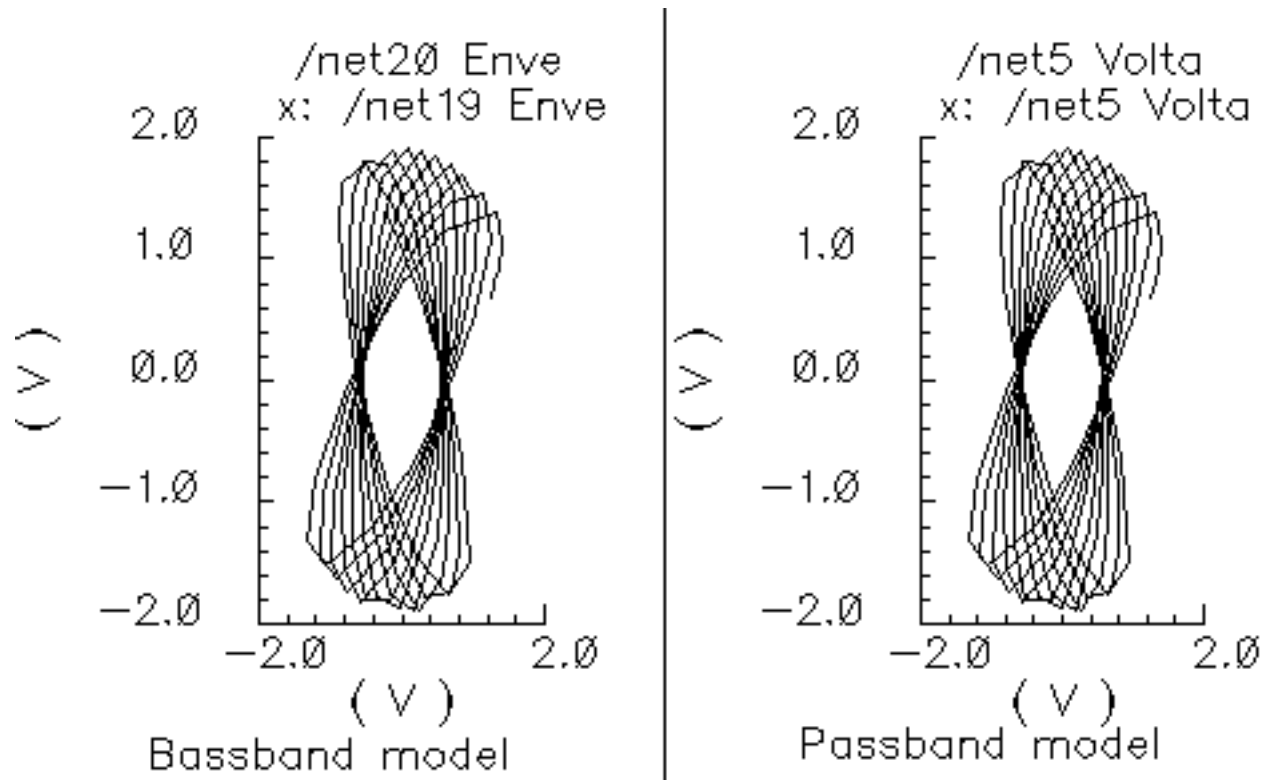
quad\_and\_phase\_error\_demo Circuit



To see these effects

1. Set up an ENVLP analysis with `carrier` as the Clock Name. Simulate 10 us of action and save the first harmonic.
2. When the analysis completes, open the Envelope Following Direct Plot form and set the sweep to `time`. Plot the two outputs of the `IQ_mod_BB` block.
3. Change the x-axis to be the I-output. You should see the left trajectory in Figure [D-28](#).
4. Add a subwindow for the passband equivalent result.
5. In the Direct Plot form, change the sweep to `harmonic time` and plot the real and imaginary parts of the first harmonic of the `IQ_modulator` output voltage.
6. Change the x-axis to be the real part of the first harmonic. You should now have two pictures that match those in Figure [D-28](#).

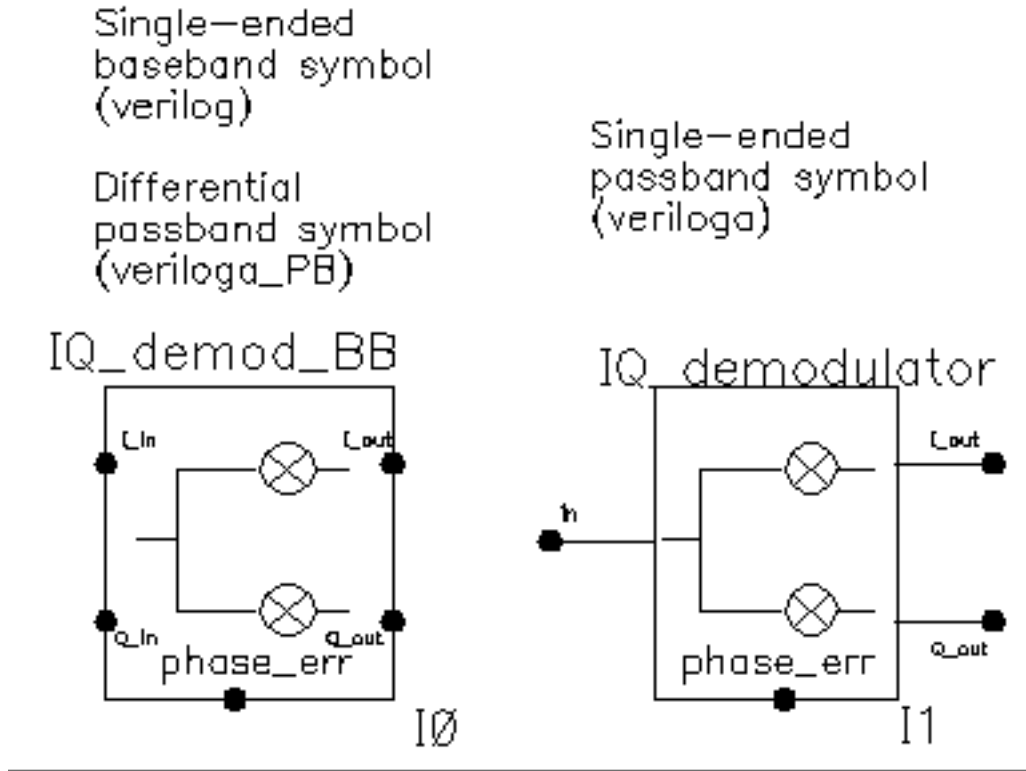
### Output Trajectories



## **IQ Demodulator**

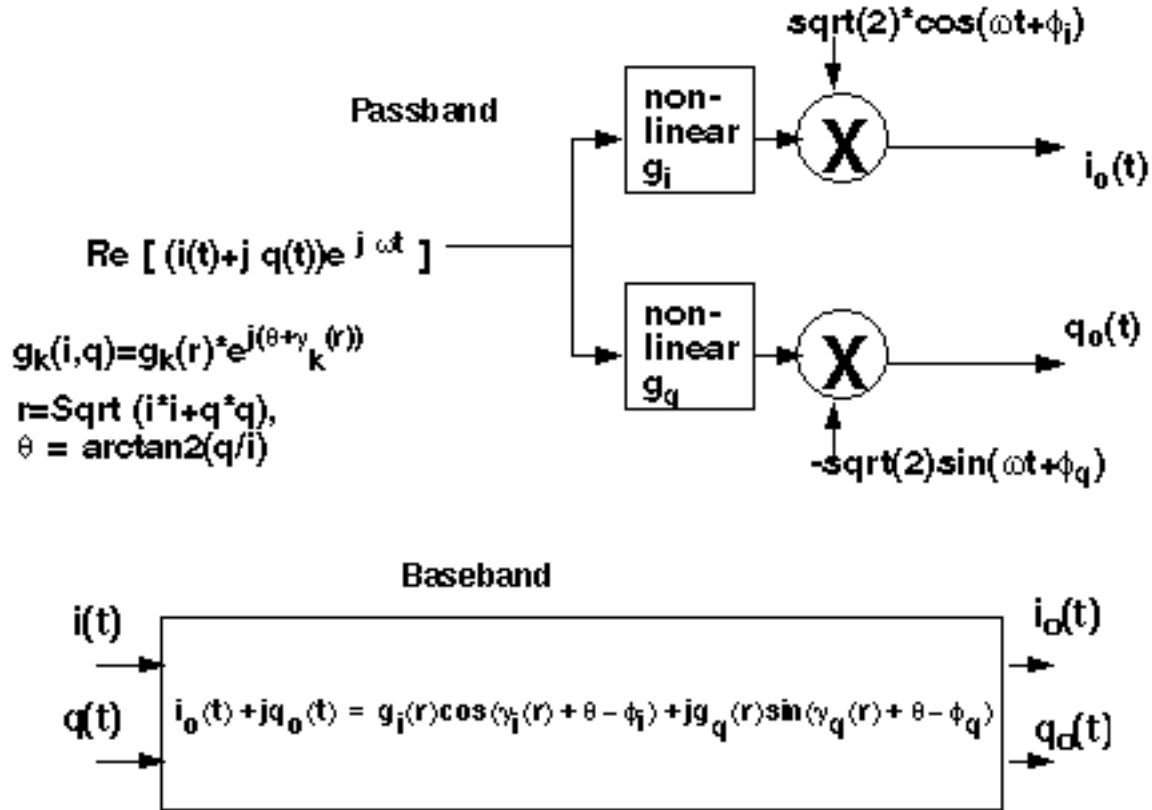
(baseband = IQ\_demod\_BB, passband = IQ\_demodulator)

**Figure D-29 Baseband and Passband IQ Demodulator Models**



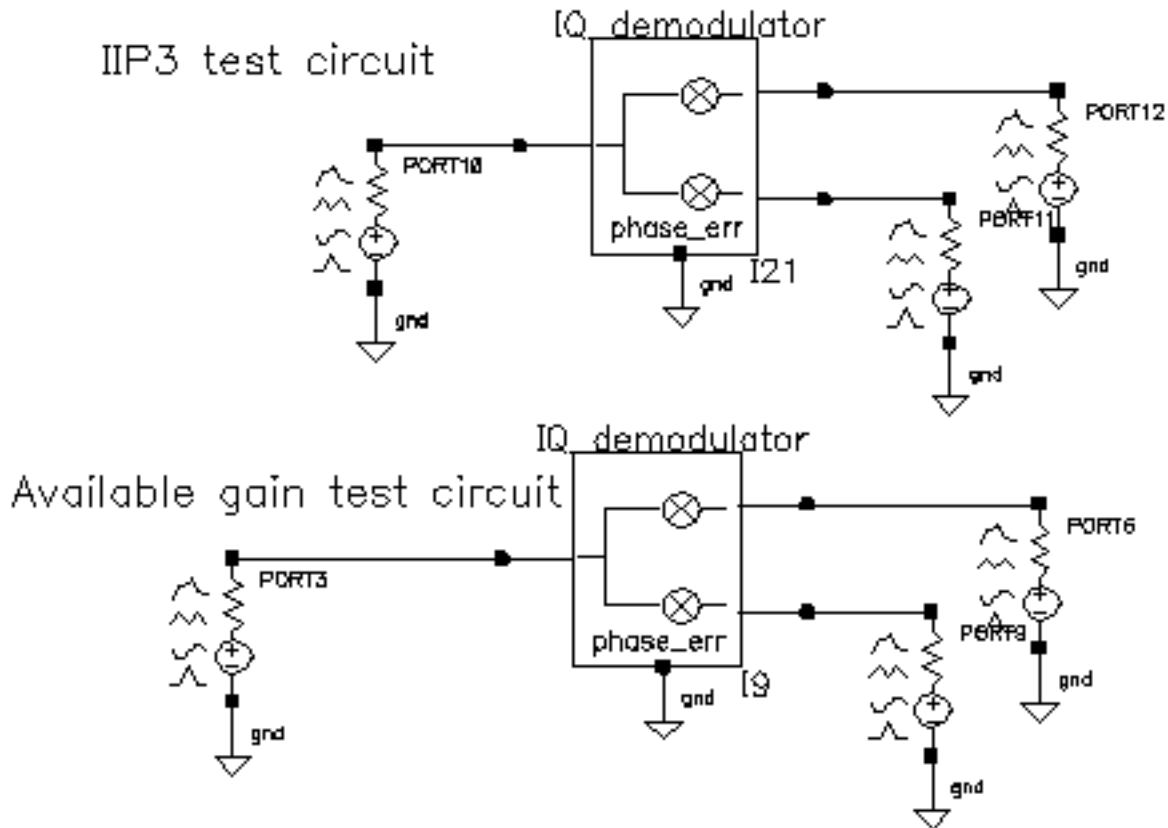
The `IQ_demodulator` converts RF (or IF) to baseband. Figure [D-30](#) shows exactly what the passband demodulator model does. The parameters are like those in the modulator blocks except saturation is specified by input referred IP3 instead of by 1 dB compression point. IP3 was chosen over the 1 dB compression point for specifying saturation because the demodulator usually lies in the receive path and receiver blocks are usually specified with IP3.

Figure D-30 IQ Demodulator Calculations



The circuit called `demod_ip3` in the `testbenches` category of the `rfLib` shows how the gain and IP3 parameters are defined. Figure [D-31](#) shows the schematic. Both the input and the output resistances are matched.

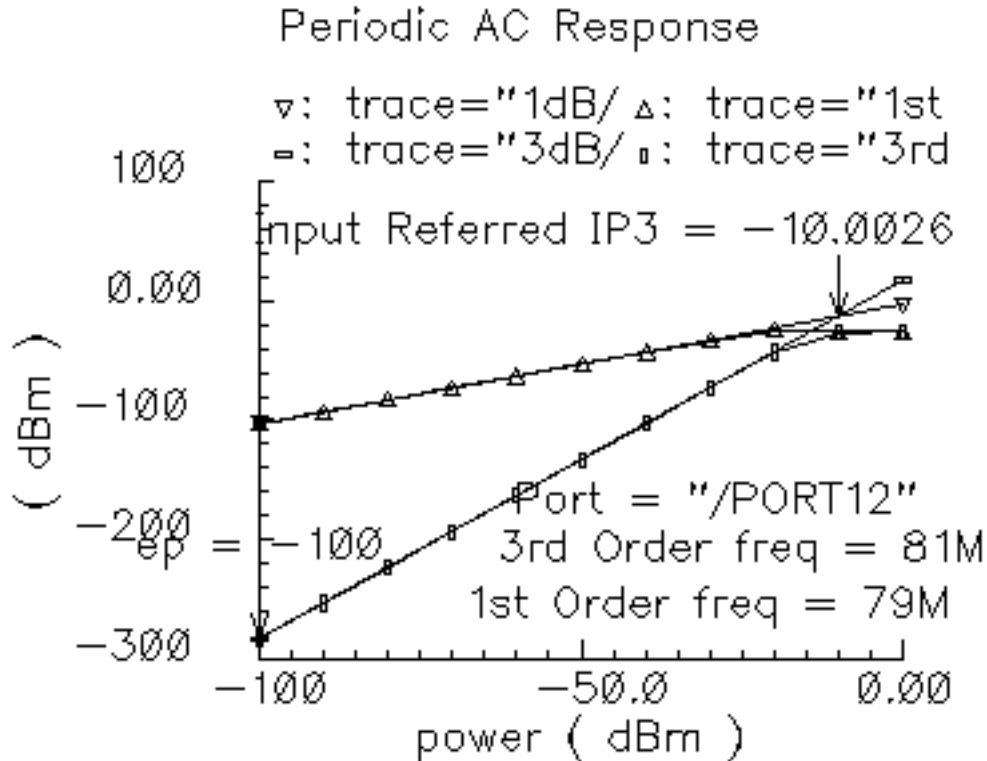
Figure D-31 The demod\_IP3 Schematic



1. Recall the demod\_IP3 circuit and set up a swept PSS analysis. Let the *Beat Frequency* be *Auto Calculated*. Keep 2 harmonics. Sweep the *power* parameter from -100 to 0 in 10 linear steps.
2. Set up a single point PAC analysis at 921 MHz and keep the -25 and -21 sidebands.
3. After running the analysis, from the PAC output window plot the input-referred IP3 curves with 81 MHz as the 3rd order sideband and 79 MHz as the 1st order sideband. The procedure is similar to the mixer IP3 example covered in “IQ Modulator Models” on page 821. Use *Variable Sweep* for the *Circuit Input Power* and -100 for the *Extrapolation point*. Make sure to plot Input Referred IP3. Click the I-output port in the top circuit. You should see -10 dBm as the IP3, just as specified. Figure D-31 shows the IP3 plot. Note that 1st order line indicates the gain is 3dB below the specified gain of 0 dB. That is because not all of the power lies at 1000 MHz-921 MHz = 79 MHz; Some of the power lies at 1000 MHz + 921 MHz = 1921 MHz. Use the bottom test circuit to measure available power gain. The bottom circuit drives the demodulator at the same

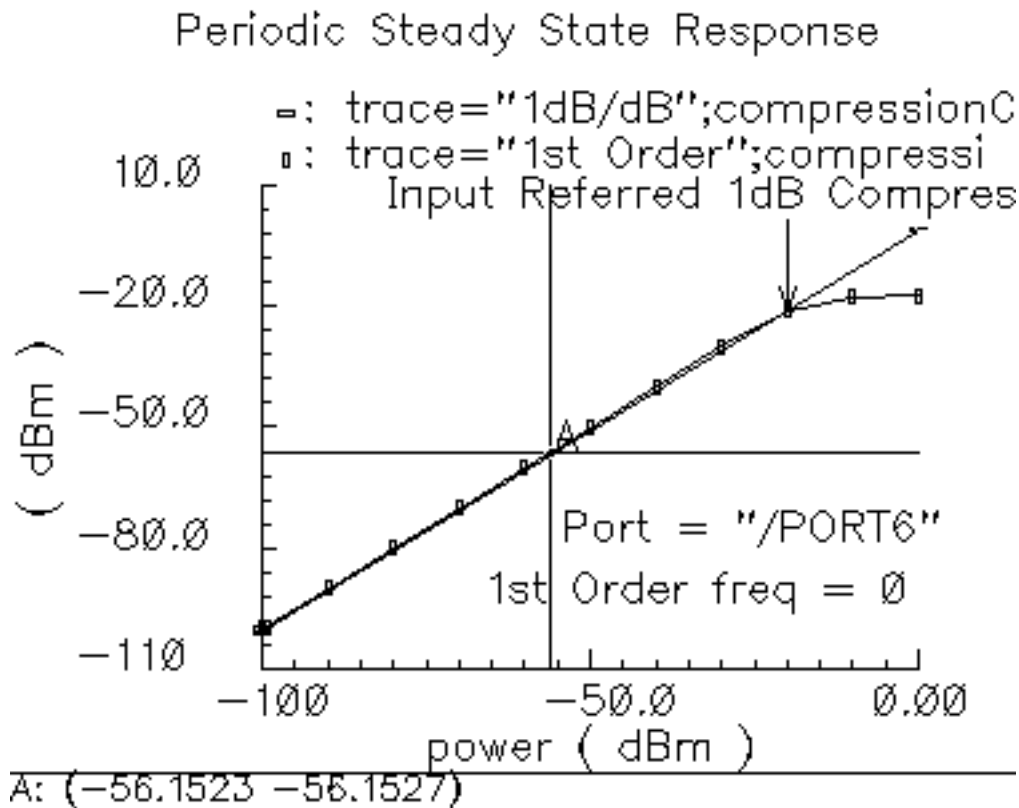
frequency as the demodulator's internal local oscillator, which runs at 1 GHz. Now the output power is not split, it lies in the zero harmonic of the I-output.

### Demodulator IP3



4. Plot the 1dB compression point at the port loading the I-output of the bottom circuit. Use the zero<sup>th</sup> harmonic. The ratio of output to input power should be unity in the linear region. Figure D-32 shows the compression point plot. The measured 1dB compression point is of no use in this test. We want the gain. At low power levels where the gain is constant, the gain is as specified.
5. Remember, in this test circuit the load resistance and output resistance are equal so that the output power is maximal. Also, the input resistance equals the source resistance so that the horizontal axis truly equals input power.

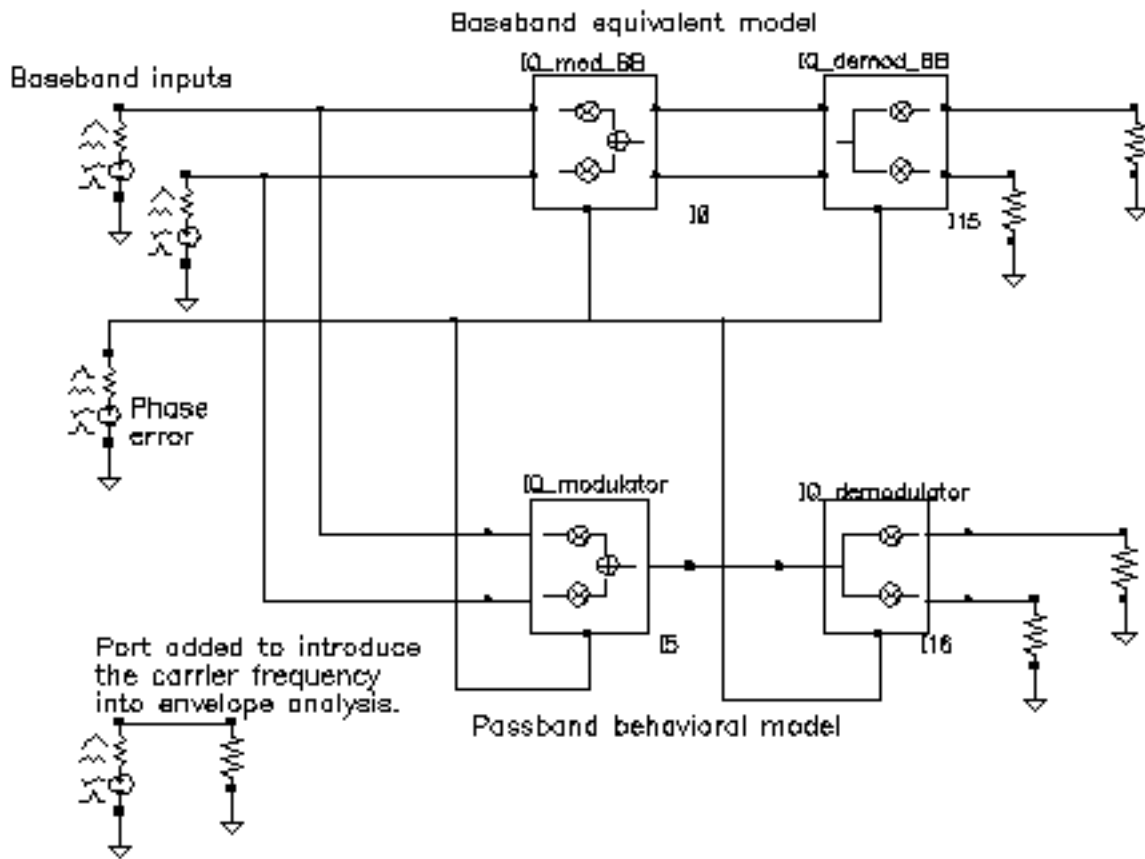
Figure D-32 Demodulator Available Power Gain



Phase errors behave like their counterparts in the modulator models except for a change of sign. Quadrature error behaves exactly as it does in the modulator models. Figure D-32 shows a test circuit for illustrating the relationships between phase error and quadrature error in the modulators and demodulators. The test circuit is called *mod\_demod\_test* and is listed in the *testbenches* category. The test circuit also shows that the passband and baseband models give comparable results, as they should, as long as the passband carrier is not severely clipped. The baseband input trajectory is a complex 1 MHz tone, which produces a circular input trajectory. The demodulator outputs are not matched and are not symmetric with respect to I and Q paths. The modulators and demodulators are not perfectly linear and the non-linearities are asymmetric with respect to I and Q. The modulators and demodulators are driven by the same phase error and the quadrature error parameters are a common variable set to 0.785 radians.



### mod\_demod\_test Circuit



To use the `mod_demod_test` circuit:

1. Recall the circuit and set up a 5 us Envelope analysis with `carrier` as the *Clock Name*.
2. After the analysis completes, plot the `IQ_mod_BB` outputs and make the `I_out` signal the x-axis.
3. Open a subwindow and in it, plot the harmonic time waveforms of the `IQ_modulator` output. Use the first harmonic and plot the real and imaginary waveforms. Make the real waveform the x-axis.
4. Open a third subwindow and stretch the Waveform Display window so that the third subwindow appears below the first window.
5. Plot the time waveforms at the `IQ_demod_BB` outputs and make the `I_out` waveform the x-axis.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

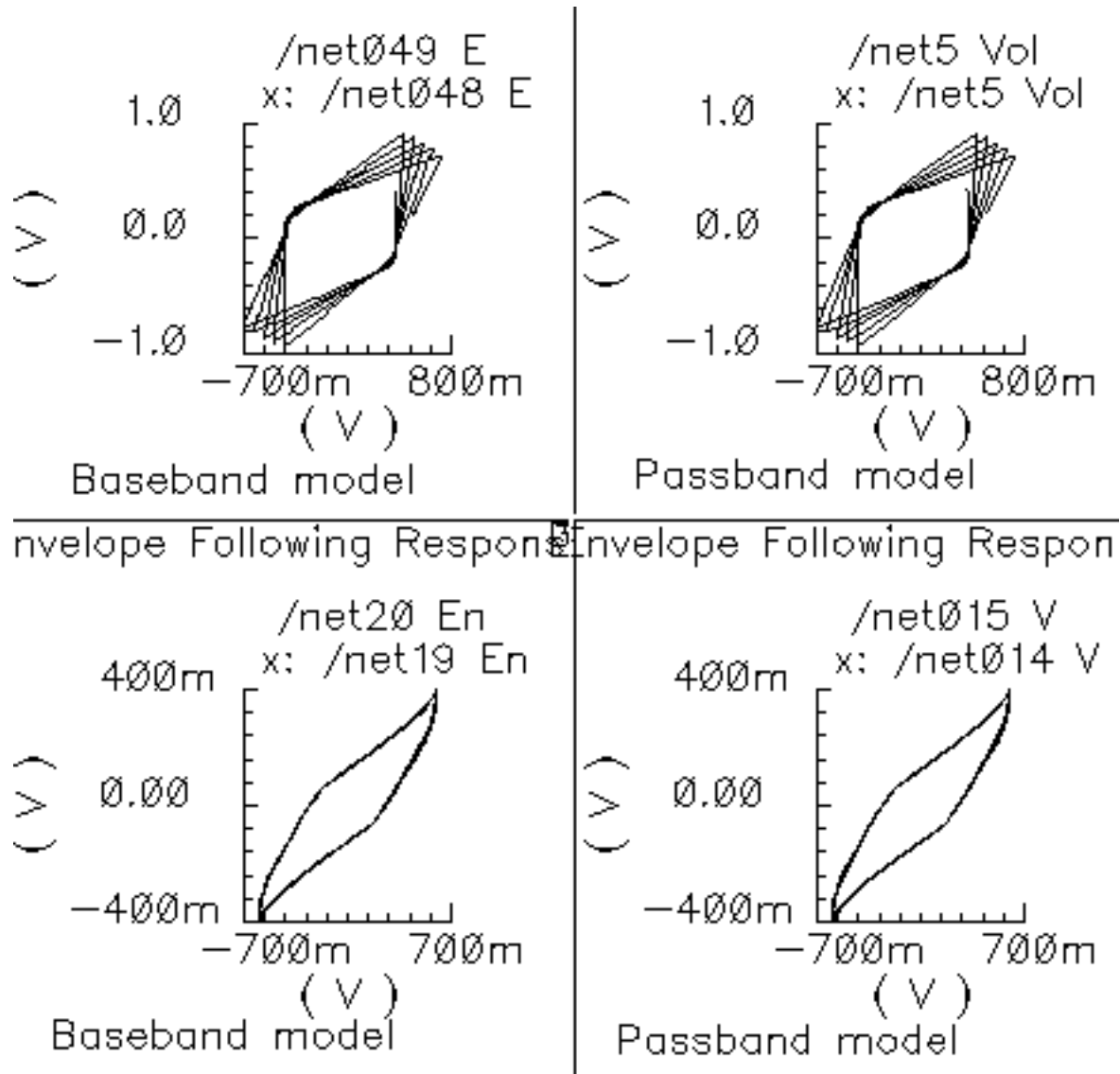
### The RF Library

---

6. Open a fourth subwindow and plot the harmonic time results at the `IQ_demodulator` outputs but this time use the zero<sup>th</sup> harmonic and only plot the real parts. Make the `I_out` waveform the x-axis. Figure [D-33](#) shows what you should now see.

The leftmost pictures are from the baseband models and the rightmost are from the passband models. Passband and baseband models agree quite well. The top pictures are the voltages at nodes that lie between the modulator and demodulator. Quadrature error squashes the baseband trajectory at that node. The trajectory precesses because phase error ramps up linearly with time just like in the last test. The non-linearities produce the sharp corners. The bottom trajectories do not precess because the same phase error rotates the demodulator output in the reverse direction; driven by the same phase error ramp, the demodulator undoes the precession introduced in the modulator. The demodulator outputs are nearly in phase because the quadrature errors of  $\pi/4$  in the modulators and demodulators add to give a total quadrature error of  $\pi/2$ , which in this case puts the baseband I and Q outputs nearly in phase with each other.

Figure D-33 mod\_demod Results



## RF-to-IF and IF-to-RF Mixers

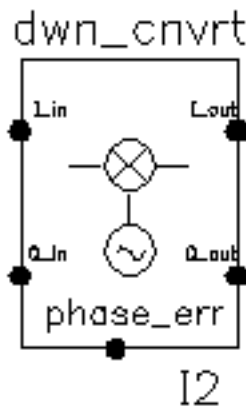
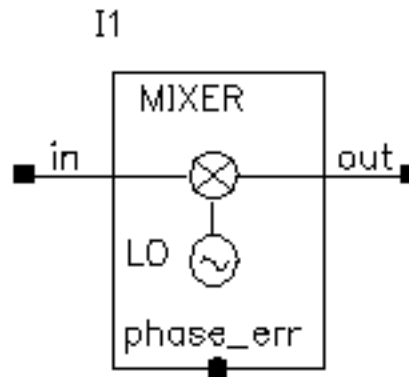
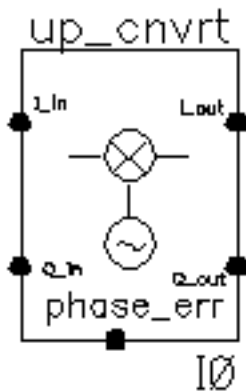
(passband = MIXER\_PB, baseband = dwn\_cnvrt and up\_cnvrt)

**Figure D-34 Baseband and Passband Mixer Models**

Single-ended  
baseband symbol  
(verilog)

Differential  
passband symbol  
(veriloga\_PB)

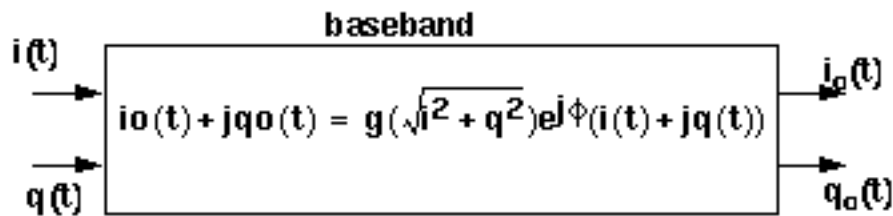
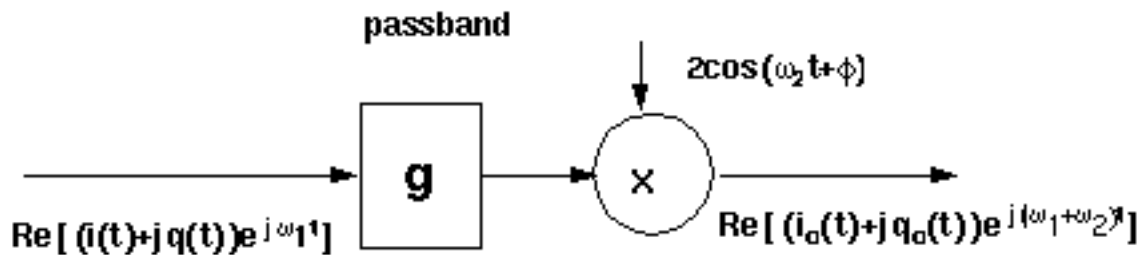
Single-ended  
passband symbol  
(veriloga)



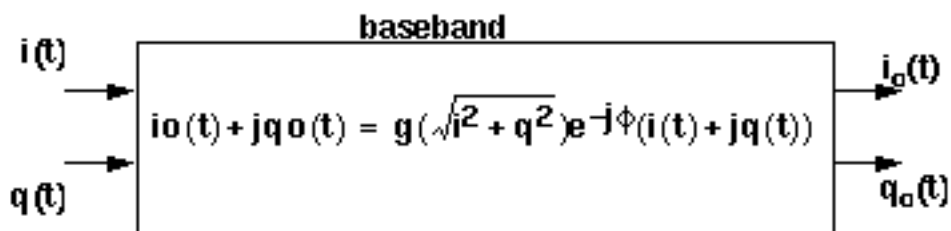
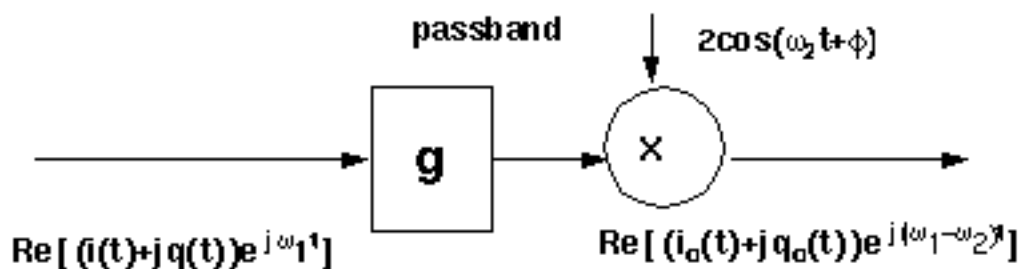
MIXER\_PB is a passband model that converts RF to IF and IF to RF. dwn\_cnvrt model is a baseband equivalent model of a mixer used to convert from RF to IF. up\_cnvrt model is a baseband equivalent model of a mixer used to convert from IF to RF. There are some minor

differences in the baseband models that depend on whether conversion is up or down. Figures D-34 and D-35 show what the models do.

**Calculations for up\_cnrt Mixer**



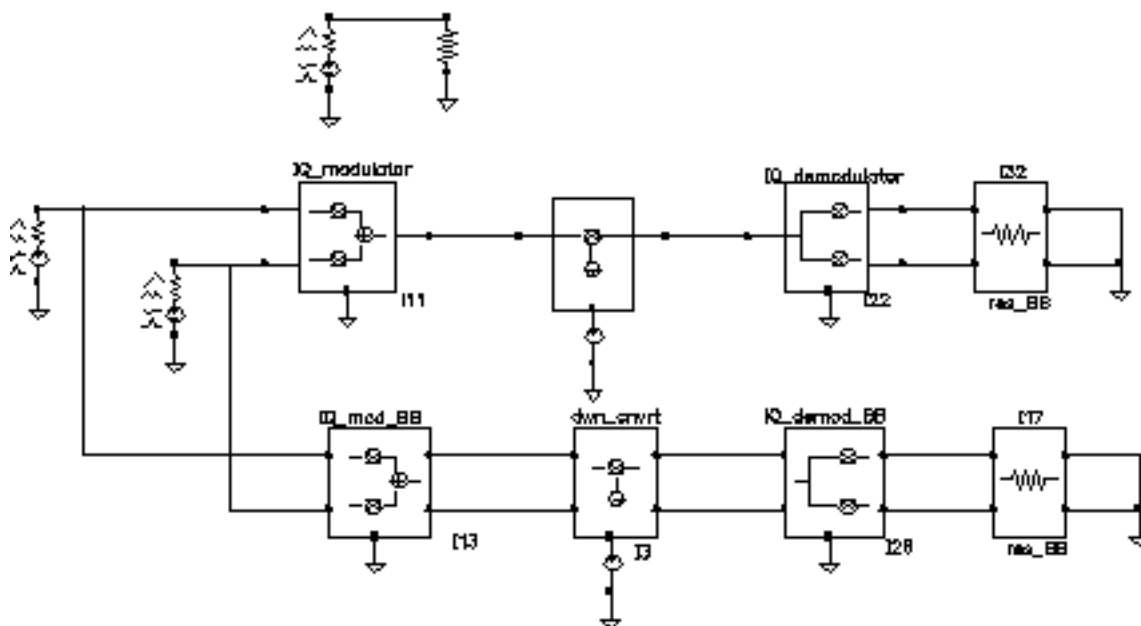
**Figure D-35 Calculations for dwn\_cnrt Mixer**



The noise figure and IP3 parameters are defined in “[Some Common Model Parameters](#)” on page 793. Unlike the `IQ_demodulator`, the IP3 test circuit can be used to define the available power gain because the gain is defined from the input frequency to just one sideband.

Typically the mixer would be used to create an IF stage. In that case, it is difficult to obtain a simple (i.e. filterless) envelope analysis that overlays waveforms to show how well baseband and passband models agree. The test circuit shown in Figure D-35, which is listed as `dwn_cnvt_test` in the `testbenches` category of the `rLib`, shows the relationship between baseband and passband models. The top branch of the circuit consists of passband models. The bottom branch consists of baseband models.

### **dwn\_cnrt\_test Circuit**

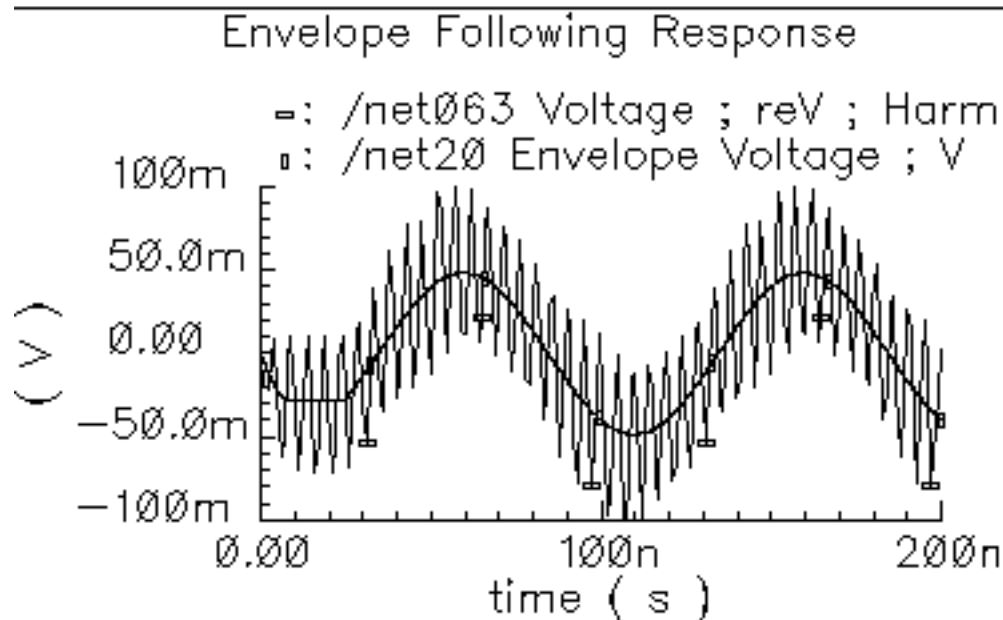
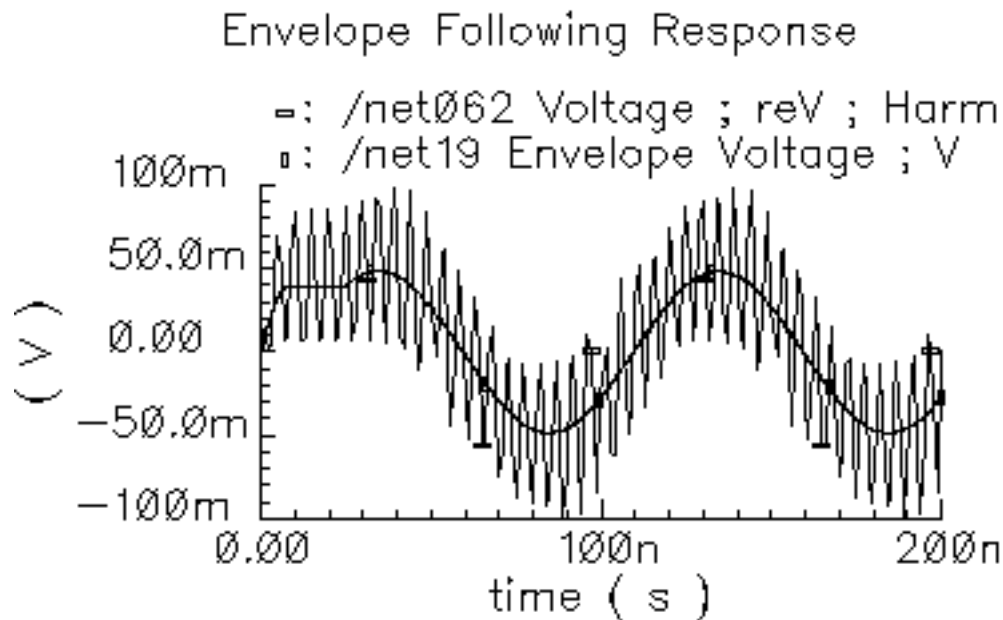


To see the relationship

1. Recall the circuit and set up a 200 ns envelope analysis with `fclock` as the *Clock Name*. Keep 1 harmonics1.
2. After the analysis completes, plot the “time” waveform at the `I_out` pin of the `IQ_demod_BB` model. Append to the plot, the harmonic-time, real part of the zero harmonic of the `I_out` pin on the `IQ_demodulator` model.

3. Open a subwindow and do the same for the  $Q$  outputs. You should now see a picture like the one in Figure D-35.

### Output from an Envelope analysis



Let's trace the input signal through the passband branch to understand these results. A complex baseband 10 MHz tone drives both branches. The modulator's local oscillator is 1 GHz so that the `IQ_modulator` output is at 1.01 GHz. There is no 990 MHz sideband because the input baseband trajectory is a circle ( $= \sin + j\cos$ ), which represents a complex tone. The mixer local oscillator is 900 MHz, which when mixed with 1.01 GHz, produces 110 MHz and 1.91 GHz. The `IQ_demodulator` local oscillator is 100 MHz, which produces 10 MHz, 210 MHz, 2.01 GHz, and 1.81 GHz. The 10 MHz and 210 MHz terms dominate the zero harmonic at the demodulator outputs. The higher frequencies average out to nearly zero. The baseband output is the 10 MHz term and that is what the baseband branch generates, as shown in Figure D-35. A Transient analysis actually runs about 13 times faster than envelope on this circuit. Figure D-36 compares the same outputs using a Transient analysis. The Transient analysis shows that the zero harmonic of the envelope analysis averaged out all frequencies above the envelope clock frequency (1 GHz).

### Testing the `up_cnvr` Mixer

There is a test circuit for the `up_cnvr` model similar to the test circuit containing the `dwn_cnvr` model. The `up_cnvr` model is called `up_cnvt_test` and is shown in Figure D-35. It is also in the `testbenches` category of `rfLib`. The steps parallel those for the `dwn_cnvr` model.

### `up_cnvt_test` Circuit

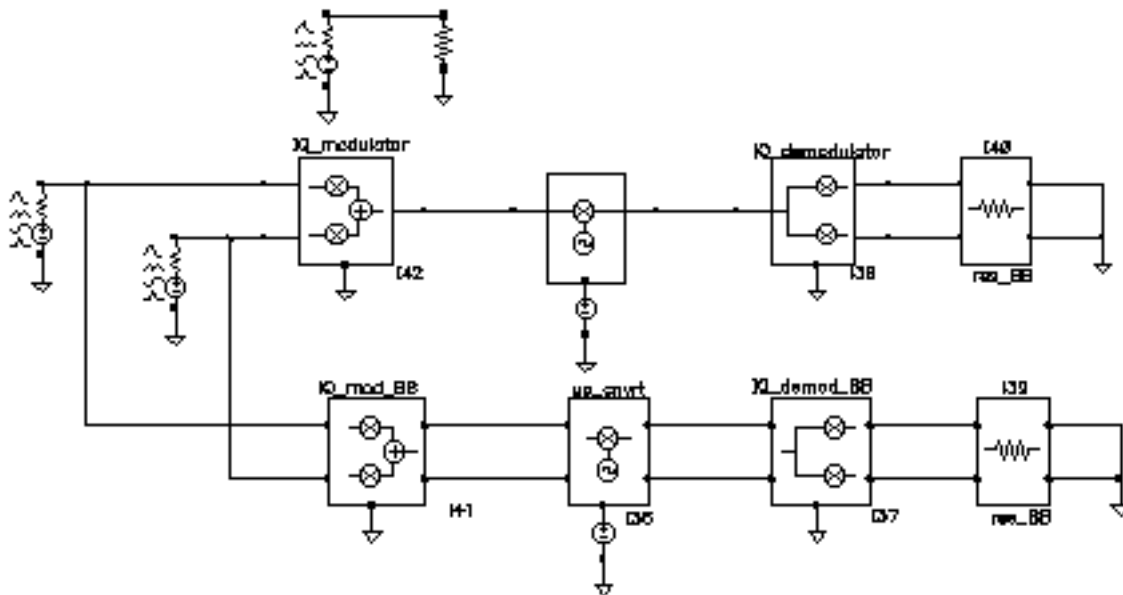
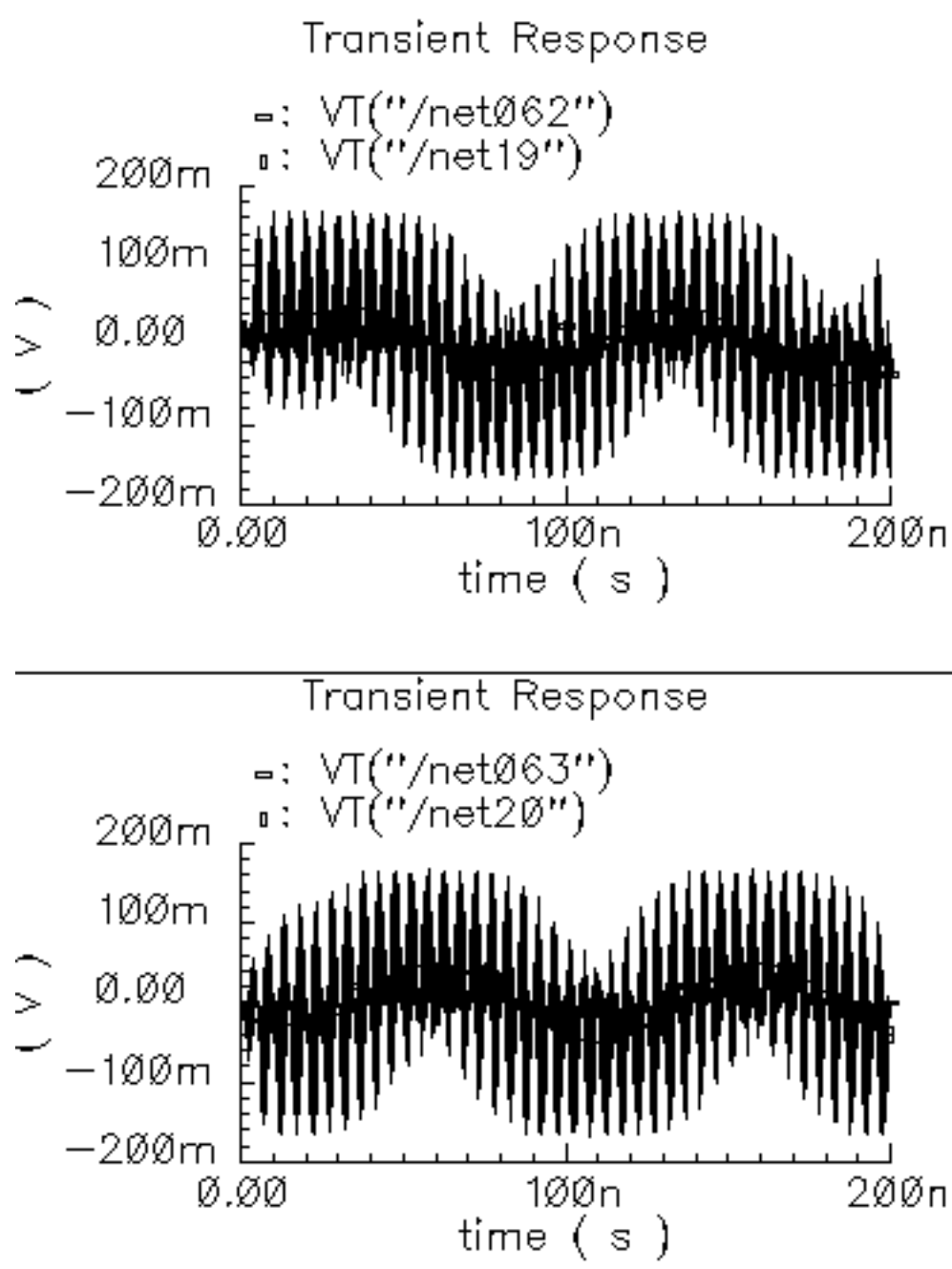




Figure D-36 Results from a Transient Analysis



## Passive Devices

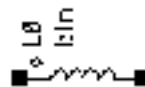
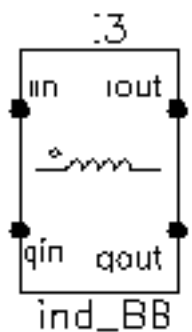
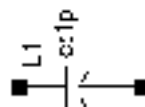
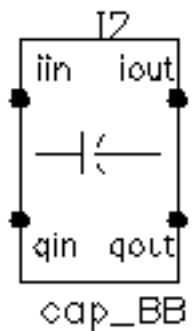
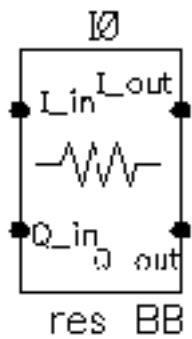
(res\_BB, cap\_BB, ind\_BB)

**Figure D-37 Circuit**

Single-ended  
baseband symbol  
(verilog)

Differential  
passband symbol  
(veriloga\_BB)

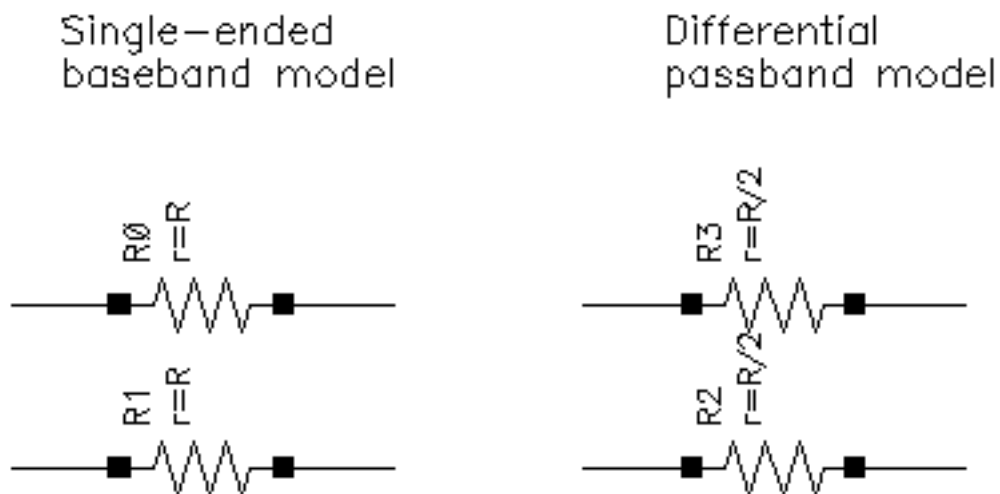
Single-ended  
passband symbol  
(veriloga)



## Resistors

Besides the resistance, the baseband resistor model has a parameter for turning its thermal noise on or off. The baseband resistor is intended for use at a passband node because its noise is doubled. (This was discussed in the section entitled “Relationship between baseband and passband noise”). Figure D-38 shows the symbol, baseband, and passband models. The total noise in the differential passband resistor model equals the noise in one resistor of  $R$  Ohms.

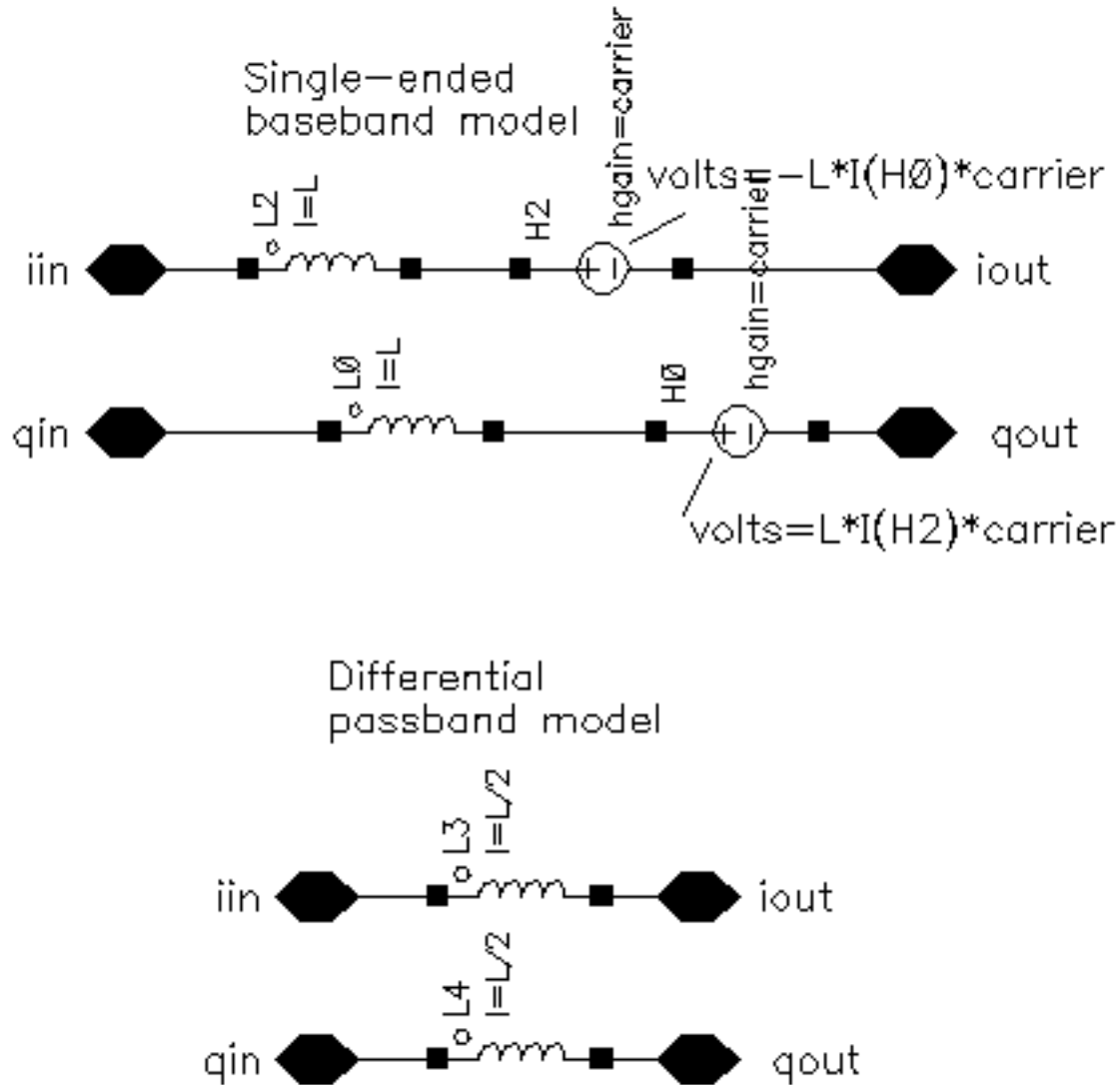
Figure D-38 Resistor Model



## Inductors

The library overview described the basic idea behind baseband modeling of inductors. The baseband inductor model requires one additional parameter besides the inductance, the carrier frequency. Figure D-39 shows equivalent schematics of the baseband and differential passband inductor models. The inductor models are noiseless.

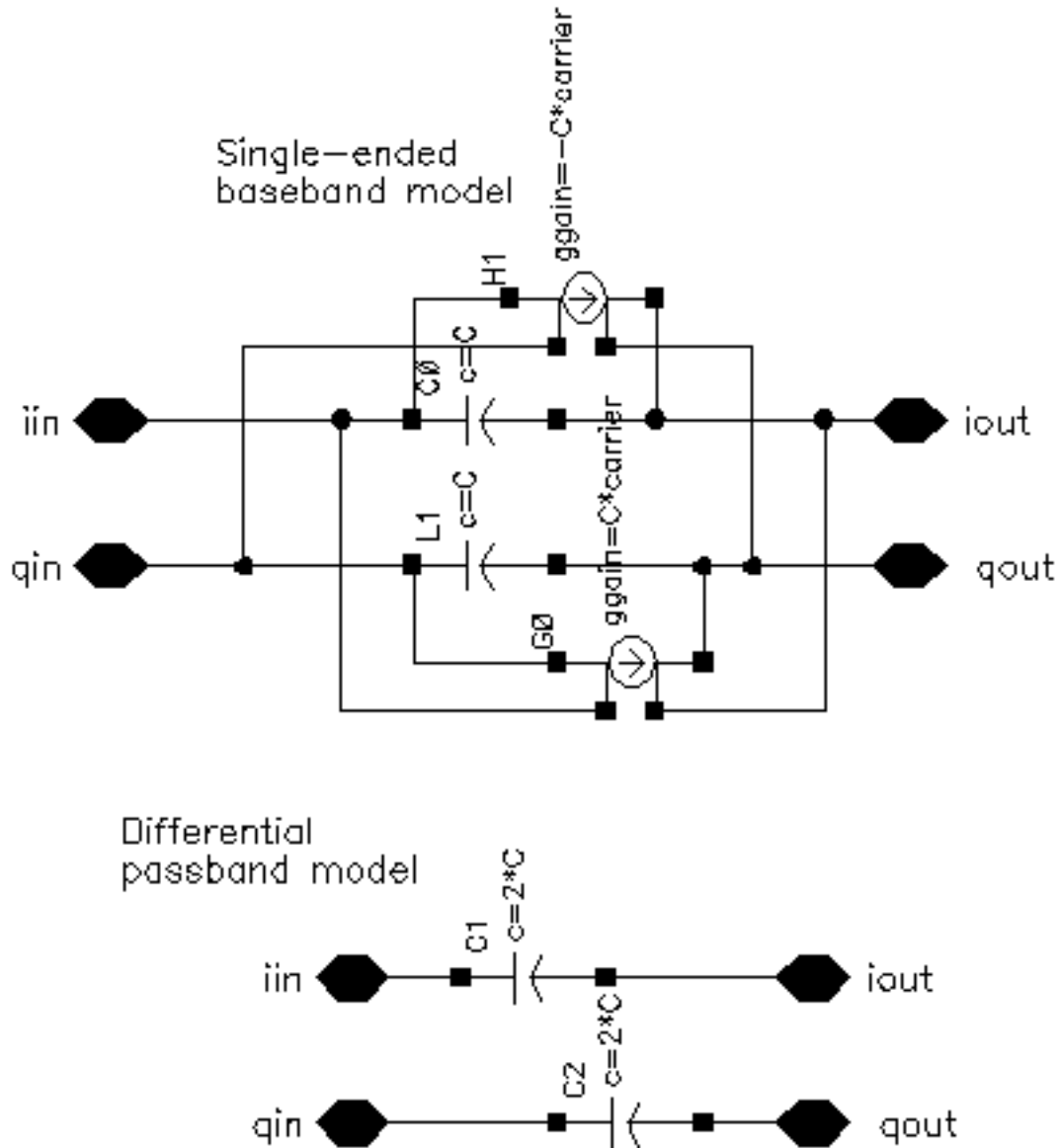
Figure D-39 Inductor Model



### Capacitors

The capacitor is the mathematical dual of the inductor. Figure [D-40](#) shows the baseband and differential passband capacitor models.

Figure D-40 Capacitor Model



### RLC Test Circuits

The two circuits discussed below demonstrate how passband and baseband reactive elements are related. The circuit in Figure D-40 shows a simple passband RLC circuit driven by a modulated carrier. The circuit in Figure D-41 shows the associated baseband equivalent circuit model. The circuits are *PB\_ind\_cap\_test* and *BB\_ind\_cap\_test*. Both circuits reside in the *rfLib* under the *testbenches* category.

### Simple Passband RLC Circuit

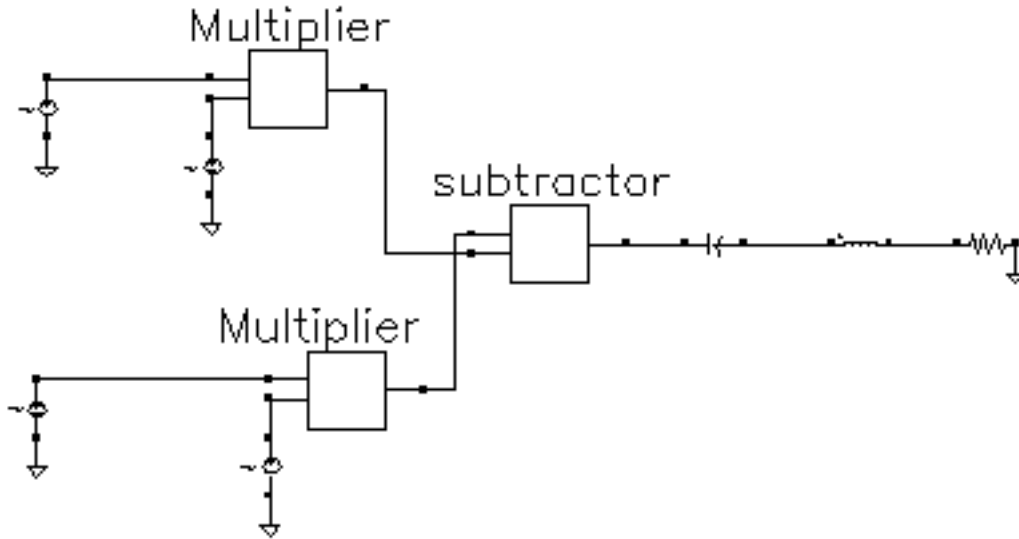
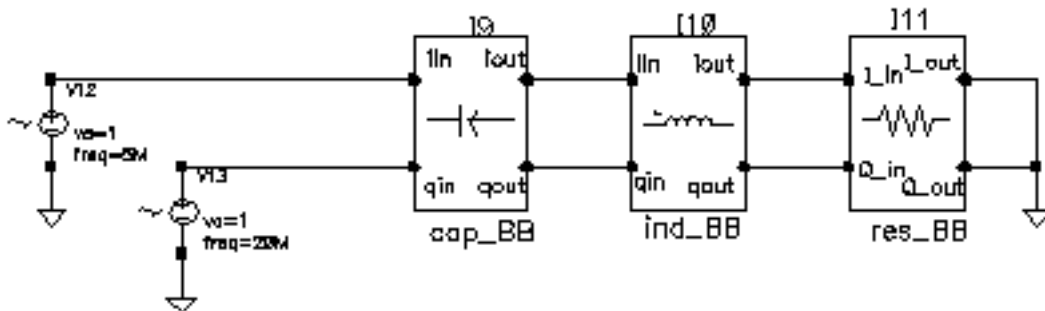


Figure D-41 Baseband Equivalent To Figure 1-63



The following steps explain how to simulate each circuit and overlay the results.

1. Recall the `PB_ind_cap_test` circuit and bring up an analog design environment window. Set up a 200 ns Envelope analysis. Select `carrier` as the *Clock Name*. Set the *Output Harmonics* to 1.
2. Run the analysis and plot the real and imaginary parts of the *harmonic-time voltage* across the resistor. Use 1 for the harmonic number.
3. Recall the `BB_ind_cap_test` circuit and run a 200 ns transient analysis. Note the faster run time. That is the whole point to suppressing the carrier but it is only useful if the results match. Plot the `I_in` and `Q_in` voltages of the resistor model.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

4. To overlay the results, bring up a waveform calculator.
5. Click the *wave* button on the calculator then click one of the Envelope waveforms. If the waveform turns yellow you may have to hit the escape button a few times and click *clear* and *clst* a couple of times in the calculator then try again.
6. Make active the waveform display tool with the transient results then click *Plot* in the calculator.
7. Repeat the last two steps for the other Envelope waveform. You should see the waveforms in Figure [D-41](#). The two models agree very well. The resonant frequency of the series RLC branch is just over 500 MHz. Only by riding on a carrier can the 5 MHz and 20 Mhz baseband signals propagate to the resistor at their original voltage levels. The baseband model accurately predicts the effects of the RLC circuit on the baseband signal. There are two effects, one due to phase shift at the carrier frequency and one due to filtering of the baseband signal itself.
8. In the waveform display tool that overlays the results, change the x-axis to be one of the I-signals. You should get the picture shown in Figure [D-42](#). The tilt in the resulting Lissajous plot indicates phase shift at the carrier frequency but not at the baseband frequencies. The aspect ratio of the Lissajous figure indicates the 20 MHz component is attenuated more than the 5 MHz component. The baseband model captures both effects well.

## Waveforms

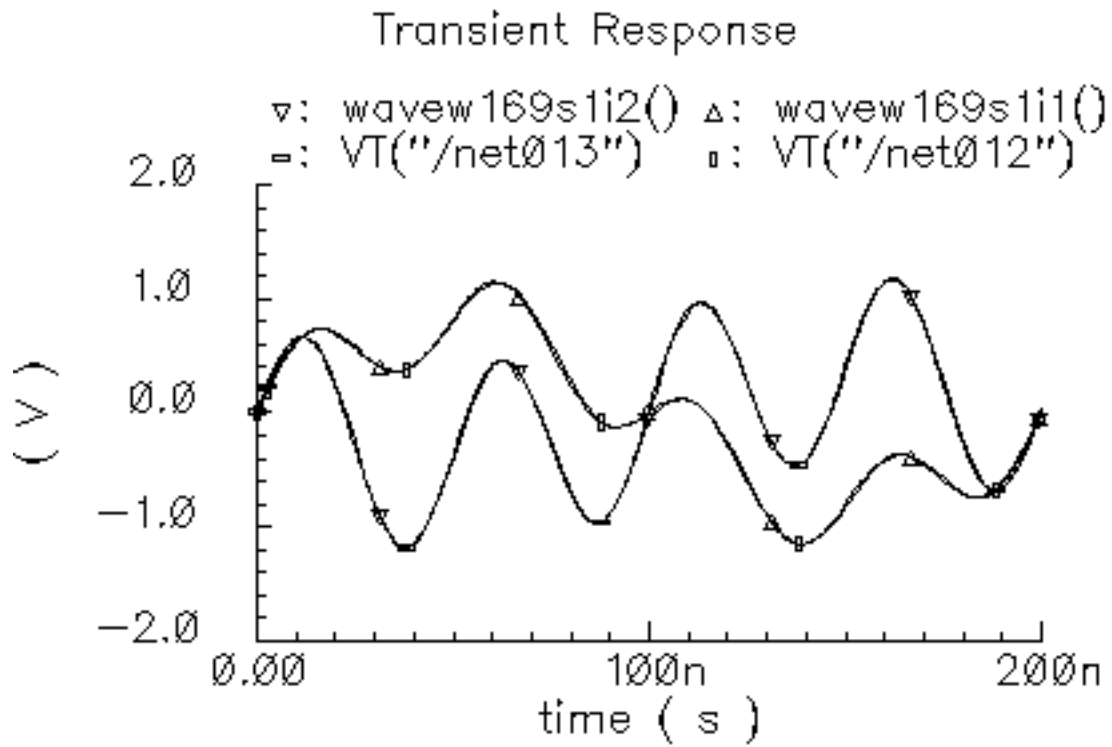
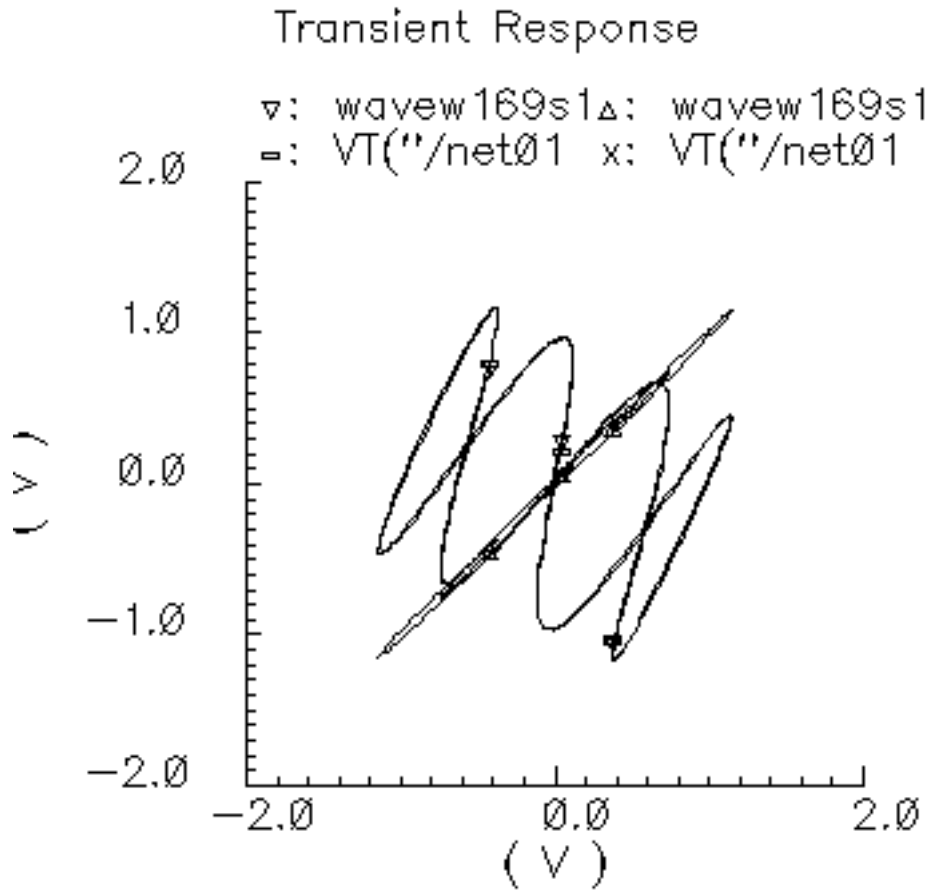




Figure D-42 Lissajous plot



## Linear Time Invariant Filters

Figure D-43 Butterworth Filters

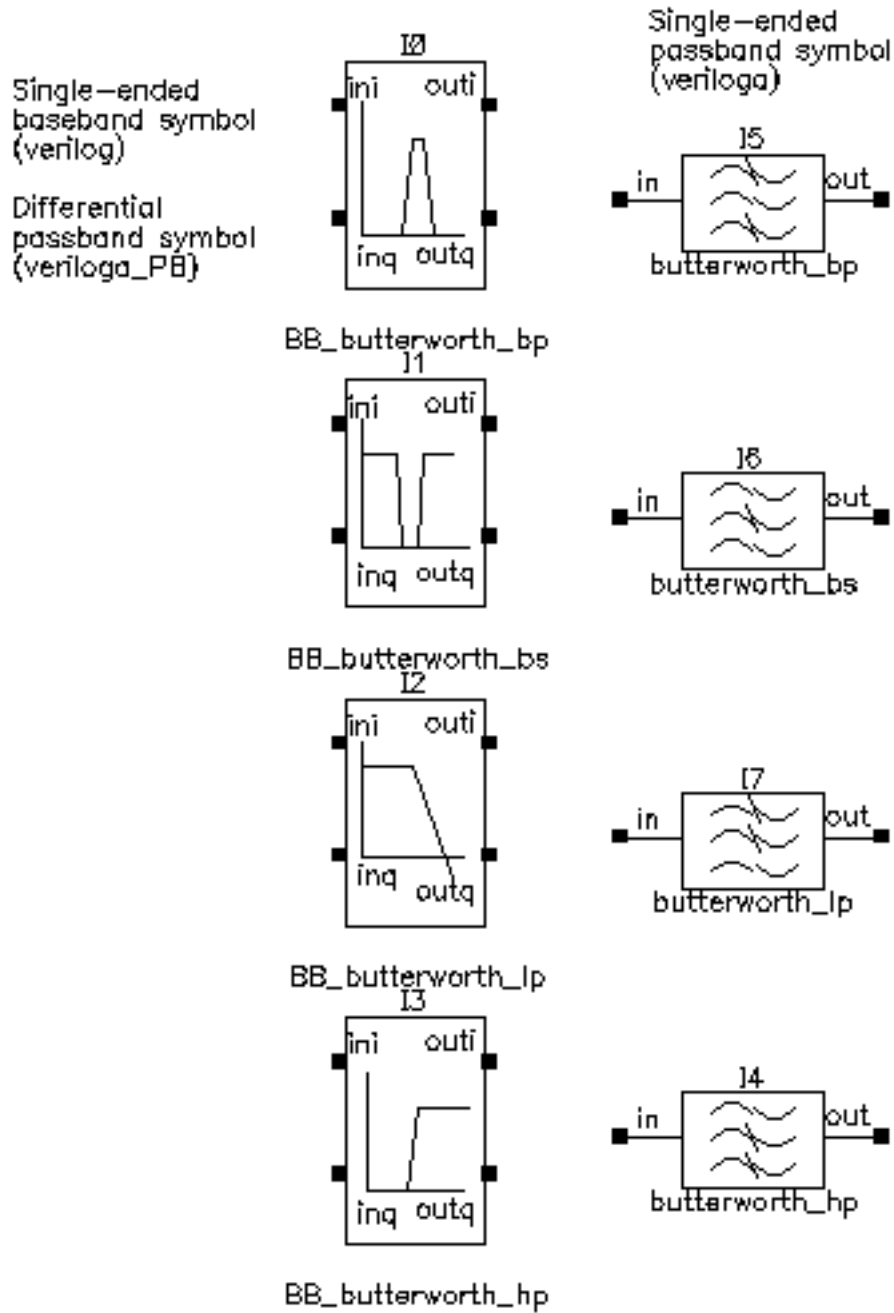


Figure D-44 Chebyshev Filters

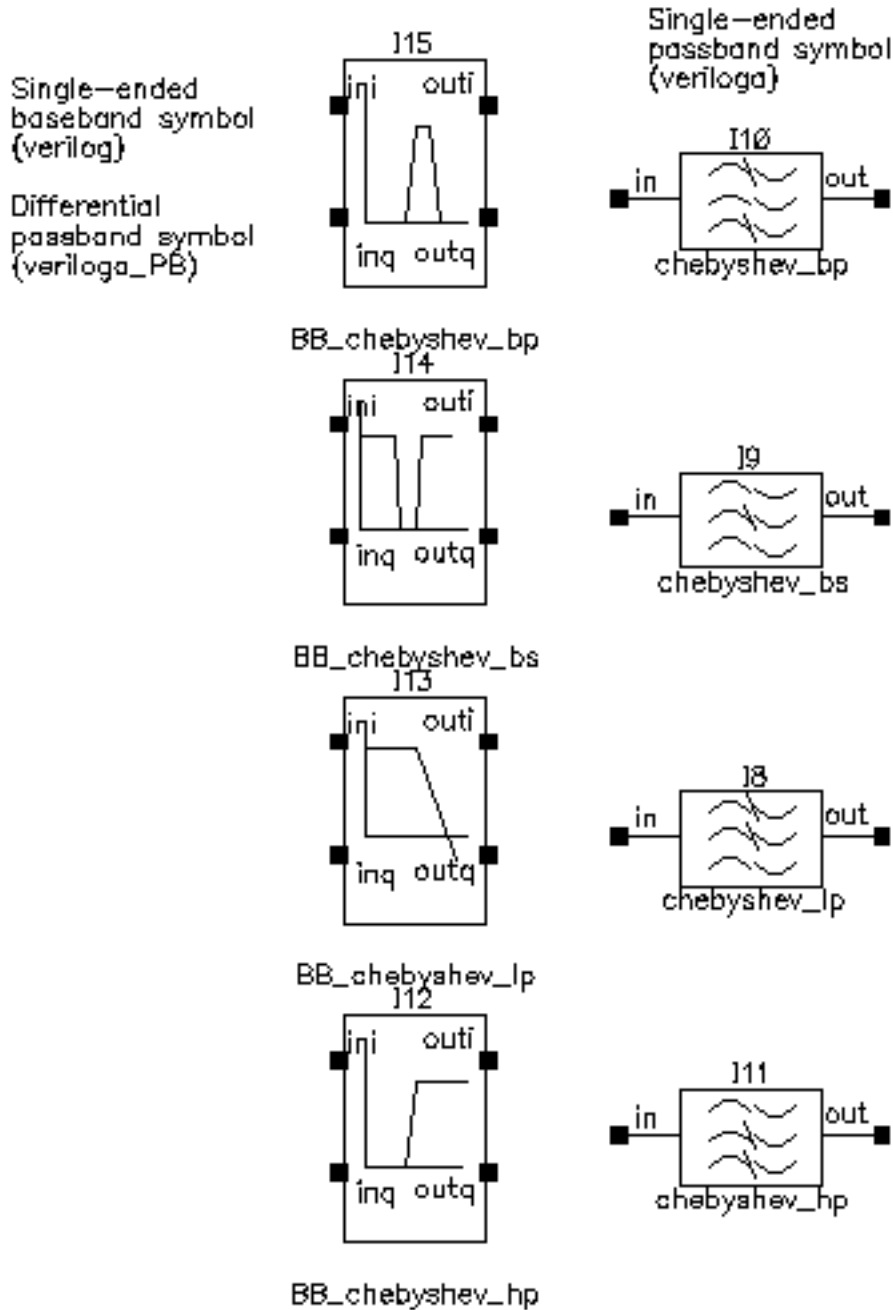


Figure D-45 Butterworth Filters (Laplace Implementation)

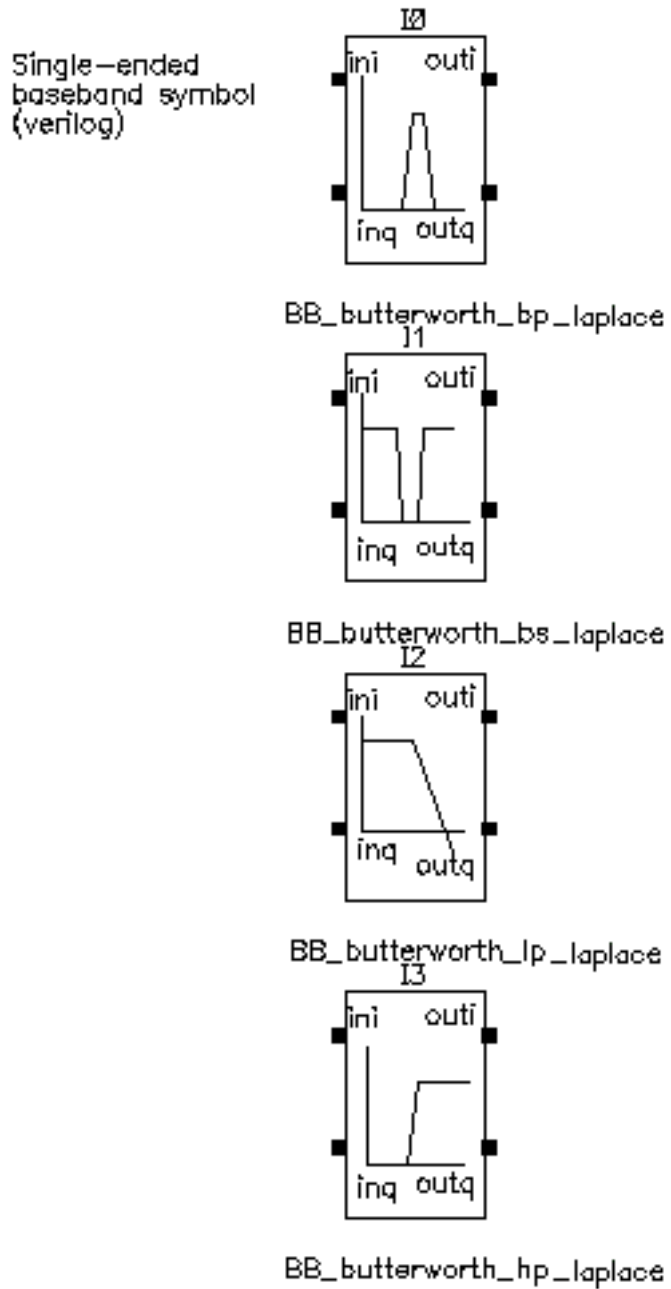
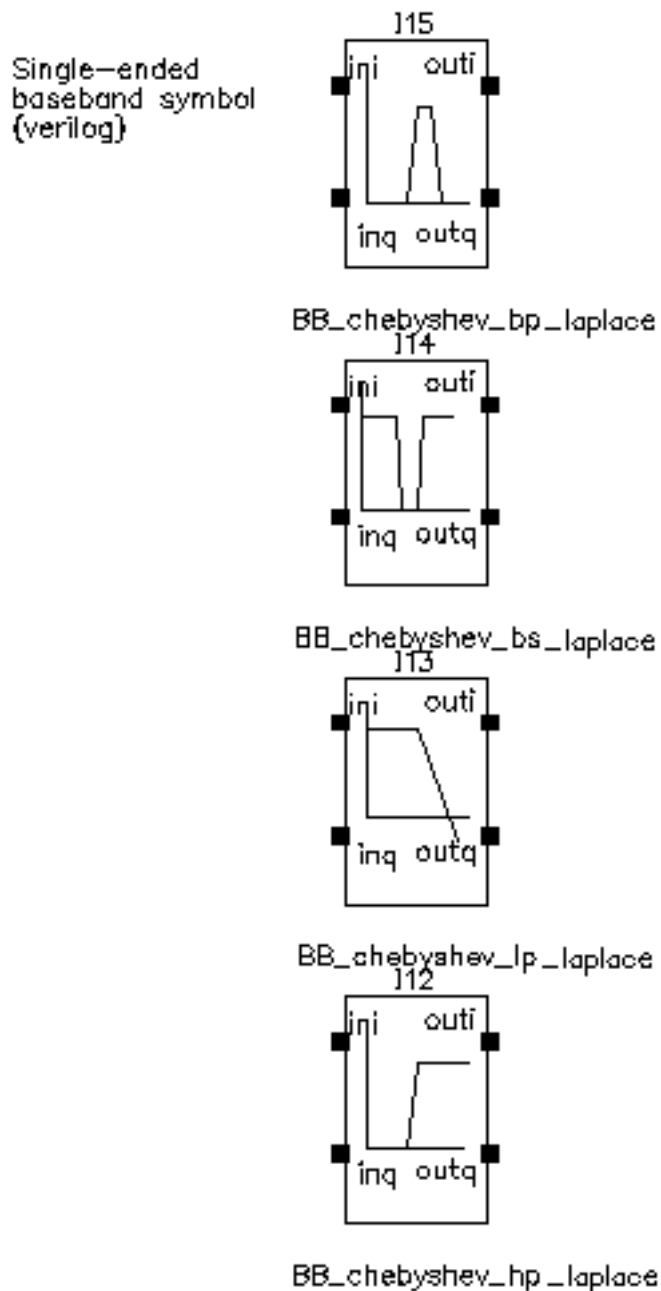


Figure D-46 Chebyshev Filters (Laplace Implementation)



The single-ended passband filters are the filters in the original RF AHDL library described in Appendix D. The single-ended baseband filters are baseband versions of the filters described in Appendix D. The passband views of the baseband equivalent filter models are differential versions of the filters described in Appendix D.

All filter models are LC ladder networks built up during the netlisting step. The only difference between the single-ended passband and baseband models is that the baseband models use baseband equivalent inductor and capacitor models, which means the baseband filter models have twice as many nodes as their passband counterparts. The only difference in the parameter list is that the baseband models require the carrier frequency. The carrier frequency is independent of all other parameters. It is the frequency at which the baseband signal is referenced. If the input signal rides on a carrier that is offset from the reference carrier, the input baseband signal should be spun at the offset frequency before entering the filter model. This can be done at the preceding mixer stage by ramping the phase\_err pin at a rate equal to the frequency offset.

### **Parameters (All parameters are defined in terms of passband models.)**

Note: The input impedance must be less than or equal to the output impedance to produce the desired filter response.

#### ***Input and Output impedance:***

The input and output impedances are resistive reference impedances used to compute the inductors and capacitors that make up the filter. **The input and output impedances are not terminal impedances.** If you load one side of a filter from the library with a resistor of R Ohms and drive the other side with a frequency-swept current source of 1 Amp, you see a voltage of R Volts across the current source at the passband frequency. (If the filter is a bandstop filter the passband is on either side of the stop band.) The following steps demonstrate this point with the four Butterworth filters.

Bring up a new schematic in any library to which you can write.

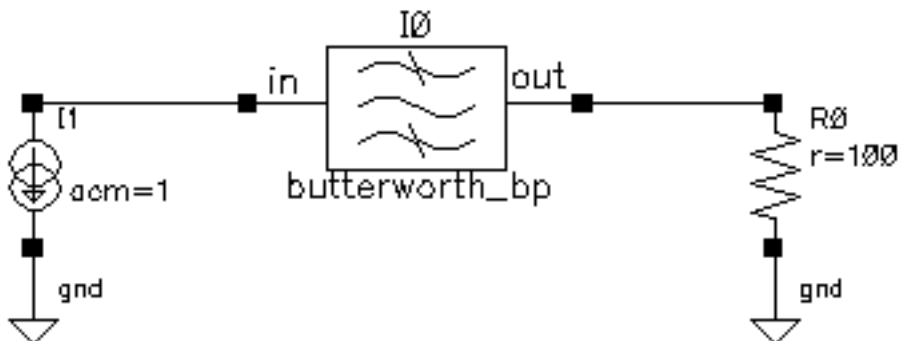
1. Instantiate a butterworth\_bp (bandpass) filter with the parameters as shown in

**Figure D-47 Filter parameters**

<b>Filter Order</b>	3
<b>Input impedance</b>	200
<b>Output impedance</b>	200
<b>Center frequency (Hz)</b>	1e9
<b>Relative bandwidth</b>	0.1
<b>Insertion loss (dB)</b>	0

2. Drive the left terminal (the “input”) of the filter with a DC current source and set the source’s AC magnitude to 1 Amp.
3. Load the right terminal (the “output”) of the filter with a 100 Ohm resistor. The schematic should look like [Figure D-48](#) on page 855.

**Figure D-48 Filter impedance test circuit**



4. Set up an AC analysis. Sweep frequency from 950MHz to 1.05MHz in 100 linear steps.
5. Run the analysis.
6. Plot the voltage across the current source. You should see the voltage across the current source dip to 100 Volts right at the center frequency of 1GHz. At the center frequency,

the impedance looking into a bandpass filter equals the impedance loading the other side, regardless of the filter's input and output impedance parameters.

7. Change the filter to `butterworth_bs` (bandstop filter).
8. Check and save the schematic.
9. In the AC analysis setup, change the AC analysis to sweep frequency from 1Hz to 100GHz in 1 linear step. Do not add any specific points.
10. Run the analysis.
11. Plot the voltage across the current source. You should see 100 Volts at either end point of the sweep. Far away from the center frequency of a bandstop filter, the impedance looking into the filter equals the impedance loading the other side.
12. Change the filter to `butterworth_lp` (low pass filter). Set the corner frequency to 10MHz.
13. Check and save the schematic.
14. Change the AC analysis to sweep from 1Hz to 1GHz logarithmically with 10 points per decade.
15. Run the analysis.
16. Plot the voltage across the current source. Near DC, the impedance looking into a low pass filter equals the impedance loading the other side.
17. Change the filter to `butterworth_hp` (highpass filter).
18. Change the AC analysis to sweep from 1GHz to 100GHz.
19. Run the Analysis.
20. Plot the voltage across the current source. As frequency goes to infinity, the impedance looking into a highpass filter equals the impedance loading the other side.

**How are the input and output impedances used if they do not determine terminal impedances?** Let the filter be specified with input and output resistances of `rin` and `rout` respectively. If you drive the filter with a Port of `rin` Ohms and load it with a Port of `rout` Ohms, an S-parameter analysis produces an `S21` with the shape associated with the filter type and associated specifications. The following steps demonstrate this point.

1. Bring up a new schematic window in a library you can write to.
2. Instantiate a `butterworth_lp` (low pass filter) and set the parameters as shown in [Figure D-49](#) on page 857.

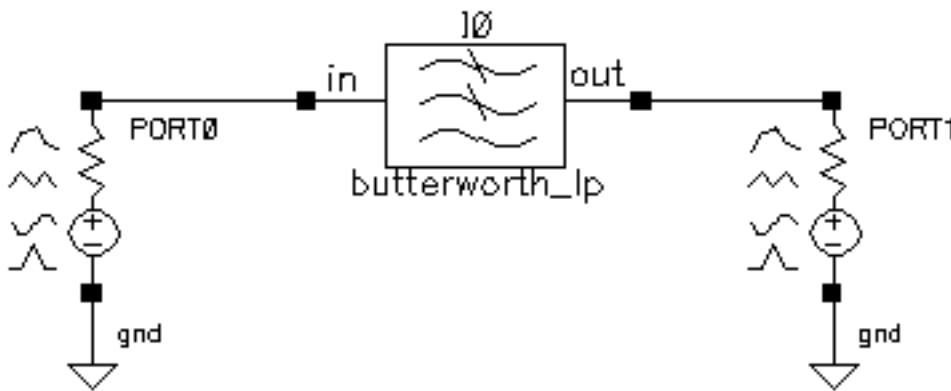


**Figure D-49 Filter parameters**

<b>Filter Order</b>	0
<b>Input impedance</b>	rin
<b>Output impedance</b>	rou
<b>Corner frequency(Hz)</b>	10M
<b>Insertion loss(dB)</b>	0

3. Drive the filter a Port. Set the Port Resistance to rin Ohms and the Port number to 1.
4. Load the filter with another Port. Set the Port Resistance to rout Ohms and the Port number to 2. The schematic should look like [Figure D-50](#) on page 857.

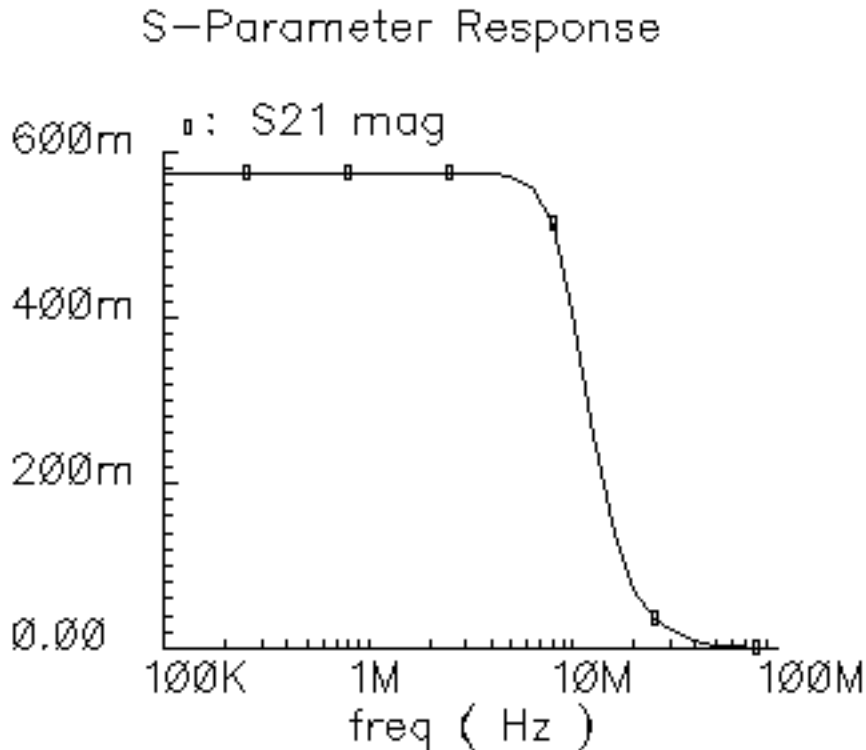
**Figure D-50 S21 test circuit**



5. Bring up an Analog Environment window. Copy variables from the cell view. Set rout to 200 Ohms and rin to 20 Ohms.
6. Set up an sp (s-parameter) analysis. Sweep frequency from 100KHz to 100MHz logarithmically with 10 points per decade.
7. Run the analysis.
8. Plot S21 and you should see the plot in [Figure D-51](#) on page 858. The corner frequency is where S21 drops to 70.71% of its maximum value. If you check, that point is right where it was specified, at 10MHz. Note that the maximum value of S21 is not 1. That is because the input and output reference resistances are not equal. In the passband, the test circuit becomes a voltage divider. Using the basic definition of S21, for this voltage

divider one can show that  $S_{21}$  equals the geometric mean of the input and output reference resistances divided by their arithmetic mean; the maximum value of  $S_{21}$  should equal  $2\sqrt{200+20}/(200+20) = 0.575$ , just as the plot shows.

**Figure D-51 S21 plot**



**Insertion Loss:**

This is the voltage drop in dB, at the passband or center frequency, the filter would suffer if it were driven and loaded with the input and output resistances respectively.

Filter noise is computed from the insertion loss according to:

```
noise_current = 2*sqrt((`db10_real(loss)-1)*1.380620e-23*$temperature/r1);
```

“r1” is the input impedance and “db10\_real()” is a function that maps loss in dB into linear attenuation. The definition is as follows:

```
`define db10_real(x) (pow(10, (x)/10))
```

***Filter Order:***

Low pass-This is total number of reactive elements in the filter. If the order is odd, there is one more capacitor than inductors. If the order is even, there are equal numbers of capacitors and inductors.

High pass-This is the total number of reactive elements in the filter. If the order is odd, there is one more inductor than capacitors. If the order if even there are equal numbers of capacitors and inductors.

Band pass and Band stop- The filter order is the number of inductor/capacitor pairs.

***Corner frequency:***

This is the frequency where S21 is 3dB below the peak response when driven and loaded with the input and output resistances respectively.

***Center frequency:***

For bandpass filters, the center frequency is where S21 is maximum. For bandstop filters, the center frequency is where S21 is a minimum.

***Relative bandwidth:***

This applies to band pass and band stop filters. It is the normalized frequency difference between points straddling the center frequency that correspond a 3dB drop (or rise) from the peak (dip) of S21. The normalization factor is the center frequency. In other words, the relative bandwidth is specified in terms of a fraction of the center frequency. A relative bandwidth of 0.1 and a carrier frequency of 1GHz means that in the case of a bandpass filter, the pass band goes from 950MHz to 1.05GHz with the band edges defined by those points lying 3dB below the peak response.

***Passband ripple:***

This applies only to Chebyshev filters. It is the peak-to-peak variation of the transfer function in the passband when the filter is driven and loaded with the input and output resistances.

## **BB\_Loss**

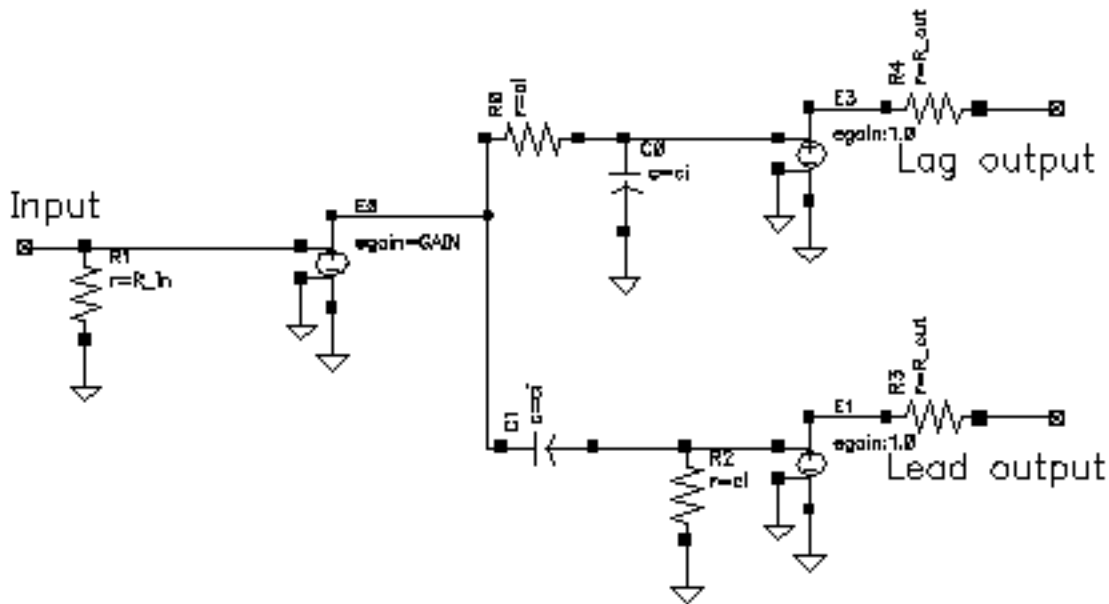
The BB\_loss element is designed to be used with error vector magnitude (EVM) calculations. EVM is defined in terms of an ideal receiver or transmitter. If you want to remove a filter's response from the ideal receiver model while leaving only the passband attenuation, replace the filter with a BB\_loss element and give it the same insertion loss as the filter. There is no passband view or counterpart for this model because a passband EVM analysis is not practical in this release.

## Phase Shifter Splitter

(shifter-splitter)

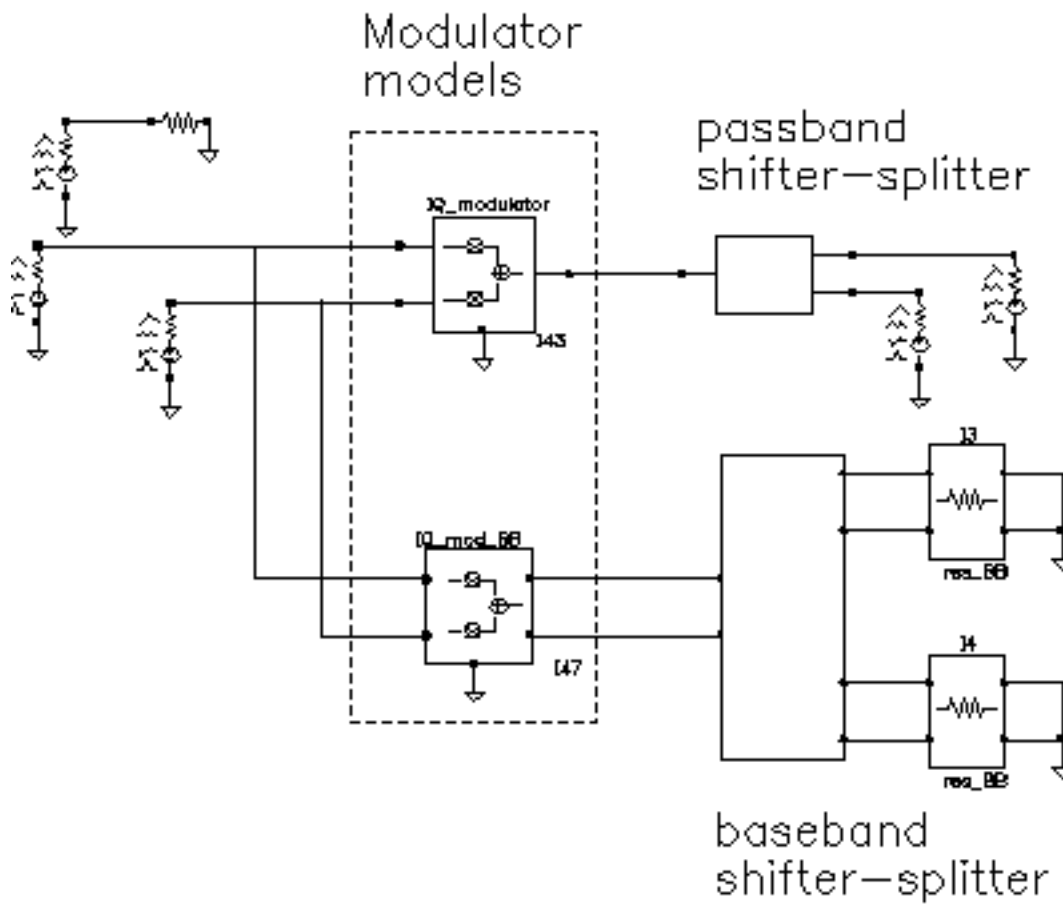
The phase shifter-splitter has one input and two outputs. The outputs are phase shifted versions of the input at the specified frequency. The phase difference between the two outputs is 90 degrees. The phase shifts are accomplished with VerilogA code that does the same thing as the circuit shown in [Figure D-52](#) on page 861. The right-most voltage-controlled-voltage-sources (vcvs) are unity gain buffers. The left-most vcvs is also a buffer but the gain is a user-defined parameter. The input resistance, output resistance, intended operating frequency, and internal resistance are also user-defined. The internal resistance and operating frequency are used to calculate the capacitance necessary to provide  $\pm 45$  degrees of phase shifts at the operating frequency. The baseband view requires the carrier frequency.

**Figure D-52 Phase Shift Circuit**



The test circuit in [Figure D-53](#) on page 862 is for comparing the baseband responses of the passband and baseband equivalent models of the shifter-splitter. The circuit can be found in rLib under the *testbenches* category. It is listed as *shifter\_splitter\_test*.

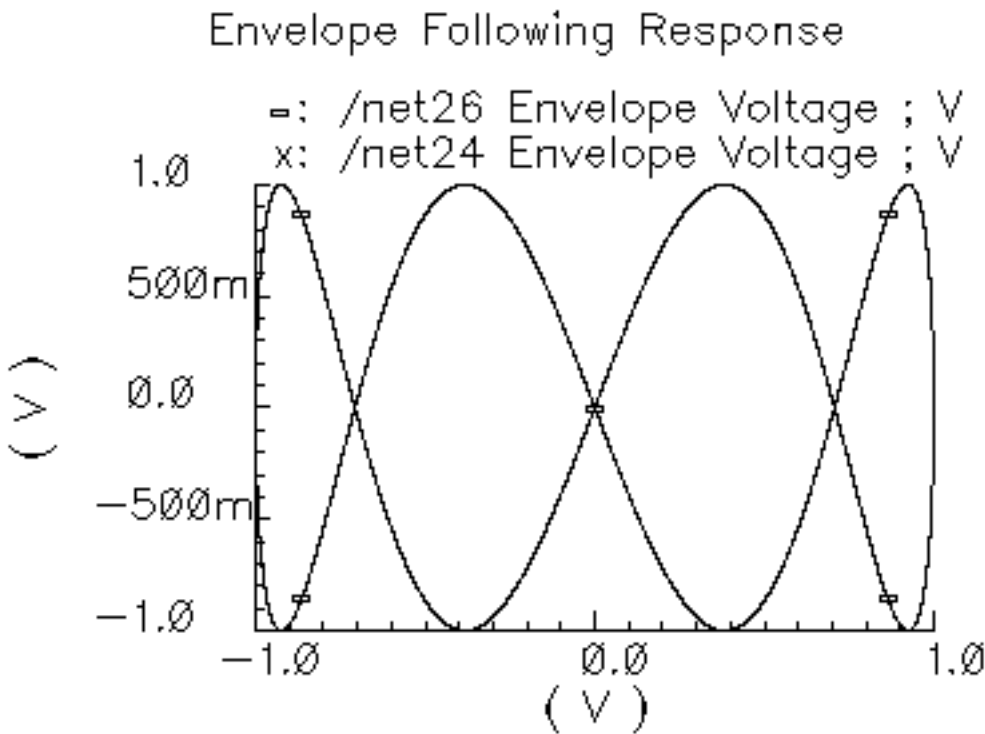
Figure D-53 shifter\_splitter\_test Circuit



The following steps produce a set of Lissajous plots that show what the shifter-splitter does. You observe phase shift in the carrier by observing the tilt of the output Lissajous figures generated by the equivalent baseband signals.

1. Bring up the test circuit in [Figure D-53](#) on page 862 and an Analog Environment window.
2. Set up a 200ns Envelope analysis with “carrier” as the Clock Name. Set the Number of harmonics to 1. Set the Envelope analysis option called *modulationBW* equal to 100MHz.
3. Run the analysis.
4. Plot the “time” waveforms of the two input baseband signals. Change the x-axis to be the I-signal. You should see the Lissajous plot in [Figure D-54](#) on page 863.

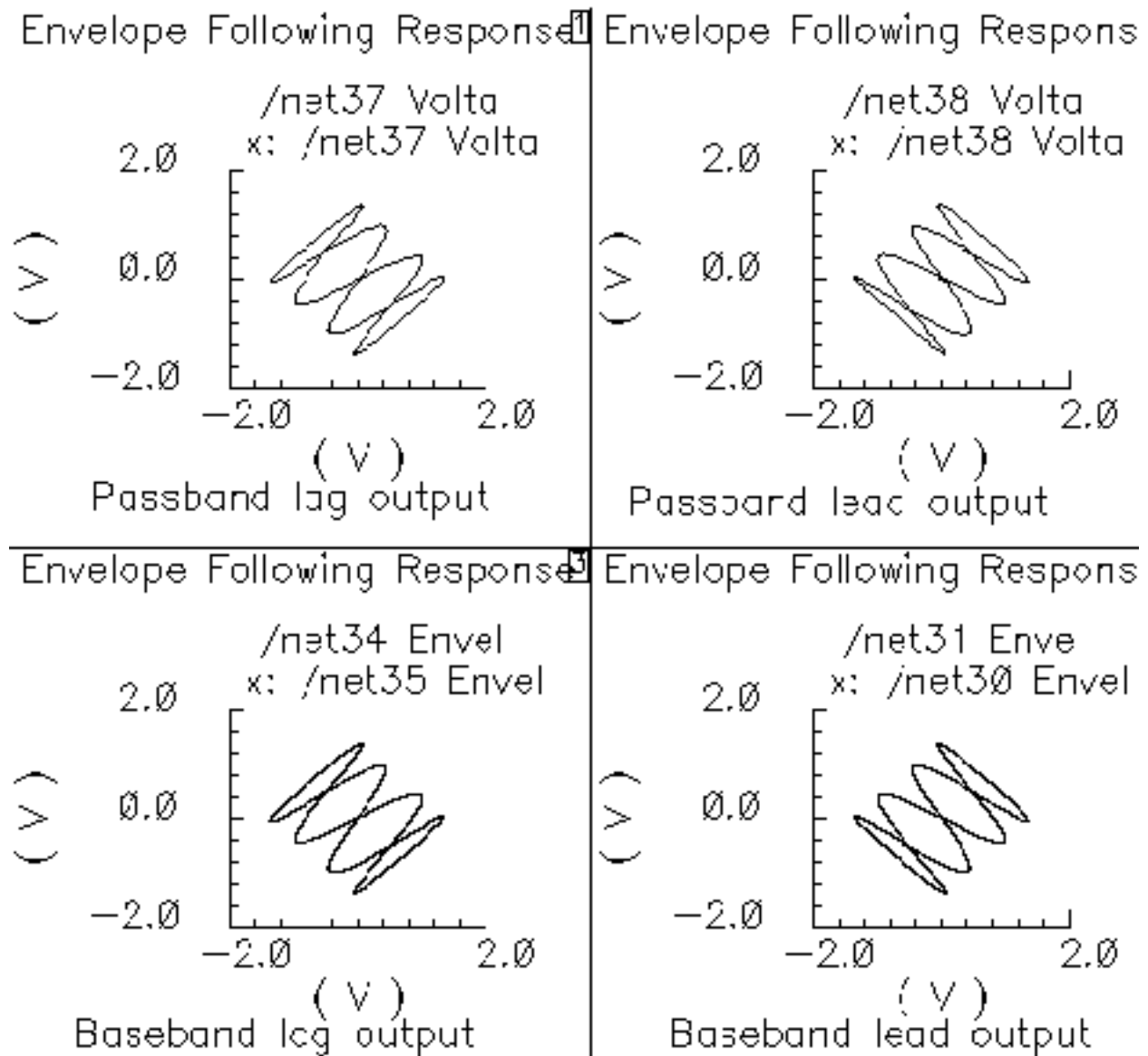
**Figure D-54 Lissajous Plot for Baseband Input Signals**



5. Reset the Waveform Display window and plot the harmonic time, 1 harmonic, real and imaginary parts of the voltage across Port2. Set the x-axis to be the real part. Note that the Lissajous plot is tilted -45 degrees from the one in [Figure D-54](#) on page 863.
6. Add a subwindow.
7. Repeat step 5 for the voltage across Port1. Notice that the Lissajous plot is tilted +45 degrees with respect to the Lissajous plot in [Figure D-54](#) on page 863.
8. Add another subwindow.
9. Plot the “time” waveforms at the lag\_I and lag\_Q outputs of the BB\_shifter\_splitter model. Set the x-axis to be the lag\_I waveform. The Lissajous plot should match the one produced in step 5.
10. Add another subwindow.
11. Repeat step 9 for the “lead” outputs of the BB\_shifter\_splitter model. The Lissajous plot should match the one produced in step 7. Aside from the labels, your Waveform Display tool should look like [Figure D-55](#) on page 864. The time-results of the baseband model

faithfully duplicates the passband results but without simulating the carrier. The baseband model can be run with Spectre RF transient analysis.

**Figure D-55 Comparison of Lag and Lead times for Passband and Baseband Models**



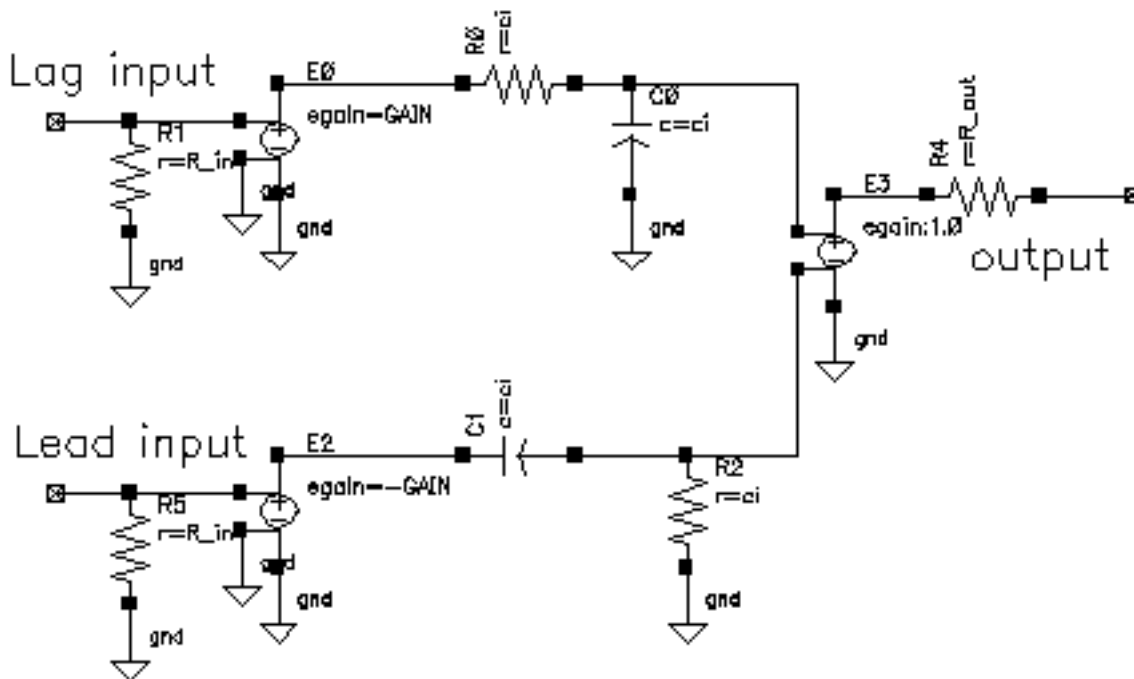


## Phase Shifter Combiner

(shifter-combiner)

The phase shifter-combiner has two inputs and one output. The inputs are phase shifted by +/- 45 degrees then added together to form the output. All terminals are buffered and have the specified terminal resistances. The phase shifts are accomplished with VerilogA code that does the same thing as the circuit shown in [Figure D-56](#) on page 865. The gains of the left-most voltage-controlled-voltage sources are user-defined. The input resistance, output resistance, intended operating frequency, and internal resistance are also user-defined. The internal resistance and operating frequency are used to calculate the capacitance necessary to provide 45 degrees of phase shifts at the operating frequency. The baseband view requires that the carrier frequency be specified.

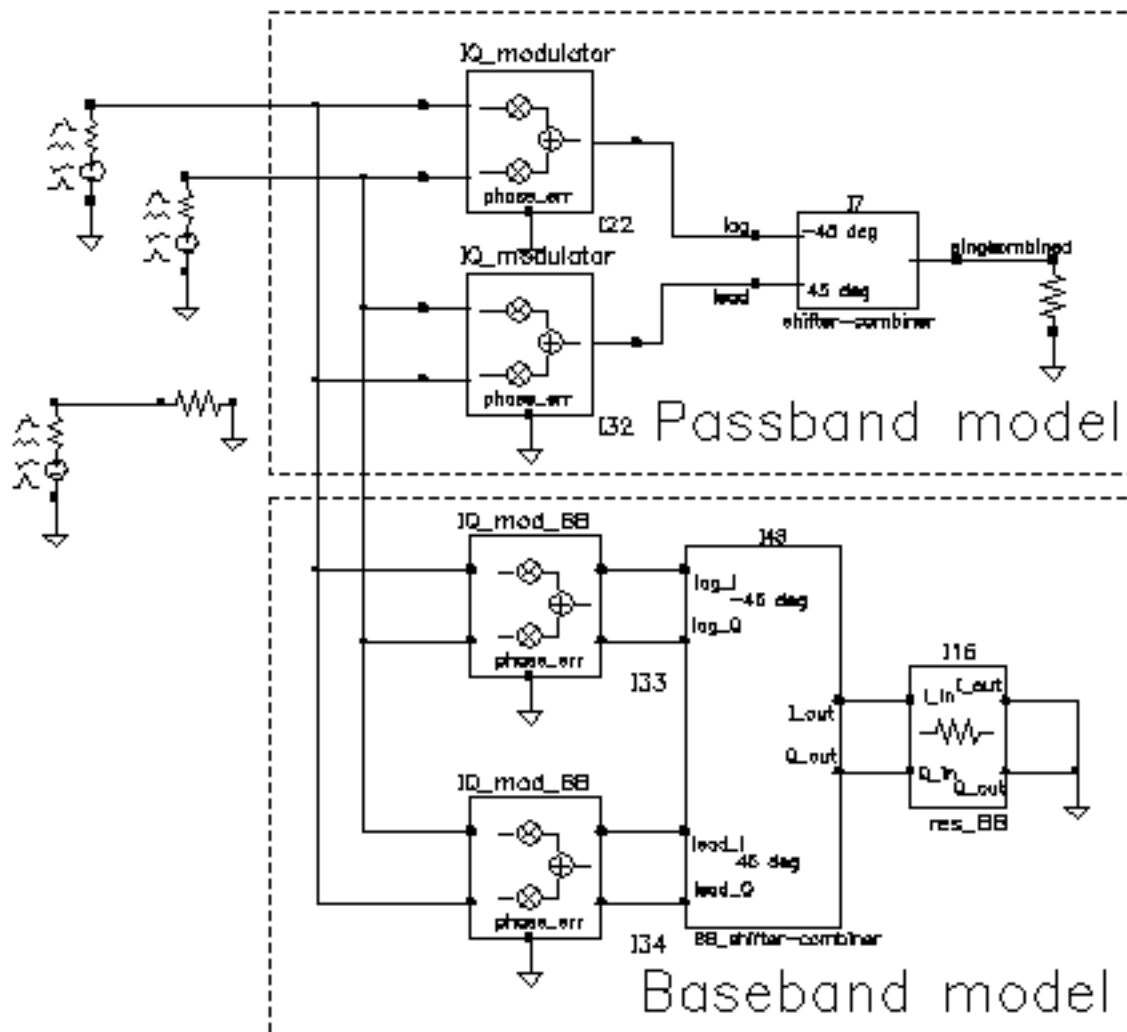
**Figure D-56 Phase Shift Circuit**



The shifter-combiner can be used to eliminate one phase of the carrier. The test circuit in [Figure D-57](#) on page 866 shows a simple test to demonstrate the idea. The circuit is in the rfLib under the *testbenches* category and listed as *shifter\_combiner\_test*. The top circuit is a passband model and the bottom circuit is the baseband equivalent. Baseband input signals are mixed up to 1GHz then passed into the shifter-combiner. The baseband signal

contains 10MHz and 20MHz components. The modulators and shifter-combiner are arranged to produce only a 20MHz signal riding on the carrier.

Figure D-57 shifter\_combiner\_test Circuit



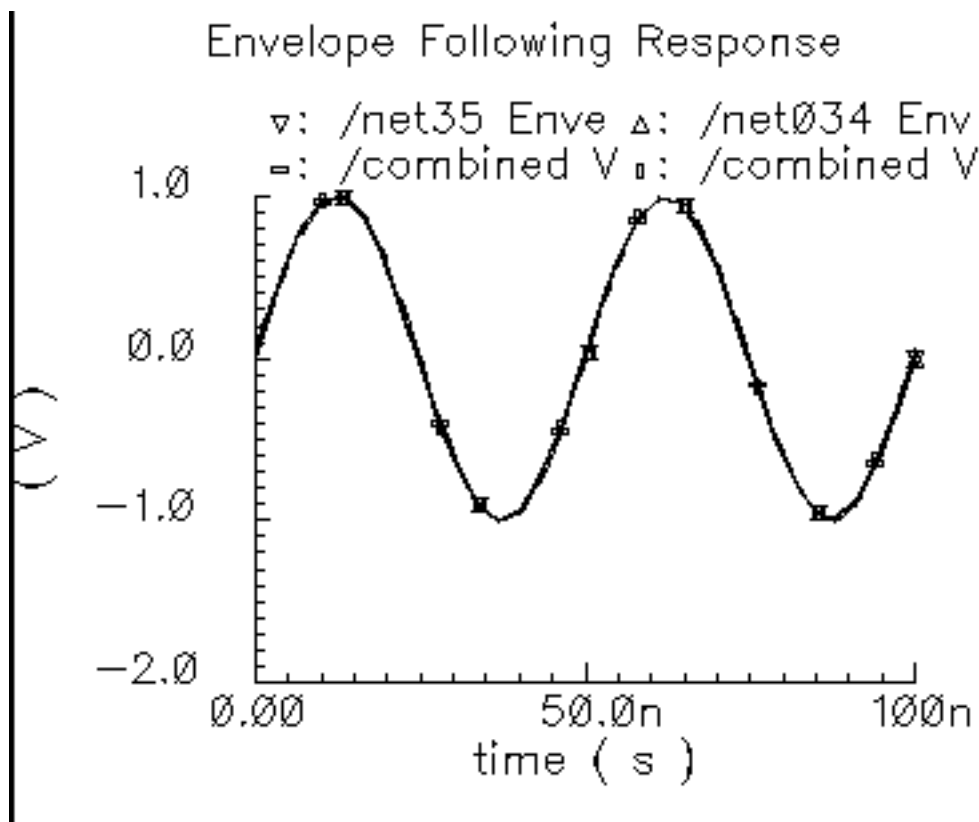
1

To check this assertion:

1. Bring up the test circuit and an Analog Environment window.
2. Set up a 100ns Envelope analysis on the circuit with the Clock Name set to “carrier” and the modulationbw option set to 40MHz. Set the Harmonic number to 1.
3. Run the analysis.

4. Plot the harmonic time, 1 Harmonic, real and imaginary parts of the passband shifter combiner output.
5. Append to the plot, the “time” waveforms at I\_out and Q\_out pins of the BB\_shifter\_combiner model. [Figure D-58](#) on page 867 shows what you should now see in the Waveform Display window. All waveforms are the same and they contain only the 20Mhz baseband signal. The 10Mhz baseband input signal does not propagate to the output.

**Figure D-58 shifter\_combiner\_test results**



One application of the shifter-combiner is an image rejection receiver. [Figure D-59](#) on page 868 shows a very simple example of an image rejection receiver. [Figure D-60](#) on page 868 shows the baseband equivalent model of the receiver. Both examples are in the *rfExamples* directory and are listed as *image\_reject\_rcvr\_PB* and *image\_reject\_rcvr\_BB*. The local oscillator runs at 1GHz and the RF carrier is 1.1GHz, which places the image at 900Mhz. This example shows one of the limitations of the baseband equivalent models.

Figure D-59 A Simple Image Rejection Receiver

Passband model of an ideal image rejection receiver.

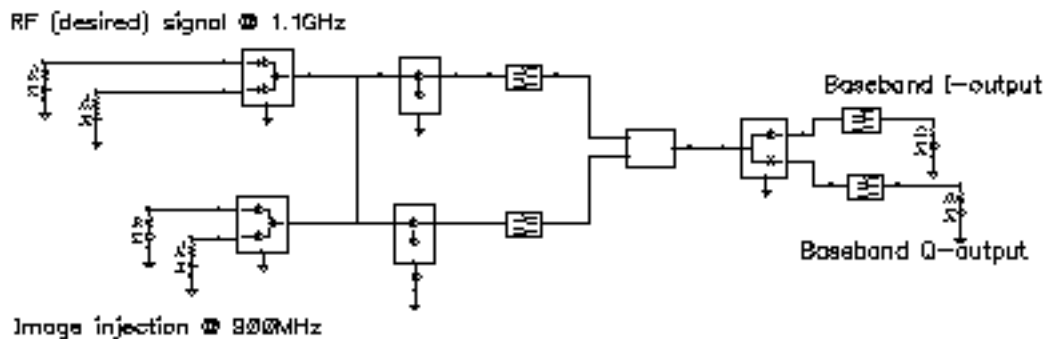
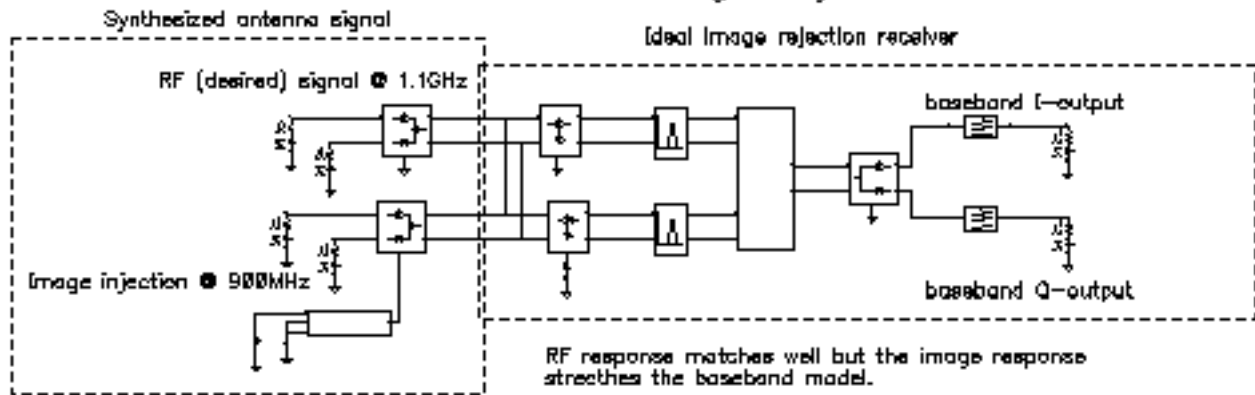


Figure D-60 Baseband Equivalent Model of the Image Rejection Receiver

Based model of an ideal image rejection receiver.



1. Bring up the passband test circuit and an Analog Environment tool.
2. Set up a PSS analysis. You need to add the 1.1GHz, 1GHz, and 900Mhz fundamental tones. Give them arbitrary but distinct names. AutoCalculate the Beat Frequency, which should be 2MHz. You need not save more than the 1st harmonic. Set the PSS maxstep option to 20ps so that it accurately simulates the oscillators hidden inside the Verilog-A modules.
3. Run the analysis.

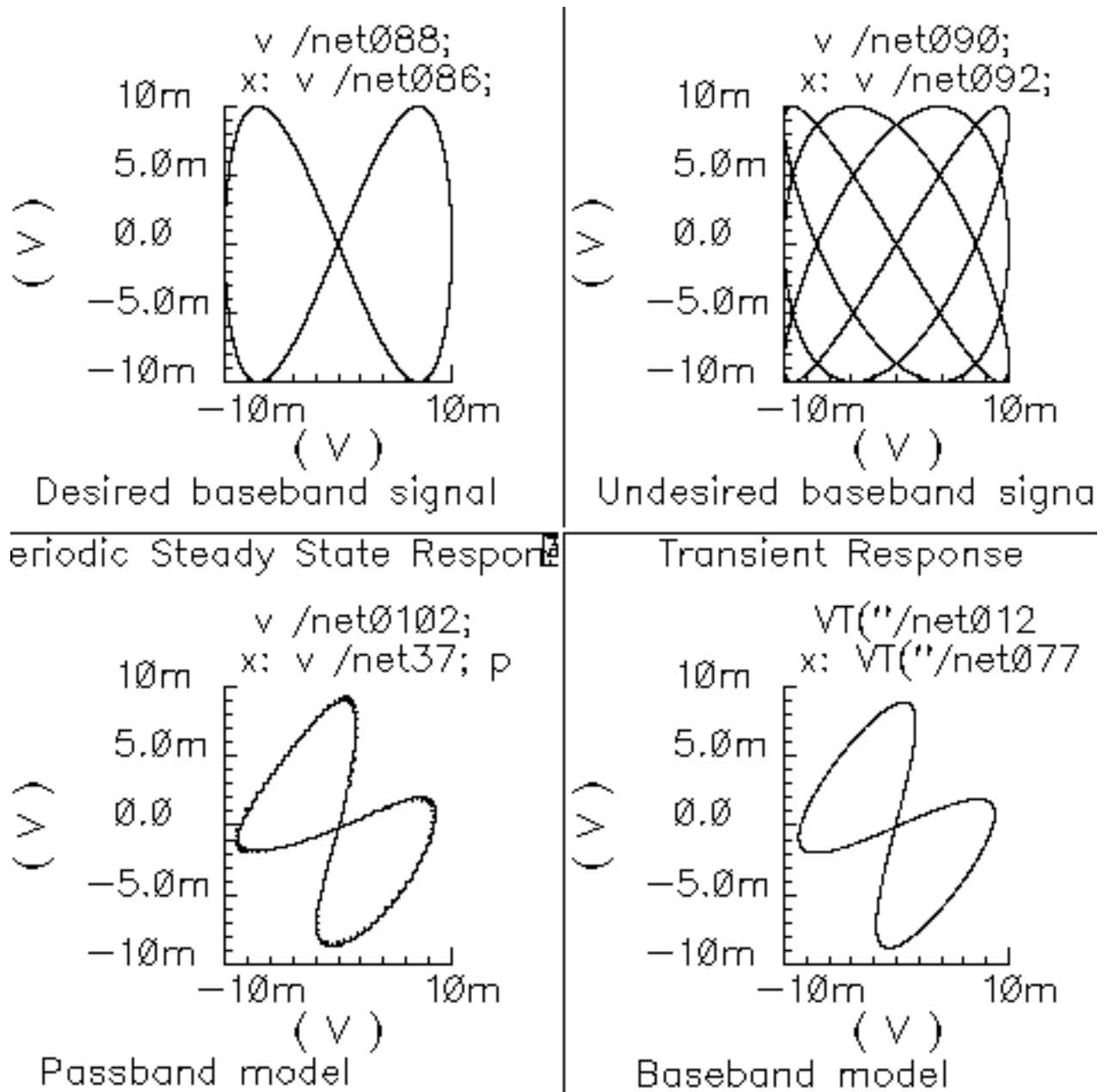
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

4. Plot the voltages across Ports 5 and 6. Set the x-axis to be the voltage across Port 6. This is a Lissajous plot of the desired baseband signal, the one riding on the 1.1GHz carrier.
5. Add a subwindow.
6. Plot the voltages across Ports 8 and 7. Set the x-axis to be the voltage across Port 8. This is a Lissajous plot of the undesired baseband signal, the signal riding on the image of the carrier at 900MHz.
7. Add another subwindow.
8. Plot the I and Q- baseband outputs. Set the x-axis to be the I-output. The Lissajous plot is a tilted version of the desired baseband signal, indicating that most of the image was successfully rejected.
9. Bring up the baseband equivalent receiver model and another Analog Environment tool.
10. Run a 10us Transient analysis with 9.5us as the output start in the analysis options and maxstep set to 250ps. The phase\_err pin on the image signal generator is being driven to spin the output at 200MHz, the frequency difference between the desired frequency and image frequency.
11. Add another subwindow to the Waveform Display tool showing the passband results and make sure it is active.
12. Plot the I and Q baseband outputs from the baseband equivalent receiver model. Set the x-axis to be the I-output. You may need to make the scales on the last two plots are the same. Aside from the labels, the Waveform Display tool should look like [Figure D-61](#) on page 870.

Figure D-61 Lissajous Plots for Baseband Signals



The baseband equivalent receiver model indeed rejects the image but the rejection is over-estimated. If you look closely, the baseband output of the passband model contains more ripple from the image. The over-attenuated ripple in the baseband model is explained as follows.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

---

Recall the rotating reference frame analogy for baseband modeling. With respect to the rotating 1.1GHz reference frame, the image signals rotate counter-clockwise at twice the IF, 200MHz in this case. The lower left block in the baseband receiver model spins the modulator output at -200MHz by ramping the phase error pin. The -200MHz signal propagates through the IF bandpass filters, as it should, because the response of the baseband model of the filter peaks at DC and at minus 200MHz. The trouble occurs in the final downconversion to baseband. In the baseband model, the final low pass filters severely attenuate the -200MHz image signal. However, in the passband model, image power at minus 100MHz contributes to the baseband signal through the low pass filters with less attenuation.

This example highlights one of the limitations of baseband equivalent models: at any point in the system, the signal should not have a bandwidth larger than any carrier (RF or IF) of the system. For this example, the baseband model is only valid for input RF signals between 1GHz and 1.2GHz.

The limitation is somewhat moot because the idea behind a baseband equivalent model is to suppress all carriers. To simulate the image response with the baseband model we had to include a 200MHz source! We would have been better off simply not suppressing the 100MHz IF carrier, i.e. using baseband models for the RF stages but passband models for the IF stages.

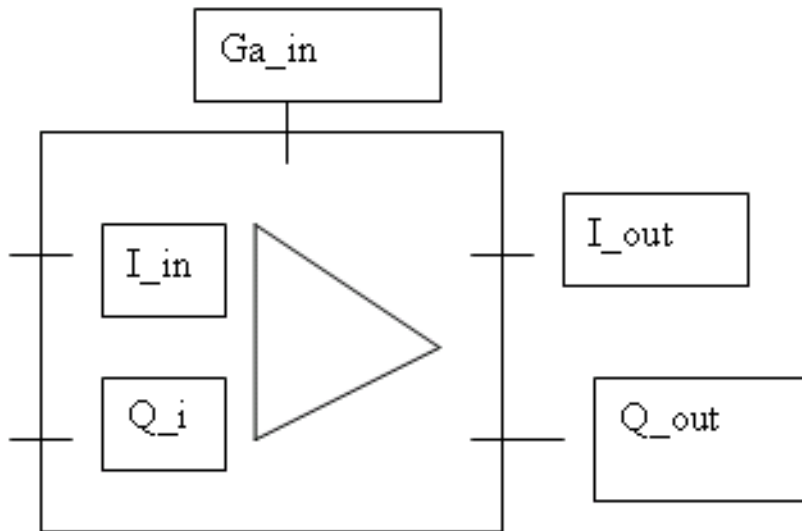
In summary, an all-baseband equivalent model of an image rejection receiver is only good for simulating the response to the desired RF signal, not the image response.

## Variable Gain Amplifier Model

(baseband = VGA\_BB)

Only the Baseband view is available.

Figure D-62 Variable Gain Amplifier Model



The following parameters specify the variable gain amplifier model.

Baseband model only.

- gpv = voltage gain per volt on the G\_in pin.
- cpdb = db compression point, measured in dbm, referred to the output.
- psinf = Output phase shift as the input power goes to infinity.
- pscp = Output phase shift at the 1db compression point.
- shp = Determines how fast the phase shift occurs with increasing input power. A larger number delays the shift but makes the shift rise faster as a function of input signal level.
- cw = Determines the direction of the phase shift. The phase shift is in one direction only.
- +1 = counter-clockwise
  - -1 = clockwise

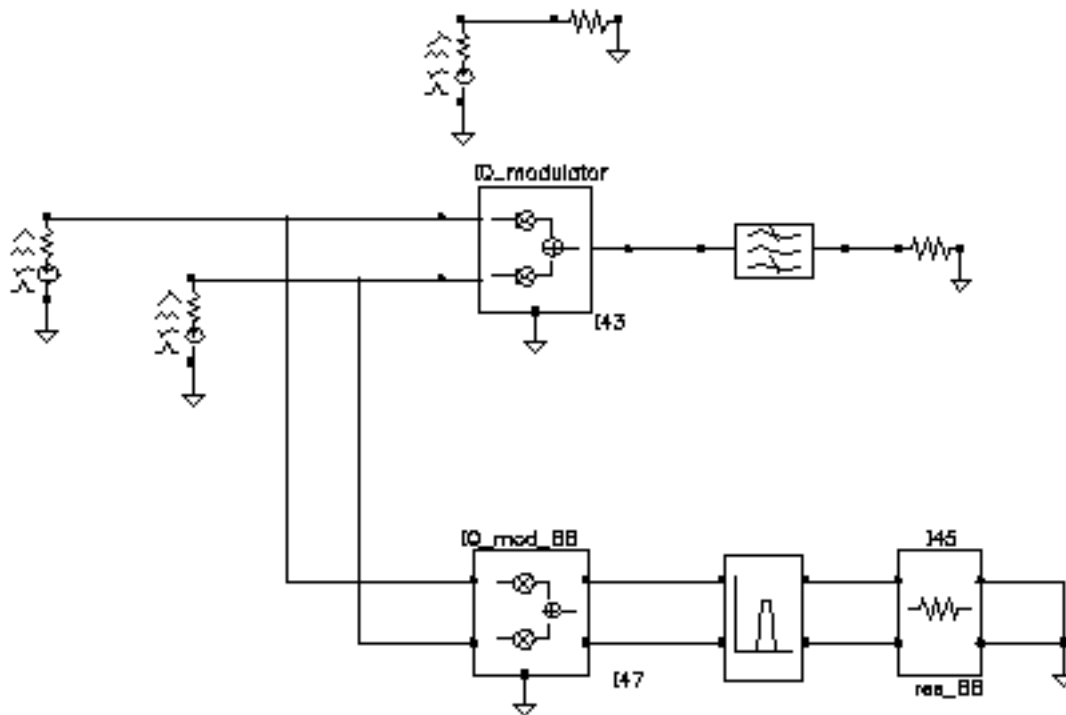


- ❑ 0 = no phase shift (no am/pm conversion)
- rin = input resistance
- rout = output resistance

## Comparison of Baseband and Passband Models

The circuit in [Figure D-63](#) on page 873 shows how well the baseband and passband filters agree. The I-input is a 5MHz 1 volt peak sinusoid and the Q-input signal is a 20MHz 1 volt peak signal. The filter has a center frequency of 1.1GHz and a relative bandwidth of 0.1. The modulator LO is 1GHz. To make the analysis more interesting the carrier is not exactly aligned with the filter center frequency and the terminals are not matched. The circuit is listed as *PB\_BB\_filter\_comparison* in the testbenches category of the *rfLib*.

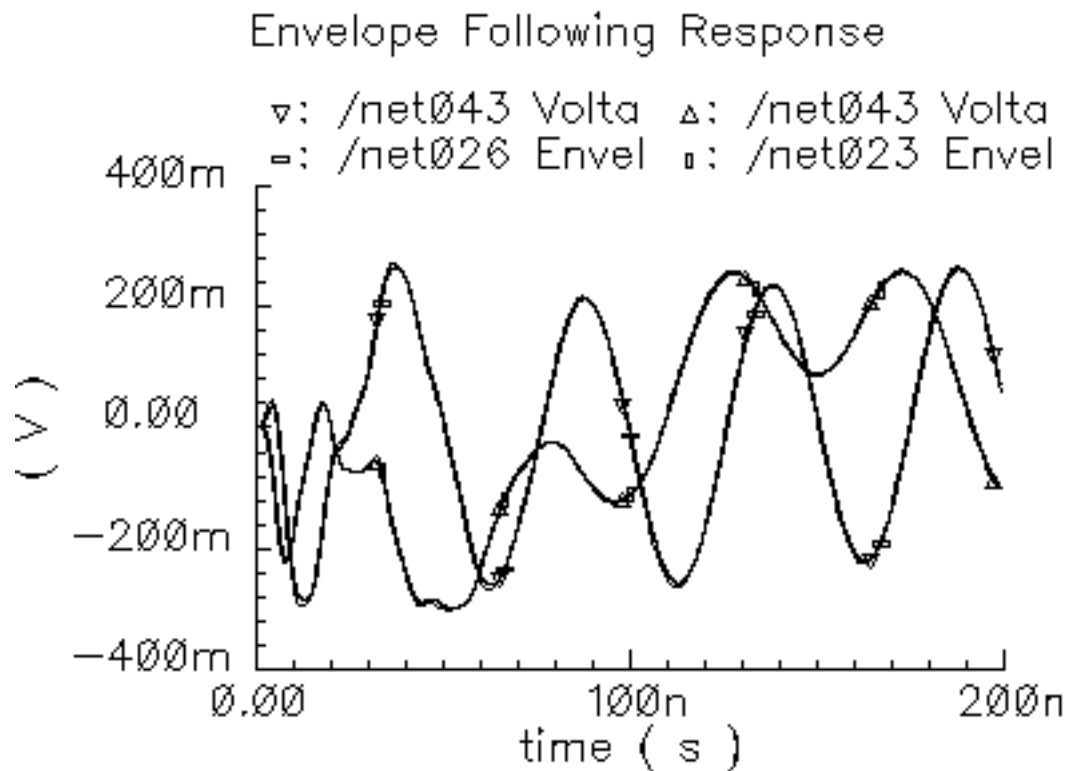
**Figure D-63 PB\_BB\_filter\_comparison Circuit**



1. Bring up the test circuit and an Analog Environment window.
2. Set up an Envelope analysis with “carrier” as the Clock Name. Set `reltol` in the analog options to  $1e-5$ . You can use the default `reltol` of  $1e-3$  but you do not get the waveforms close to the baseband results.

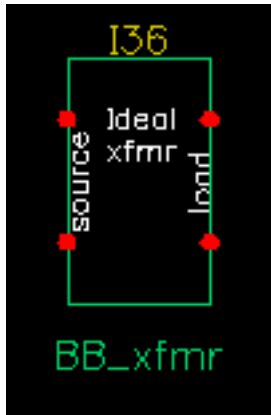
3. Plot the “time” waveforms of the BB\_butterworth\_bp outputs. These waveforms are the response of the baseband equivalent model.
4. Plot the “harmonic time”, 1 harmonic, real and imaginary waveforms at the butterworth\_lp output. These waveforms are the baseband waveforms extracted from a passband model. Figure D-64 on page 874 overlays the baseband and passband results. The baseband and passband filter models produce identical equivalent baseband waveforms. The slight offset in time is due to the ambiguity associated with deciding whether to plot a time-varying Fourier coefficient at the beginning or at the end of a clock cycle.

**Figure D-64 I and Q Baseband Equivalent Outputs**



## Measurement Blocks

### Ideal Transformer (BB\_xfmr)



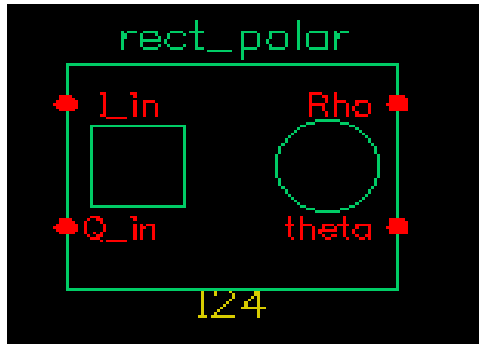
The ideal transformer is in the *measurement* category of the *rfLib*. Its purpose is to help designers transform between different resistances.

- Parameters: source resistance  $R_s$  and load resistance  $R_l$ .
- Inputs = i and q input voltages, i and q output currents.
- Outputs = i and q output voltages, i and q input currents defined as follows:

$$\begin{bmatrix} i_{iin} \\ i_{qin} \end{bmatrix} = \sqrt{\frac{R_l}{R_s}} \begin{bmatrix} i_{iout} \\ i_{qout} \end{bmatrix}$$

$$\begin{bmatrix} v_{iout} \\ v_{qout} \end{bmatrix} = \sqrt{\frac{R_l}{R_s}} \begin{bmatrix} v_{iin} \\ v_{qin} \end{bmatrix}$$

## Rectangular-to-Polar Transformation (rect\_polar)



The rectangular-to-polar block is in the measurement category. The only parameters are input and output resistances. The inputs are the baseband signal in Cartesian coordinates, the outputs are the baseband signal in polar coordinates.

Parameters: Input and output resistances.

- Inputs =  $i$  and  $q$  volts.
- Outputs =  $\rho$  and  $\theta$  volts, such that

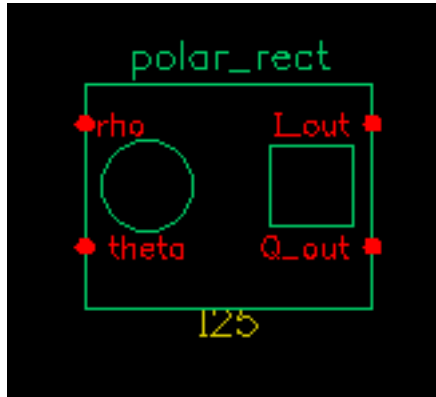
$$\rho = \sqrt{i*i + q*q}$$

and

$$\theta = \text{atan}[q/i] + \pi*(1 + \text{sgn}[i]) / 2$$

where  $\theta$  is in radians and with appropriate checks for the  $i = 0$  case

## Polar-to-Rectangular Transformation (polar\_rect)



The polar-to-rectangular block is in the measurement category. The only parameters are input and output resistances. The inputs are the baseband signal in polar coordinates, the outputs are the baseband signal in rectangular coordinates.

Parameters: Input and output resistances.

Inputs =  $\rho$  and  $\theta$  volts.

■ Outputs =  $i$  and  $q$  volts, such that

$$i = \rho * \cos(\theta)$$

$$q = \rho * \sin(\theta)$$

## Instrumentation and Terminating Blocks

- `comms_instr`
- `offset_comms_instr`
- `instr_term`

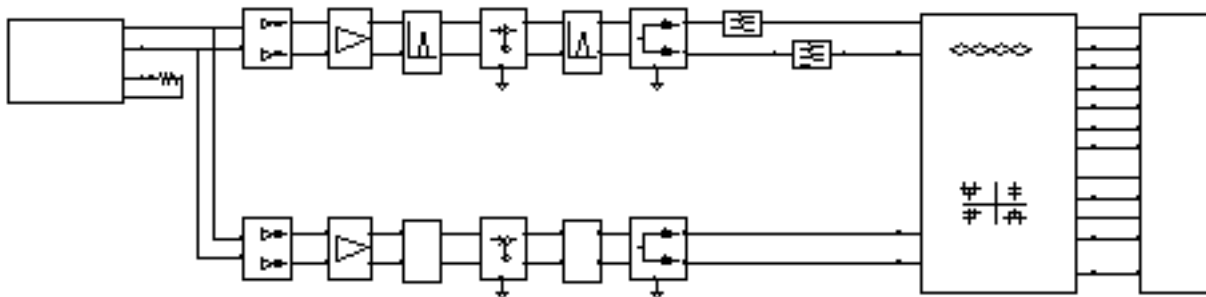
The instrumentation block is in the testbench category and generates waveforms that can be used to create eye-diagrams, eye-diagram statistics, scatter plots, and rms error-vector-magnitude. An example was given in “[Use Model and Design Example](#)” on page 576. This section summarizes the parameters and outputs.

There are two kinds of instrumentation blocks, one called `comms_instr` and one called `offset_comms_instr`. The offset block is identical to the first one except the sampling time for scatter plots and eye-diagram statistics are delayed by half a symbol period. The delay makes it possible to plot symbols in an offset QPSK modulation scheme.

There is one other block in the library associated with the two instrumentation blocks, the `instr_term` model. The `instr_term` block simply loads all instrumentation output pins with 50 Ohms. The `instr_block` simply keeps the schematic editor from complaining about unconnected pins, nothing more.

[Figure D-65](#) on page 878 shows how the `offset_comms_instr` and `instr_term` blocks should be used. The `comms_instr` block is used similarly. The circuit consists of two branches driven from a single baseband signal generator. The top branch is the non-ideal receiver model, the bottom branch is an ideal version of the top branch. The ideal version is ideal as you like. The ideal branch computes ideal symbol locations in the complex plane. The instrumentation block compares ideal and non-ideal symbols to compute error-vector-magnitude.

**Figure D-65 EVM setup**



**Parameters:**

**symbols per second.** This parameter is necessary for generating the sawtooth that is used as the x-axis to generate eye-diagrams. It also determines the rate at which the input waveforms are sampled.

**I-sampling delay.** This parameter sets the phase of the symbol sampler. It is referenced to the eye-diagram of the I-output. Estimate the optimal delay by doing one simulation just to get the eye-diagram. The optimal delay is the time from the leftmost part of the eye-diagram, which should be zero, to the time at which they eye is widest.

**number of symbols.** This is the number of symbols to sweep in the eye-diagram. Sweeping two symbols ensures that you see at least one continuous eye, if it is open.

**max, min eye-diagram volts, and number of hstgm bins.** These parameters are used to compute the bins which define the eye-diagram histogram. The bin width equals (max voltage-min voltage)/(number of bins). The histogram shows the distribution of the I\_in voltage at the sampling instant.

**I-noise and Q-noise(volts^2).** These parameters set the variance of Gaussian random variables which can be added to the received symbols before anything is computed or plotted.

**statistics start time.** This parameter delays the start of any statistical computations. The purpose is to exclude start-up transients from the statistics.

**input resistance.** This parameter is the input resistance of the input terminals of the instrumentation block.

**Eye-diagrams.** To generate an eye-diagram of the I-input signal, plot the sawtooth and I-eye outputs in one waveform display tool. Change the x-axis to be the sawtooth. This is done through the x-axis menu in the waveform display tool. The procedure for generating an eye-diagram of the Q-output is the same except you use the Q-eye output.

**Histograms.** You can only generate a histogram of the I-input signal. The histogram shows the distribution of the I-input voltage at the sampling instant. To generate a histogram, plot the eye\_hist and eye\_count\_hist outputs in the same waveform display window. Change the x-axis to be the eye-hist signal then change the plot to *bar*.

**Eye-diagram statistics.** The ave\_eye output is the average absolute value of the I-input signal at the sampling instant. The root\_var\_eye output is the square root of the variance of the absolute value of the I-input voltage at the sampling instant. The voltages at these output pins represent running estimates of the associated statistics.

**Scatter plots.** A scatter plot is the I-input and Q-input samples plotted against each other. The scatter plot shows the locations of the received symbols. To generate a scatter plot, plot I\_scatter and Q\_scatter in the same waveform display window then change the

x-axis to be the I\_scatter signal. Finally, change the plot to plot data points only. A scatter plot of the reference model can be generated similarly by replacing I\_scatter and Q\_scatter with Iref\_scatter and Qref\_scatter.

**rms Error Vector Magnitude.** The rms EVM is defined as the square root of the sum of the squares of the vectorial differences between the ideal and non-ideal received symbols, normalized to the rms value of the magnitude of the ideal received symbols. The output voltage at this pin is represents a running calculation of the rms EVM. The normalized EVM is in percent.



## Baseband Drive Signals (BB\_driver, CDMA\_reverse\_xmit and GSM\_xmtr)

You can import a baseband signal into a Spectre RF transient simulation using a `port` component by setting the `port` component's *Source type* to *pwl* and specifying the path to a file containing the baseband signal data. You can also use *ppwlf* sources which recognize SPW format.

Use one of the three baseband signal generators in the measurement category

- `BB_driver`
- `CDMA_reverse_xmit`
- `GSM_xmtr`

The baseband signal generators can drive J-models directly. When driving top-down models, they should be used with a `BB_driver` model.

### BB\_driver

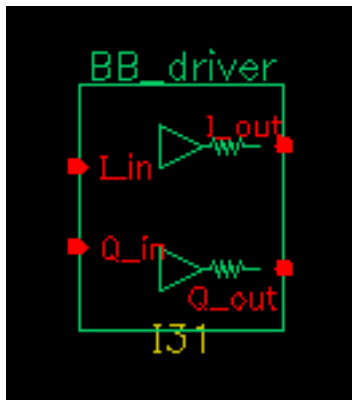
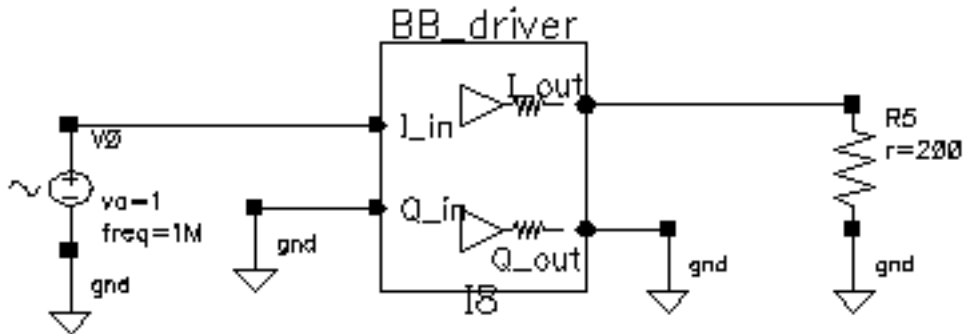


Figure D-66 BB\_driver



The BB\_driver converts peak volts into dBm. It is intended for use with the baseband signal generators. The BB\_driver is nothing more than an identical pair of voltage-controlled current sources with identical output resistances. The I\_out pin is independent of the Q\_in pin and the Q\_out pin is independent of the I\_in pin. The following example describes what this model does. [Figure D-66](#) on page 882 shows the set up.

1. Build the schematic. Set the Output resistance parameter to 200 Ohms and the “dBm-out@1v peak in” parameter to 10.
2. Load one output with 200 Ohms and drive the associated input with a sinusoidal voltage that has a peak voltage of 1 volt.
3. Simulate a few cycles with a Transient analysis. The power delivered to the 200 Ohm load is 10 dBm, which corresponds to a peak voltage of 2 Volts across the 200 Ohm load.

### CDMA\_reverse\_xmit

[Figure D-67](#) on page 883 shows the symbol for the CDMA\_reverse\_xmit generator with the BB\_driver. [Figure D-68](#) on page 883 shows the signal trajectory for the CDMA\_reverse\_xmit generator.

Figure D-67 CDMA\_reverse\_xmit generator

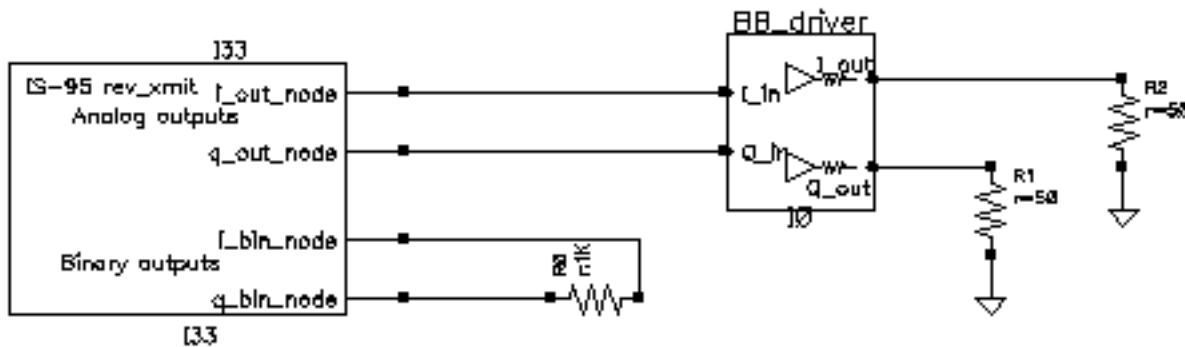
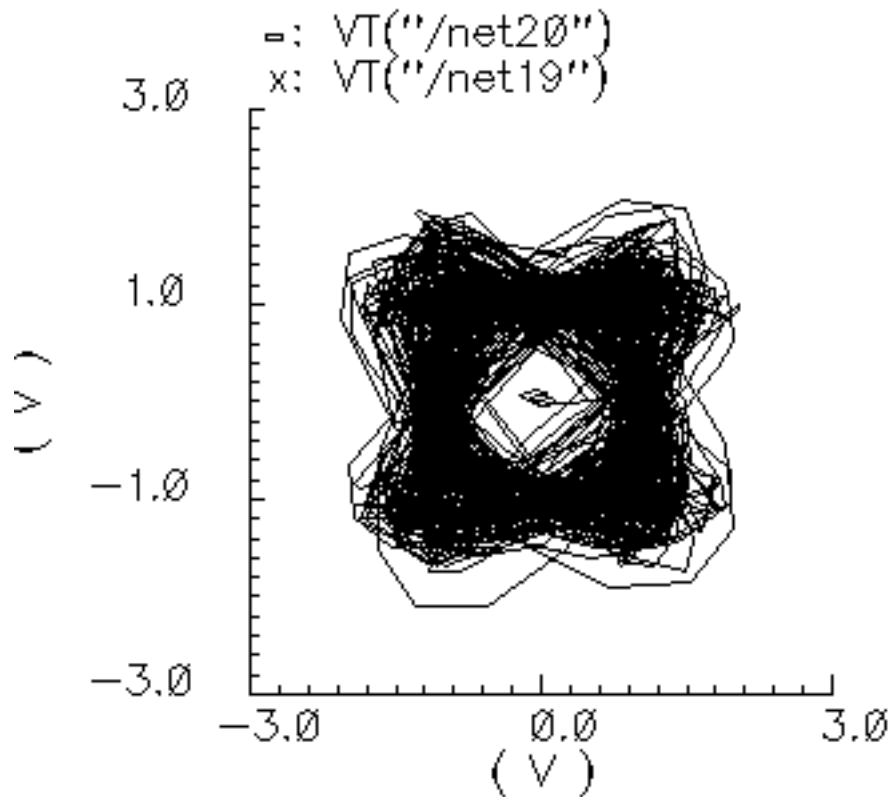


Figure D-68 Signal Trajectory for CDMA\_reverse\_xmit



This model generates an IS-95 reverse link transmitter signal. The modulation is offset QPSK. A single stream of random bits is spread with two different spreading sequences to generate

the I and Q outputs. The spreading sequences are generated with pseudo noise generators that have 15 states. The resulting chip rate is 1.2288 Megachips per second on each output. (A chip is a piece of a bit. It is the product of a bit in the spreading sequence and the original bit that is being spread.) Each chip is over sampled at a 4:1 rate. Each stream of samples drives its own 48-tap FIR filter. The Q-output is delayed by 2 samples, half a chip, to convert the QPSK signal to offset QPSK.

The `i_out_node` and `q_out_node` pins produce voltages equal to the digitally filtered I and Q outputs.

The `i_bin_node` and `q_bin_nodes` are the binary outputs. The binary outputs are the raw unfiltered chips.

The model has three parameters:

**seed:** This is the seed for the random number generator that generates the bits.

**amplitude:** This sets the peak amplitude of the `i_out_node` and `q_out_node` pins.

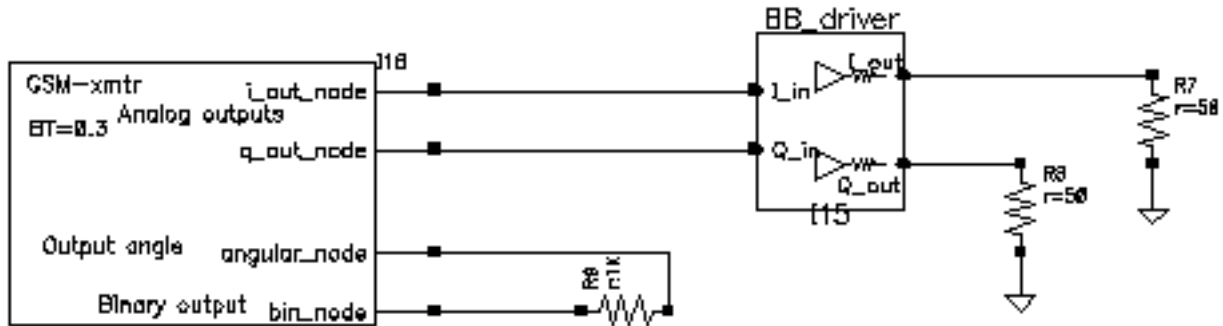
**t-rise\_fall:** The digitally filtered outputs jump from one value to the next in a time specified by this parameter. To generate outputs consisting of data points connected by straight lines, set `t-rise_fall` to the chip (or symbol) period. For more of a staircase waveform, make this parameter a small fraction of the symbol period.

The spreading sequences and FIR filters can be redefined through the Modelwriter.

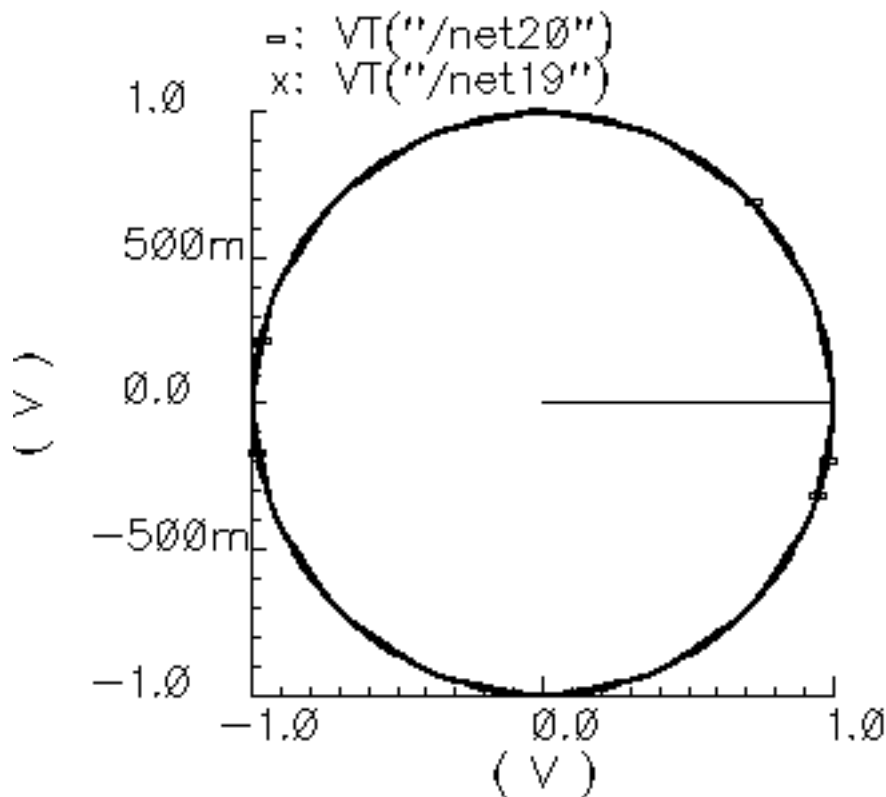
## **GSM\_xmtr**

Figure D-69 on page 885 shows the symbol for the `GSM_xmtr` generator with the `BB_driver`. Figure D-70 on page 885 shows the signal trajectory for the `GSM_xmtr` generator.

**Figure D-69 GSM\_xmtr generator**



**Figure D-70 Signal Trajectory for the GSM\_xmtr**



The GSM signal generator starts with a bit sequence of three zeros, 8.25 ones, and three zeros. GSM modulation is 0.3 GMSK [11]. This pattern models 3 stop bits, 8.25 guard bits, and 3 start bits. The fractional guard bits are important in eliminating two sharp dips in the

power spectral density. The “frame” concludes with 142 random bits. The bit rate is 270833.333 bits per second. Each bit is over sampled at a 4:1 rate then filtered with a Gaussian FIR filter that has 32 taps. The filtered bits modulate the frequency of a VCO. The gain of the VCO equals  $0.25 \cdot \text{bit\_rate} \cdot \text{oversample\_rate}$  Hz/volt.

The voltage on the `i_out_node` pin equals the amplitude, a user-specified parameter, scaled by the cosine of the VCO phase.

The voltage on the `q_out_node` pin equals the amplitude scaled by the sine of the VCO phase.

The `angular_node` pin produces a voltage numerically equal to the phase of the VCO in radians.

The `bin_node` is a voltage representing the bits that drive the FIR filter.

Aside from the amplitude parameter, all parameters are used as they are in the `CDMA_reverse_xmit` model.

The FIR filter can be modified through the Model Writer.

### **pi\_over4\_dqpsk**

[Figure D-71](#) on page 886 shows the symbol for the `pi_over4_dqpsk` with the `BB_driver`.

[Figure D-72](#) on page 887 shows the signal trajectory for the `pi_over4_dqpsk`.

**Figure D-71 pi\_over4\_dqpsk generator**

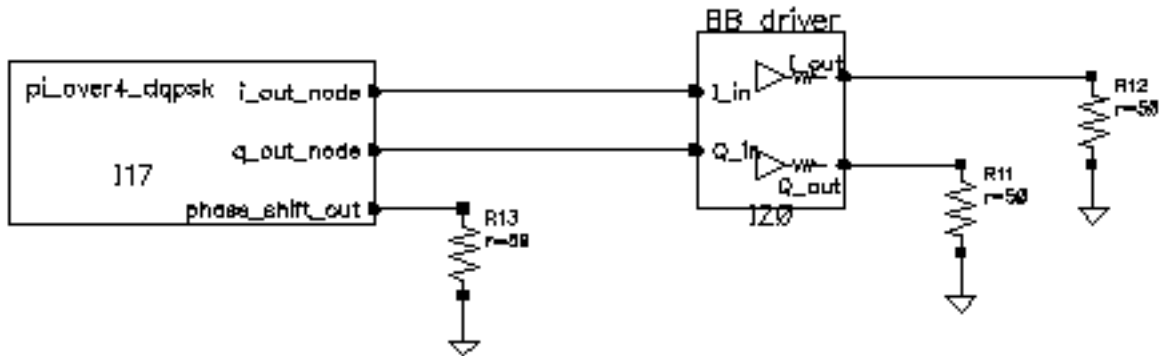
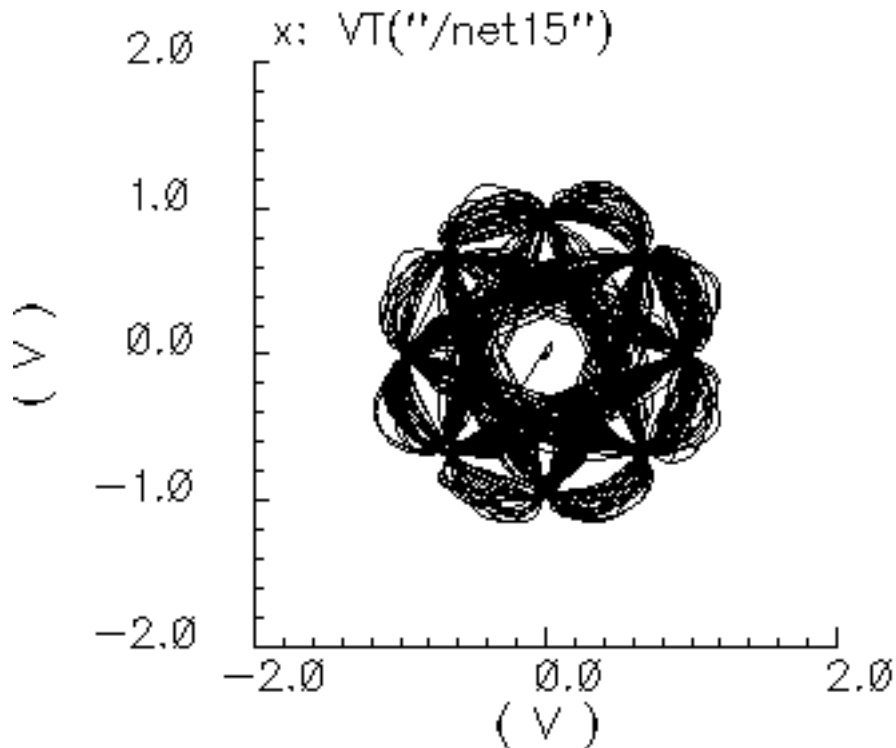


Figure D-72 Signal Trajectory for the pi\_over4\_dqpsk



In  $\pi/4$  DQPSK modulation, information rides on one of four possible phase shifts in the carrier. The possible phase shifts are  $\pm 45$  degrees and  $\pm 135$  degrees. The model selects one of the four phase shifts at random then computes I and Q pulses that drive FIR filters. The I-pulse equals the amplitude parameter, scaled by the cosine of the resulting absolute phase. Absolute phase equals the sum of all past phase shifts plus the present one. The initial phase is zero.

The Q-pulse equals the amplitude parameter, scaled by the sine of the absolute phase. The I and Q pulses drive their own 64-tap FIR filters. The I and Q filters are identical. The filters are raised cosine filters with a 0.35 roll off factor. The rate at which the phase shifts occur is 24300 shifts per second. The FIR filters operate at 8 times that rate.

The voltages on the `i_out_node` and `q_out_nodes` pins are the filter outputs.

The `phase_shift_out` pin is a voltage numerically equal to the phase shift.

Aside from the amplitude parameter, all parameters are used as they are in the `CDMA_reverse_xmit` model.

The FIR filter can be modified through the Model Writer.

## References

- 1 M. Jeruchim, P. Balaban, K. Shanmugan, "Simulation of Communication Systems", Plenum Press, 1992.
- 2 E. Lee, D. Messerschmitt, "Digital Communication", Kluwer Academic Publishers, 1994.
- 3 A. Fitzgerald, C. Kingsley Jr., S Umans, "Electric Machinery", Fifth edition, McGraw Hill.
- 4 R. Stein and W. Hunt Jr., "Electric Power System Components", Van Nostrand Reinhold Company, 1979.
- 5 W. Leonhard, "Control of Electrical Drives", Springer-Verlag 1990.
- 6 P. Kraus and O. Wasynchuk, "Electromechanical Motion Devices", McGraw-Hill Book Company, 1989.
- 7 P. Vas, "Vector Control of AC Machines", Oxford University Press, 1990.
- 8 G. Kron, "A Short Course in Tensor Analysis for Electrical Engineers", John Wiley and Sons, Inc, 1942.
- 9 J. Chen, "Rotor Reference Frame Models of a Multiloop 2-phase Motor Drive in Brushless DC and MicroStepping Modes", 1995 Intersociety Energy Conversion Engineering Conference, Orlando, FLA.
- 10 A. Harrysson, M. Ziren, "System Level Simulations of a Receiver for W-CDMA", Master of Science Thesis, Department of Applied Electronics, Lund Institute of Technology, 12/99.
- 11 T. Rappaport, "Wireless Communications, Principles and Practice". 1996. Prentice Hall.

## WCDMA Blocks

### Introduction

WCDMA (wideband code-division multiple-access) is one of the air interfaces for the third generation of wireless communications developed within the framework of International Mobile Telecommunications (IMT)-2000. WCDMA transmitters spread encoded user data at a relatively low rate but at a much wider band (5Mcps) and WCDMA receivers separate the desired information based on the unique code of the intended user.



In release IC 6.1.2 and later, Cadence provides Verilog-A modules for simulating WCDMA behavior. The modules are located in `rfLib`.

The included modules are:

- [“QPSK Modulation/Mapping”](#) on page 889
- [“DL Common Channel Generator”](#) on page 890
- [“Spreading”](#) on page 891
- [“Scrambling/Scrambling Code Generator”](#) on page 892
- [“OCNS Generator”](#) on page 892
- [“SCH Generator/Multiplexer”](#) on page 893
- [“Power Adjustment”](#) on page 894
- [“Square-Root Raised Cosine Filtering”](#) on page 894

The following sections describe these blocks.

## QPSK Modulation/Mapping

**Figure D-73** `wcdma_qpsk` symbol



This block has two inputs, `I_in` and `Q_in`, and two outputs, `outI` and `outQ`. The inputs receive random input from outside the module. The outputs produce QPSK signals in baseband.

- This module includes the interleaving and encoding of data.
- You can use either random input from the outside or internal random bits. If `enable_input` is on, external data is used, otherwise internal random bits are used.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

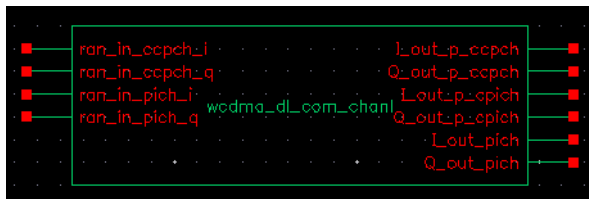
### The RF Library

The instance parameters for the `wcdma_qpsk` block are listed in the following table:

Name	Meaning	Type	Default Value	Range
<code>bits_per_integer</code>	Number of bits concerted into integer	integer	2	[1:31]
<code>frame_time</code>	The time of one frame	real	1.0/15000.0	(0:inf)
<code>mapping_mode</code>	Mapping type	string	user_defined	Binary_gray, gray_binary, user_defined
<code>usr_mapping_vec</code>	Bit mapping between input and output	integer	{0, 3, 1, 2}	
<code>phase_offset</code>	Initial phase	real	0	
<code>enable_input</code>	1 means to use outside input; otherwise 0	Boolean	1	0, 1
<code>seed</code>	Used for random	type	2	
<code>samples</code>	Number of samples in one frame	integer	12345	

## DL Common Channel Generator

Figure D-74 `wcdma_dl_com_chan` symbol



This module, with four inputs and six outputs, generates the pCPICH (primary common pilot), PICH (paging indicator channel) and pCCPCH (primary common control physical channel). For PICH and pCCPCH, either external or internal random signals can be selected according

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

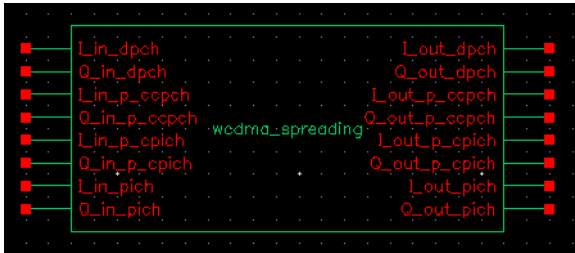
### The RF Library

to the value of the `enable_input` parameter. The parameters of the module are listed in the following table:

Name	Meaning	Type	Default Value	Range
<code>sample</code>	Sample time	real	1.0/15000	
<code>pich_seed</code>	Seed for PICH	integer	12345	
<code>ccpch_seed</code>	Seed for CCPCH	integer	98765	
<code>enable_input</code>	Enable input if 1; otherwise disable input.	integer	1	1, 0

## Spreading

Figure D-75 `wcdma_spreading` symbol



This module spreads the data over the OVSF codes. The parameters of the `wcdma_spreading` module are listing in the following table:

Name	Meaning	Type	Default Value	Range
<code>sf</code>	Spread factor	integer	128	[4:512]
<code>dpch_code</code>	OVSF index for DPCH	integer	10	
<code>pcpich_code</code>	OVSF index for PCPICH	integer	0	
<code>pich_code</code>	OVSF index for PICH	integer	4	
<code>pccpch_code</code>	OVSF index for PCCPCH	integer	1	

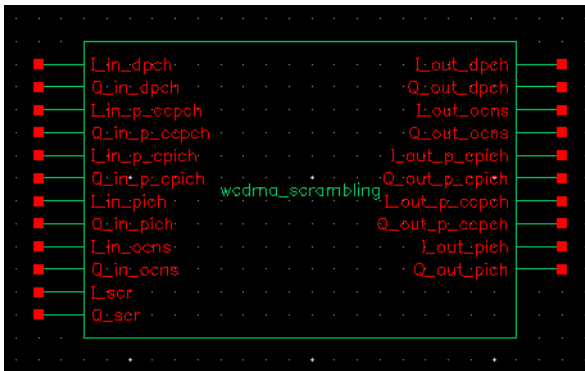
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

frame_time	Frame time	real	1.0/15000	
oversample	Oversample	integer	128	

## Scrambling/Scrambling Code Generator

**Figure D-76** wcdma\_scrambling symbol



This module scrambles the spread code. The parameters of the `wcdma_scrambling` module are listed in the following table:

Name	Meaning	Type	Default Value	Range
frame_time	Frame time	real	1.0/15000	(0:inf)
numChipsOut	Number of chips	integer	256	

## OCNS Generator

**Figure D-77** wcdma\_ocns symbol



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

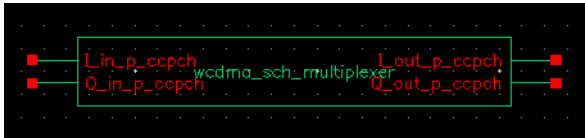
---

This module produces the combination of 16 dedicated data channels. The parameters of the module are listed in the following table:

Name	Meaning	Type	Default Value	Range
frame_time	Frame time	Real	1.0/15000	(0:inf)
numChipsOut	Number of chips	integer	256	
sf	Spread factor	integer	128	[4:512]
enable_input	Enable input if 1, otherwise disable input.	integer	1	1, 0

## SCH Generator/Multiplexer

Figure D-78 wcdma\_sch\_multiplexer symbol

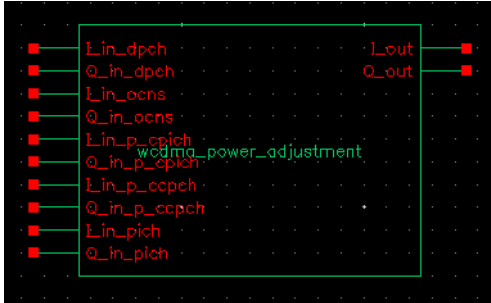


This module produces the synchronization channel and performs SCH multiplexing. The parameters of the module are listed in the following table:

Name	Meaning	Type	Default Value	Range
frame_time	Frame time	real	1.0/15000	(0:inf)
numChipsOut	Number of chips	integer	256	
ssc_num	Scrambling code group number	integer	64	[1:64]
hada_order	Hadamard matrix order	integer	8	

## Power Adjustment

**Figure D-79 wcdma\_power\_adjust symbol**

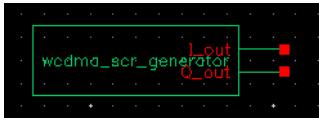


This module applies the weight to different channels. The parameters of the module are listed in the following table:

Name	Meaning	Type	Default Value	Range
frame_time	Frame time	real	1.0/15000	(0:inf)
numChipsOut	Number of chips	integer	256	
power_dpch	Power for DPCH	real	-5.5	
power_cpich	Power for CPICH	real	-10	
power_pich	Power for PICH	real	-15	
power_ccpch	Power for CCPCH	real	-12	

## Square-Root Raised Cosine Filtering

**Figure D-80 wcdma\_scr\_generator symbol**



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### The RF Library

This module up-samples and filters the input. The parameters of the module are listed in the following table:

Name	Meaning	Type	Default Value	Range
frame_time	Frame time	real	1.0/15000	(0:inf)
numChipsOut	Number of chips	integer	256	
group_delay		integer	6	
alpha		real	0.22	
over_samples		integer	8	

## GMSK Block

GMSK (Gaussian minimum shift keying) is a simple but efficient approach to digital modulation that provides the properties of narrow-band techniques, sharp cutoffs in frequency, lower overshoot impulse response, and preservation of the filter output pulse area. These qualities result in low phase distortion and make GMSK suitable for coherent demodulation. The GMSK approach is used in the Global System for Mobile Communication (GSM).

In release IC 6.1.2 and later, Cadence provides a Verilog-A modules for simulating GMSK behavior. The module is located in `rfLib`.

**Figure D-81 GMSK symbol**



The `inbit` input supports the use of external random generators. Usually `inbit` is disabled by specifying `enable_input = 0`.

The parameters of the instance are listed in the table below:

Name	Meaning	Type	Default Value	Range
BT	BT product	real	0.3	
initial_phase	Initial phase	real	0	

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
The RF Library

---

Ec	Power per bit in dB	real	-115	
frame_time	Frame time	real	6.0/1625000.0	
pulse_length	Pulse length of the filter	integer	4	
samples	Samples in one frame	integer	4	
initial_syms	Initial state of the filter.	integer	{1, 1, 1}	Length of array must be less than pulse length
enable_input	1 if using external random input, otherwise 0	integer	1	0 or 1
seed	Seed for internal random generator	integer	21	



---

# Plotting Spectre S-Parameter Simulation Data

---

This appendix describes the equations used by the Waveform Calculator to plot data generated by Spectre S-parameter simulations.

Using the Waveform Calculator in the analog design environment, you can plot the following S-parameter data:

- [Network Parameters](#)
- [Two-Port Scalar Quantities](#)
- [Two-Port Gain Quantities](#)
- [Two-Port Network Circles](#)
- [Equation for VSWR \(Voltage Standing Wave Ratio\)](#)
- [Equation for GD \(group delay\)](#)

Use the buttons located in the *Function* area of the S-parameter Direct Plot form to specify the type of data to plot.

## Network Parameters

You can plot S, Y, Z, and H network parameters.

S-parameters, Y-parameters, and Z-parameters (denoted as SP, YP, and ZP on the Direct Plot form) are defined for circuits with any number of ports. H-parameters (denoted as HP on the Direct Plot form) are defined only for two-port circuits.

You can plot parameters on polar charts, Smith charts, or on rectangular plots after applying a *Modifier* option. The dB conversion uses  $20 \log_{10} X$  because the parameters represent scalar ratios (for example, voltage).

## Equations for Network Parameters

For the ZP, YP, and HP parameters, Spectre returns S-parameters to the analog design environment. The environment converts them, as needed, to the equivalent Z, Y, and H matrixes using standard published methods. Spectre calculates S-parameter values.

SP (S-parameter) values

SP (S-parameter) values are calculated by Spectre.

ZP (Z-parameter) equation

The Z-parameter equation is as follows

$$Z_m = [Z_{ref}][I + S_m][I - S_m]^{-1}[Z_{ref}]$$

Where

- $S_m$  is the N-port S-parameter matrix
- $I$  is the N x N identity matrix
- $Z_{ref}$  is the characteristic impedance of the port
- $Z_m$  is the resulting Z-parameter matrix

## YP (Y-parameter) equations

The Y-parameter equations are as follows

$$Y_m = [Y_{ref}][I - S_m][I + S_m]^{-1}[Y_{ref}]$$

Where

- $S_m$  is the N-port S-parameter matrix
- $I$  is the N x N identity matrix
- $Y_{ref}$  is a diagonal matrix defined as

$$Y_{ref} = \frac{1}{\sqrt{\Re\{Z_i\}}}$$

Where

- $Z_i$  is the terminating impedance at port  $i$
- $Y_m$  is the resulting Y-parameter matrix

### The HP (H-parameter) equations

The HP (H-parameter) equations only apply to two-port circuits.

$D$  is

$$D = (1 - S_{11})(1 + S_{22}) + S_{21}S_{12}$$

$H_{11}$  is

$$H_{11} = \frac{[(1 + S_{11})(1 + S_{22}) - S_{21}S_{12}]Z_{ref1}^2}{D}$$

$H_{21}$  is

$$H_{21} = \left(\frac{-2S_{21}}{D}\right)\frac{Z_{ref1}}{Z_{ref2}}$$

$H_{12}$  is

$$H_{12} = \left(\frac{2S_{21}}{D}\right)\frac{Z_{ref1}}{Z_{ref2}}$$

and  $H_{22}$  is

$$H_{22} = \frac{(1 - S_{11})(1 - S_{22}) - S_{21}S_{12}Z_{ref}^2}{D}$$

Where

- $Z_{ref1}$  is the terminating impedance at port 1
- $Z_{ref2}$  is the terminating impedance at port 2

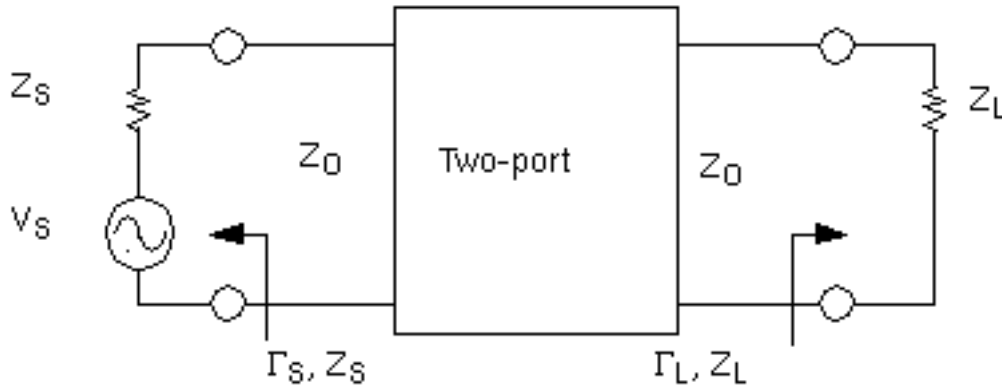
## Two-Port Scalar Quantities

The two-port scalar quantities include

- $NF_{min}$  is minimum noise figure
- NF is noise figure
- $R_n$  is equivalent noise resistance
- $G_{min}$  is Optimum Noise reflection coefficient
- $B_{1f}$  is alternative stability factor
- $K_f$  is stability factor

You can plot two-port scalar quantities only against frequency. In addition, you can plot them only on rectangular charts. Figure [E-1](#) illustrates a generic two-port circuit that defines impedances and reflection coefficients.

**Figure E-1 A Generic Two-Port Circuit**



In Figure [E-1](#)

- $Z_s$  is the source impedance
- $\Gamma_s$  is the input reflection coefficient
- $Z_L$  is the load impedance
- $\Gamma_L$  is the load reflection coefficient
- $Z_0$  is the characteristic impedance

## Equations for Two-Port Scalar Quantities

### $G_{min}$ (Optimum Noise Reflection Coefficient) Equation

$G_{min}$  (also known as  $\Gamma_{min}$  in the literature) is the reflection coefficient associated with minimum noise figure. You can plot  $G_{min}$  on a Smith chart or, by using the `Modifier` field, on a rectangular chart.

$$G_{min} = \Gamma_{min} = \frac{Z_{on} - Z_0}{Z_{on} + Z_0}$$

Where

- $Z_0$  is the characteristic impedance
- $Z_{on}$  is the source impedance associated with minimum noise figure ( $NF_{min}$ )

### **NF<sub>min</sub> (Minimum Noise Figure) and NF (Noise Figure) Equations**

The NF<sub>min</sub> and NF plots are controlled by the *Modifier* option in the Direct Plot form.

Plot noise figure (NF) in dB by setting *Modifier* to *dB10*

- Plot noise factor (F) by setting *Magnitude*

Use NF<sub>min</sub> and NF only for two-port circuits.

$$NF = 10\log_{10}F$$

Where

- F is the noise factor
- NF<sub>min</sub> is the minimum noise figure

The NF<sub>min</sub> values are calculated by Spectre

The NF (noise figure) equation is calculated by the analog design environment from NF<sub>min</sub>, G<sub>min</sub> (Γ<sub>min</sub>), and r<sub>n</sub>. You can specify the optional source reflection coefficient Γ<sub>S</sub> as an argument if you use the analog design environment Waveform Calculator. From the Direct Plot form, the analog design environment assumes Γ<sub>S</sub> to be 0 (input matched to reference termination).

$$NF = NF_{min} + \frac{4r_n|r_s - r_{min}|^2}{(1 - |r_s|^2)(1 + |r_{min}|^2)}$$

Where

$$r_n = \frac{R_n}{Z_o}$$

Here r<sub>n</sub> is the *normalized* equivalent noise resistance.

R<sub>n</sub> (equivalent noise resistance)

R<sub>n</sub> plots equivalent noise resistance. The R<sub>n</sub> values are calculated by Spectre.

## Stability Factors

$K_f$  and  $B_{1f}$  plot the Rollet stability factor and its intermediate term. Use these parameters only for two-port circuits.

$D$  is

$$D = S_{11}S_{22} - S_{21}S_{12}$$

$K_f$  is

$$K_f = \frac{1 - |S_{11}|^2 - |S_{22}|^2 + |D|^2}{2|S_{21}||S_{12}|}$$

and  $B_{1f}$  is

$$B_{1f} = 1 + |S_{11}|^2 - |S_{22}|^2 - |D|^2$$

## Two-Port Gain Quantities

The following gain quantities are valid only for two-port circuits

$G_A$  (available gain) is the power gain obtained by optimally matching the output of the network.

$G_P$  (power gain) is the power gain obtained by optimally matching the input of the network.

$G_T$  (transducer gain) shows the insertion effect of a two-port circuit. This quantity is used in amplifier design.

$G_{umx}$  (maximum unilateral transducer power gain)

$G_{max}$  (maximum available gain) shows the transducer power gain when there exists a simultaneous conjugate match at both ports.

$G_{msg}$  (maximum stable gain) shows the gain that can be achieved by resistively loading the two-port such that  $k = 1$  and then simultaneously conjugately matching the input and output

ports. For conditionally stable two-ports, you can approach the maximum stable gain as you reduce the input and output mismatch. If you attempt a simultaneous conjugate match and  $k < 1$ , the two-port oscillates.

## Equations for Two-Port Gain Calculations

### $G_A$ (Available Gain) Equations

Available gain equations for output conjugately matched.

$$G_A = \frac{|S_{21}|^2(1 - |\Gamma_S|^2)}{|1 - S_{11}\Gamma_S|^2(1 - |\Gamma_2|^2)}$$

Where

$$\Gamma_2 = S_{22} + \frac{S_{12}S_{21}\Gamma_S}{1 - S_{11}\Gamma_S}$$

**Note:** When you use the S-parameter Direct Plot form,  $G_S$  is set to zero, and therefore available gain ( $G_A$ ) is plotted as

$$G_A = \frac{|S_{21}|^2}{1 - |S_{22}|^2}$$

To plot  $G_A$  for nonzero values of  $G_S$ , use the analog design environment Waveform Calculator.

### $G_P$ (Power Gain) Equations

Power gain equations for input conjugately matched.



$$G_P = \frac{|S_{21}|^2(1 - |\Gamma_L|^2)}{|1 - S_{22}\Gamma_L|^2(1 - |\Gamma_1|^2)}$$

Where

$$\Gamma_1 = S_{11} + \frac{S_{12}S_{21}\Gamma_L}{1 - S_{22}\Gamma_L}$$

**Note:** When you use the S-parameter Direct Plot form,  $\Gamma_L$  is set to zero, and therefore the power gain,  $G_P$  is plotted as

$$G_P = \frac{|S_{21}|^2}{1 - |S_{11}|^2}$$

To plot  $G_P$  for nonzero values of  $\Gamma_L$ , use the analog design environment Waveform Calculator.

### **$G_T$ (Transducer Gain) Equations**

$$G_T = \frac{(1 - |\Gamma_S|^2)|S_{21}|^2(1 - |\Gamma_L|^2)}{|(1 - S_{11}\Gamma_S)(1 - S_{22}\Gamma_L) - S_{12}S_{21}\Gamma_S\Gamma_L|^2}$$

**Note:** When using the S-parameter Direct Plot form, the analog design environment assumes that the source ( $\Gamma_S$ ) and load ( $\Gamma_L$ ) reflection coefficients are zero.  $G_T$ , therefore, plots the insertion gain.

$$G_T = |S_{21}|^2$$

Using the Waveform Calculator, you can plot  $G_T$  and specify the source and load terminations.

### $G_{max}$ (Maximum Available Gain) Equations

For  $K > 1$

$$G_{max} = \left| \frac{S_{21}}{S_{12}} \right| \left[ K - \sqrt{K^2 - 1} \right]$$

For  $K \leq 1$

$$G_{max} = \left| \frac{S_{21}}{S_{12}} \right|$$

Where  $K$  is the stability factor,  $K_f$

### $G_{umx}$ (Maximum Unilateral Power Gain) Equation

$$G_{umx} = \frac{|S_{21}|^2}{(1 - |S_{11}|^2)(1 - |S_{22}|^2)}$$

### $G_{msg}$ (Maximum Stable Power Gain) Equation

$$G_{msg} = \left| \frac{S_{21}}{S_{12}} \right|$$

## Two-Port Network Circles

NC plots constant noise contours at the input of a two-port circuit. GAC plots constant gain contours at the input port, and GPC plots constant gain contours at the output port. Gain contour values reflect an optimum match at the opposing port.

Noise and Gain circles can be plotted at a single dB value for a range of frequencies or at a single frequency for a range of dB values. If you do not enter values for the frequency range,

a circle is plotted for every simulated frequency for which a circle with the specified value exists.

SSB plots stability circles at the input port, and LSB plots stability circles at the output port. You can also specify a limited frequency range for these contours.

## Equations for Two-Port Network Circle

### NC (Noise Circle) Equations

$$N_i = \frac{(F_i - F_{min})|1 + \Gamma_{min}|^2}{4r_n}$$

where

$$\Gamma_{min} = G_{min}$$

The center is calculated using

$$C_N = \frac{\Gamma_{min}}{1 + N_i}$$

The radius is calculated using

$$r_N = \sqrt{\frac{N_i^2 + N_i(1 - |\Gamma_{min}|^2)}{1 + N_i}}$$

Where i is the index number

### GAC (Available Gain Circle) Equations

The center is calculated using

$$C_A = \frac{g_a(S_{11}^* - D^*S_{22})}{1 + g_a(|S_{11}|^2 - |D|^2)}$$

The radius is calculated using

$$r_A = \frac{\sqrt{1 - 2K_f |S_{21}S_{12}| g_a + |S_{12}S_{21}|^2 g_a^2}}{|1 + g_a(|S_{11}|^2 - |D|^2)|^2}$$

Where

$$G_a = \frac{G_A}{|S_{21}|^2}$$

And

$$D = S_{11}S_{22} - S_{12}S_{21}$$

### GPC (Power Gain Circle) Equations

The center is calculated

$$C_P = \frac{g_p(S_{22}^* - D^*S_{11})}{1 + g_p(|S_{22}|^2 - |D|^2)}$$

The radius is calculated using

$$r_P = \frac{\sqrt{1 - 2K_f |S_{21} S_{12}| g_p + |S_{12} S_{21}|^2 g_p^2}}{1 + g_p (|S_{22}|^2 - |D|^2)}$$

Where

$$g_p = \frac{G_P}{|S_{21}|^2}$$

And

$$D = S_{11} S_{22} - S_{12} S_{21}$$

### LSB (Load Stability Circle) Equations

The center is calculated using

$$C_L = \frac{S_{11} D^* - S_{22}^*}{|D|^2 - |S_{22}|^2}$$

The radius is calculated using

$$r_L = \left| \frac{S_{12} S_{21}}{|D|^2 - |S_{22}|^2} \right|$$

Where

$$D = S_{11}S_{22} - S_{12}S_{21}$$

### SSB (Source Stability Circle) Equations

The center is calculated using

$$C_S = \frac{S_{22}D^* - S_{11}^*}{|D|^2 - |S_{11}|^2}$$

The radius is calculated using

$$R_S = \left| \frac{S_{12}S_{21}}{|D|^2 - |S_{11}|^2} \right|$$

Where

$$D = S_{11}S_{22} - S_{12}S_{21}$$

### Equation for VSWR (Voltage Standing Wave Ratio)

VSWR is calculated from the S-parameters. You can plot the VSWR at any port in the circuit on a rectangular chart.

$$VSWR_i = \frac{1 + |S_{ii}|}{1 - |S_{ii}|}$$

Where i is the port number.

### Equation for ZM (Input Impedance)

You can plot input impedance if all other ports are matched

$$Z_m = \frac{1 + S_{ii}}{1 - S_{ii}} R$$

Where R the reference impedance of the port of interest and i is the port number

### **Equation for GD (group delay)**

GD (group delay) approximates the derivative of the phase with respect to frequency, normalized to 360 degrees. Units for group delay are in seconds.

$$G_d \cong \frac{-d\phi}{d\omega}$$

Group delay is calculated from the phase of the corresponding S-parameter (for example, GD<sub>21</sub> corresponds to S<sub>21</sub>).

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Plotting Spectre S-Parameter Simulation Data

---



---

## Using QPSS Analysis Effectively

---

Quasi-Periodic Steady-State (QPSS) analysis computes the quasi-periodic steady-state responses of circuits with multiple periodic inputs of differing frequencies. The QPSS analysis is a prerequisite for all quasi-periodic small-signal analyses such as the quasi-periodic AC (QPAC), quasi-periodic transfer function (QPXF), quasi-periodic S-Parameter (QPSP), and quasi-periodic noise (QPnoise) analyses provided by Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF).

A quasi-periodic signal has multiple fundamental frequencies. Closely spaced or incommensurate fundamental frequencies cannot be efficiently resolved by PSS. (Incommensurate frequencies are those for which there is no period that is an integer multiple of the period of each frequency.) QPSS allows you to compute responses to several moderately large input signals in addition to a strongly nonlinear tone which represents the LO or clock signal. The circuit is assumed to respond in a strongly nonlinear fashion to the large tone and in a weakly nonlinear fashion to the moderate tones. You might use a QPSS analysis, for example, to model intermodulation distortion with two moderate input signals. QPSS treats one of the input signals (usually the one that causes the most nonlinearity or the largest response) as the large signal, and the others as moderate signals.

The QPSS analysis employs the Mixed Frequency Time (MFT) algorithm extended to multiple fundamental frequencies. When input signals are stiff, that is, when they have large time constants, the QPSS analysis with its MFT algorithm is more efficient than transient circuit simulation. MFT also performs better than traditional algorithms such as harmonic balance when solving highly nonlinear problems.

A typical application of QPSS analysis is to predict the harmonic distortion of switched-capacitor filters that operate under widely separated fundamentals. See an example in [“Switched Capacitor Filter Example”](#) on page 923.

QPSS can also predict intermodulation distortion of a narrowband circuit that is driven by a local oscillator (LO) and two high-frequency input fundamentals that are closely spaced in frequency, such as in the down conversion stage of a receiver. See an example in [“High-Performance Receiver Example”](#) on page 925.

This appendix contains the following:

- A discussion of when to use the QPSS analysis
- A brief description of the MFT method
- A comparison of the QPSS and PSS analyses
- A comparison of the QPSS and PAC analyses
- Two examples of typical use: a switched capacitor filter and a high-performance receiver
- Procedures for setting up and running QPSS analysis

## When Should You Use QPSS Analysis

The increasing demand for low-cost, mobile communication systems increases the need for efficient and accurate simulation algorithms for RF communication circuits. These circuits are difficult to simulate because they process modulated carrier signals that consist of a high-frequency carrier and a low-frequency modulation signal. Typically, the carrier frequency ranges from 1-5 GHz while the modulation frequency ranges from 10 kHz to 1 MHz. If you were to apply a standard transient analysis to such a circuit, it might require simulating the detailed response of the circuit over hundreds of thousands of clock cycles (or millions of timepoints), a generally impractical approach.

Fortunately, many RF circuits of interest operate near a time-varying, but periodic, operating point. You can analyze some of these circuits if you assume that one of the circuit inputs produces a periodic response that you can calculate using a PSS analysis. You can treat other time-varying circuit inputs as small signals and linearize the circuit around the periodic operating point. You can then analyze small signals efficiently using the Spectre RF Periodic AC (PAC) analysis [8]. This approach lets you avoid long simulation times with transient analysis [7]. See [“QPSS and PSS/PAC Analyses Compared”](#) on page 922 for more information.

However, many RF circuits cannot be analyzed efficiently with the periodic-operating-point plus small-signal approach. For example, predicting the intermodulation distortion of a narrowband circuit, such as a receiver down converter (a mixer followed by a filter), requires calculating the nonlinear response of the mixer circuit, driven by a LO, to two closely spaced high-frequency inputs. The response to both inputs is within the band width of the filter. The steady-state response of such a circuit is quasi-periodic.

As a further complication, many multi-timescale circuits, such as mixers and switched-capacitor filters, have a highly nonlinear response with respect to one or more of the exciting inputs. Consequently, steady-state approaches such as multi frequency harmonic balance do not perform well.

The mixed frequency-time (MFT) approach used for QPSS analysis avoids these difficulties. MFT methods assume that many circuits of engineering interest have a strongly nonlinear response to only one input, such as the clock in a switched-capacitor circuit or the LO in a mixer, and respond in a weakly nonlinear manner to other inputs.

Compared to previous MFT methods [2, 4], the MFT algorithm used for QPSS analysis has the following advantages.

The MFT algorithm used for QPSS analysis

- Avoids the ill-conditioning caused by poorly chosen boundary conditions found in previous algorithms
- Uses a multi-dimensional discrete Fourier transform (DFT) scheme for cycle placement
- Uses a continuation method to enhance the global convergence of Newton's method
- Uses a matrix-implicit, Krylov-subspace-based iterative scheme that enables MFT methods to solve large problems
- Uses a preconditioning strategy that permits the iterative solver to converge rapidly

If you are unfamiliar with terminology such as *matrix-implicit*, *Krylov-subspace*, and *preconditioning*, you can find detailed descriptions in reference [6]. You can find an introduction to *continuation methods* in reference [1].

## Essentials of the MFT Method

Circuit behavior is usually described by a set of nonlinear differential-algebraic equations (DAEs) that can be written as,

$$(F-1) \quad \frac{d}{dt}Q(v(t)) + I(v(t)) + u(t) = 0$$

Where

- $Q(v(t)) \in \mathfrak{R}^N$  is typically the vector of sums of capacitor charges at each node
- $I(v(t)) \in \mathfrak{R}^N$  is the vector of sums of resistive currents at each node
- $u(t) \in \mathfrak{R}^N$  is the vector of inputs
- $v(t) \in \mathfrak{R}^N$  is the vector of node voltages

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

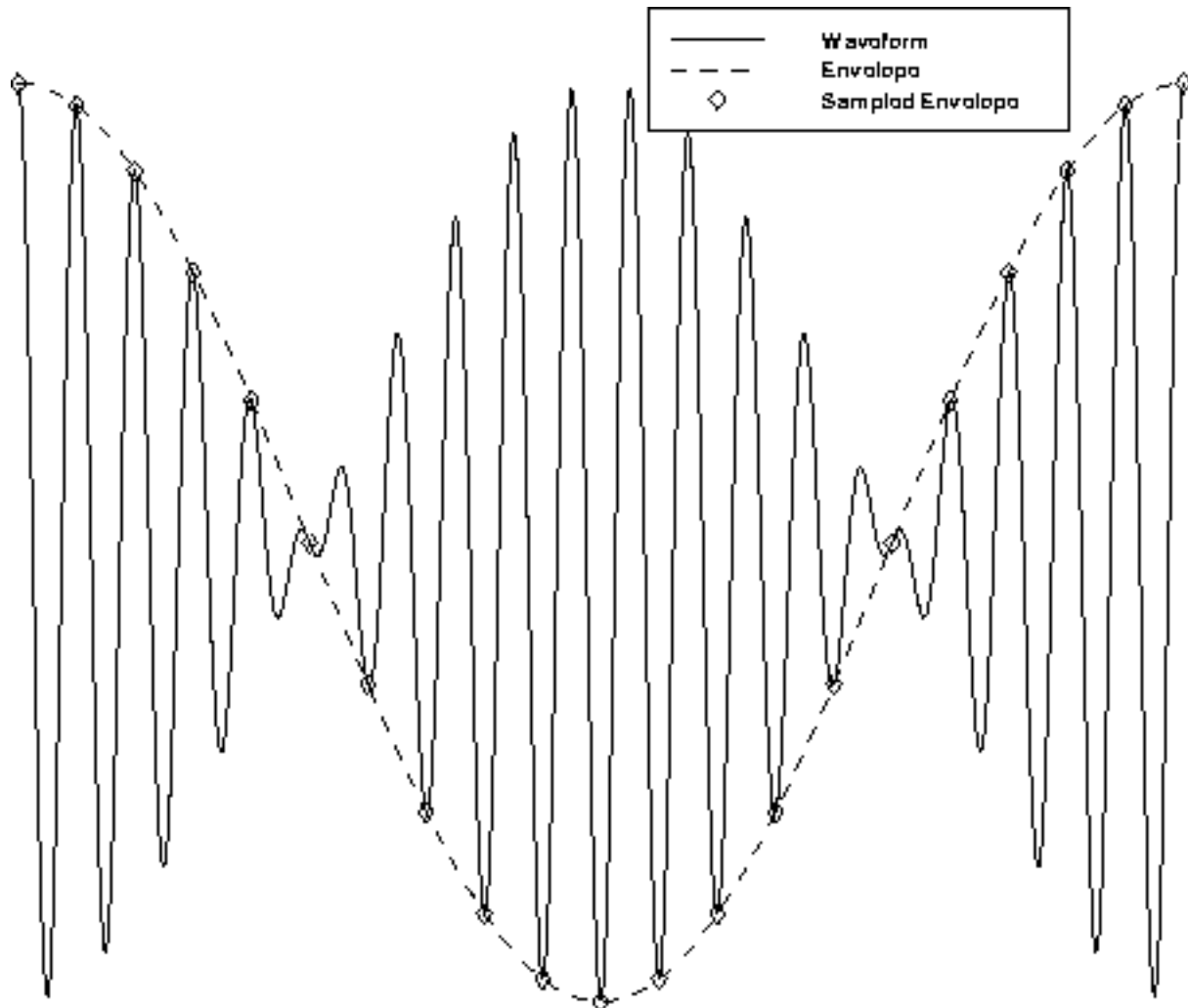
- $N$  is the number of circuit nodes

The MFT algorithm assumes that the circuit is in quasi-periodic steady-state; that is, that the signals can be represented as,

$$(F-2) \quad v(t) = \sum_k \sum_l V_{kl} e^{j2\pi(lf_0 + kf_1)t}$$

where, for simplicity, there are only two fundamental frequencies,  $f_0$  and  $f_1$ . The signal  $v(t)$  is then sampled at one of the fundamental frequencies,  $f_0$ , which is called the *clock* signal. This is shown in [Figure F-1](#) on page 917, where sampling a two-fundamental quasi-periodic signal at one of the fundamental frequencies creates a sampled waveform that is one-fundamental quasi-periodic, or simply-periodic. MFT directly finds the solution that, when sampled at  $f_0$ , is periodic in  $f_1$ .

Figure F-1 Sample Envelope Shown Sampled at the Waveform Peaks



The sample envelope shown in Figure F-1 is the waveform traced out when the signal is sampled with the clock period. The envelope is shown sampled at the peaks but this is not necessary.

The sampled waveform is,

$$(F-3) \quad \bar{v}_n = v(nT_0) = \sum_{k=-\infty}^{\infty} \bar{V}_k e^{j2\pi k f_1 t}$$

Where

$$T_0 = 1/f_0$$

Alternatively, you can also write,

$$(F-4) \quad \bar{v}_n = F^{-1}\bar{V}$$

which states that

$$\bar{v}$$

is the inverse Fourier transform of

$$\bar{V}$$

Recall that  $v$  is a solution of the circuit equations and that

$$\bar{v}$$

is simply  $v$  uniformly sampled, so given

$$\bar{v}_n$$

you can compute a subsequent sample point

$$\bar{v}_{n+1}$$

using Equation [F-5](#),

$$(F-5) \quad \bar{v}_{n+1} = \phi(\bar{v}_n, nT_0, (n+1)T_0)$$

In Equation [F-5](#),

- $\phi$  is the state transition function for the circuit

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

- $\phi(v_0, t_0, t_1)$  is the solution for the circuit equations at  $t_1$ , given that it starts from the initial condition  $v_0$  at  $t_0$ .
- Consider the  $n^{\text{th}}$  sample interval. Then let

$$x_n = \bar{v}_n$$

be the solution at the start of the interval and let

$$y_n = \bar{v}_{n+1} = x_{n+1}$$

be the solution at the end of the interval. Equation [F-5](#) uses the circuit equations to relate the solution at both ends of the interval as shown in Equation [F-6](#).

$$(F-6) \quad y_n = \phi(\bar{v}_n, nT_0, (n+1)T_0)$$

Let  $X = F_x$  and  $Y = F_y$  ( $X$  and  $Y$  are the Fourier transforms of  $x$  and  $y$ ). Then, from Equation [F-3](#) and because  $y_n = x_{n+1}$ ,

$$(F-7) \quad X_k = e^{-j2\pi k f_1 T_0} Y_k$$

Or

$$(F-8) \quad X = D_{T_0} Y$$

Where

$$D_{T_0}$$

is a diagonal delay matrix with

$$e^{-j2\pi k f_1 T_0}$$

being the  $k^{\text{th}}$  diagonal element.

Together, Equations [F-6](#) and [F-8](#) make up the MFT method.

In practice,

$$\bar{V} = X$$

is band-limited, so only a finite number of harmonics are needed. In addition, if the circuit is driven with one large high-frequency signal at  $f_0$ , which is called the *clock* signal, and one moderately-sized sinusoid at  $f_1$ , only  $K$  harmonics are needed and the method is efficient. With only  $K$  harmonics, [Equation F-6](#) on page 919 is evaluated over  $2K + 1$  distinct intervals that are spread evenly over one period of the lowest beat tone. Therefore, the total simulation time is proportional to the number of harmonics needed to represent the sampled waveform. The simulation time is independent of the period of the lowest-frequency beat tone or the harmonics needed to represent the clock signal.

[Equation F-8](#) on page 919, along with the associated Fourier transforms, relate the starting and ending points of the solution of the circuit equations over each interval, and consequently represent a boundary-value constraint on [Equation F-6](#) on page 919.

Shooting methods are the most common method for solving boundary-value problems. Their use of transient analysis to solve the circuit equations over an interval brings two important benefits.

Transient analysis handles abruptly discontinuous signals efficiently because the timestep shrinks to follow rapid transitions.

Transient analysis easily handles the strongly nonlinear behavior of the circuit as it responds to the large clock signal.

## QPSS and PSS Analyses Compared

Like PSS analysis, QPSS analysis uses a shooting Newton method. However, instead of doing a single transient integration, each Newton iteration does transient integrations over a number of nonadjacent clock periods. Each of the integrations differs by a phase-shift in each moderate input signal. The number of integrations is determined by the number of harmonics of moderate fundamentals that you specify. Given `maxharms = [k1, k2, ..., kS]`, the total number of integrations is



$$\prod_{s=1}^S (2k_s + 1)$$

Consequently, the efficiency of the algorithm depends significantly on the number of harmonics required to model the responses of moderate fundamentals.

Fortunately, the number of harmonics of the clock does not significantly affect the efficiency of the shooting algorithm. The boundary conditions of a shooting interval are such that the time-domain integrations are consistent with a frequency-domain transformation with a shift of one large-signal period.

As a Spectre RF user, you might need to run a PSS analysis to calculate the steady-state operating point of circuits with multiple input frequencies. You can do this by making the PSS beat frequency the greatest common factor of all the input frequencies.

When the greatest common factor is relatively close to the clock frequency; for example, within a factor of 10, PSS analysis might be preferable to QPSS analysis for two reasons.

Each nonlinear iteration would not require excessive integrations of clock period.

1. PSS analysis solves a much smaller linear system in each nonlinear iteration.
2. The size of the linear system resulting from QPSS analysis is

$$K = \prod_{s=1}^S (2k_s + 1)$$

times as big as that resulting from PSS analysis.

In general,

- If the ratio of *clock frequency/beat frequency* for a PSS analysis is smaller than  $K$ , a PSS analysis might be preferable to a QPSS analysis.
- For circuits such as switched-capacitor filters that operate on wide timescales, where the ratio of *clock frequency/beat frequency* for PSS analysis can be greater than 1000 — QPSS is clearly the analysis of choice.

Additional differences between the PSS and QPSS analyses are that PSS analysis does not have any restrictions on input sources, whereas QPSS analysis requires that all nonclock inputs must be sinusoidal. Also the QPSS analysis cannot be applied to autonomous circuits.

Like the PSS analysis and the periodic small-signal analyses, you must run the QPSS analysis to determine the quasi-periodic operating point before you can run the quasi-periodic small-signal analyses.

## QPSS and PSS/PAC Analyses Compared

Like PAC analysis, the QPSS analysis calculates responses of a circuit that exhibits frequency translations. However, instead of having small-signal linear behavior, QPSS models the response as having components of a few harmonics of input-signal frequencies. This permits computing responses to moderately large input signals.

PAC analysis assumes that only the clock is generating harmonics. For example, for a clock frequency  $f_c$ , and a small-signal frequency  $f_s$ , amplitudes of circuit response are generated at  $f_s + k_c f_c$  where  $k_c$  is bonded by the parameter `harms` in PSS analysis. In contrast, QPSS also permits nonclock fundamentals to generate harmonics. In the same situation, a spectrum at frequencies  $k_s f_s + k_c f_c$  is generated, where  $k_s$  and  $k_c$  are bonded by the QPSS parameter `maxharms`.

## QPSS Analysis Parameters

While QPSS analysis inherits most PSS parameters directly, the QPSS analysis adds two new parameters and extends the meaning of a few parameters. The two new parameters are the most important QPSS parameters.

- `funds`
- `maxharms`

They replace the PSS parameters, `fund` (or `period`) and `harms`, respectively.

The `funds` parameter accepts a list of names of fundamentals that are present in the sources. You specify these names in the sources using the `fundname` parameter. The simulator figures out the frequencies associated with the fundamental names.

An important feature of the `funds` parameter is that each input signal can be composed of more than one source. However, these sources must all have the same fundamental name. For each fundamental name, its fundamental frequency is the greatest common factor of all frequencies associated with the name.

The first fundamental is considered as the large signal. You can use a few heuristics to pick the large fundamental.

Pick the fundamental which is not sinusoidal.

Pick the fundamental which causes the most nonlinearity.

Pick the fundamental which causes the largest response.

The `maxharms` parameter accepts a list of numbers of harmonics needed for each fundamental.

If you do not list all the fundamental names using the `funds` parameter, the current analysis is skipped. However, if you do not specify `maxharms`, a warning message is issued, and the number of harmonics defaults to 1 for each fundamentals.

QPSS analysis expands the role of two PSS parameters.

- `maxperiods`
- `tstab`

The `maxperiods` parameter that controls the maximum number of shooting iterations for PSS analysis also controls the maximum number of shooting iterations for QPSS analysis.

The `tstab` parameter controls both the length of the initial transient integration, while only the clock tone is active, and the number of stabilizing iterations, while both the clock tone and the moderate tones are active. The stabilizing iterations run before the Newton iterations begin.

The remaining QPSS analysis parameters are inherited directly from PSS analysis, and their meanings remain essentially unchanged.

The `errpreset` parameter quickly adjusts several simulation parameters. In most cases, `errpreset` should be the only parameter you need to adjust. See [The `errpreset` Parameter in QPSS Analysis](#) in *Virtuoso Spectre Circuit Simulator RF Analysis Theory* for information about the `errpreset` parameter.

This is demonstrated by the following two examples

- A switched capacitor filter
- A high-performance receiver

## Switched Capacitor Filter Example

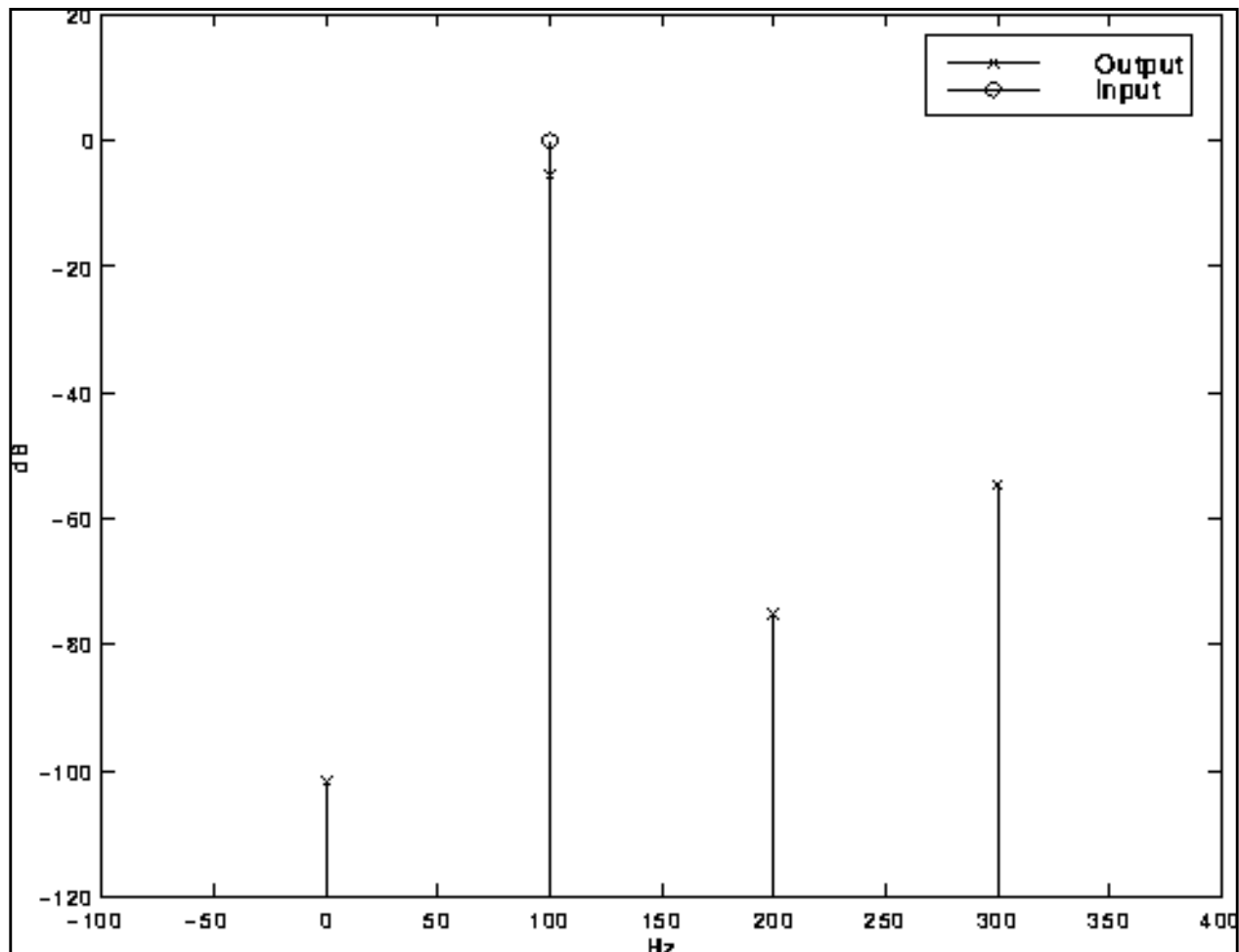
The low-pass switched-capacitor filter example has a 4 kHz bandwidth and 238 nodes, resulting in 337 equations. To analyze this circuit, the QPSS analysis was performed with an 8-phase 100 kHz clock and a 1 V sinusoidal input at 100 Hz.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

The 1000-to-1 clock-to-signal ratio makes this circuit difficult for traditional circuit simulators to analyze. Three harmonics were used to model the input signal. The eight-phase clock required about 1250 timepoints for each transient integration. The total number of variables solved by the analysis is  $337 \times (2 \times 3 + 1) \times 1250 = 2,948,750$ , slightly less than three million. The simulation completes in less than 20 minutes on a Sun UltraSparc1 workstation with 128 Megabyte memory and a 167 MHz CPU clock. A swap file is used because the analysis cannot be finished in core. For more information, see ["Memory Management"](#) on page 931. [Figure F-2](#) on page 924 shows the harmonic distortion.

**Figure F-2 Harmonic Distortion of a Switched Capacitor Filter**



## High-Performance Receiver Example

The high-performance image rejection receiver example consists of a low-noise amplifier, a splitting network, two double-balanced mixers, and two broad-band Hilbert transform output filters combined with a summing network that suppresses the unwanted sideband. A limiter in the LO path controls the amplitude of the LO. It is a rather large RF circuit that contains 167 bipolar transistors and uses 378 nodes. This circuit generated 987 equations in the simulator.

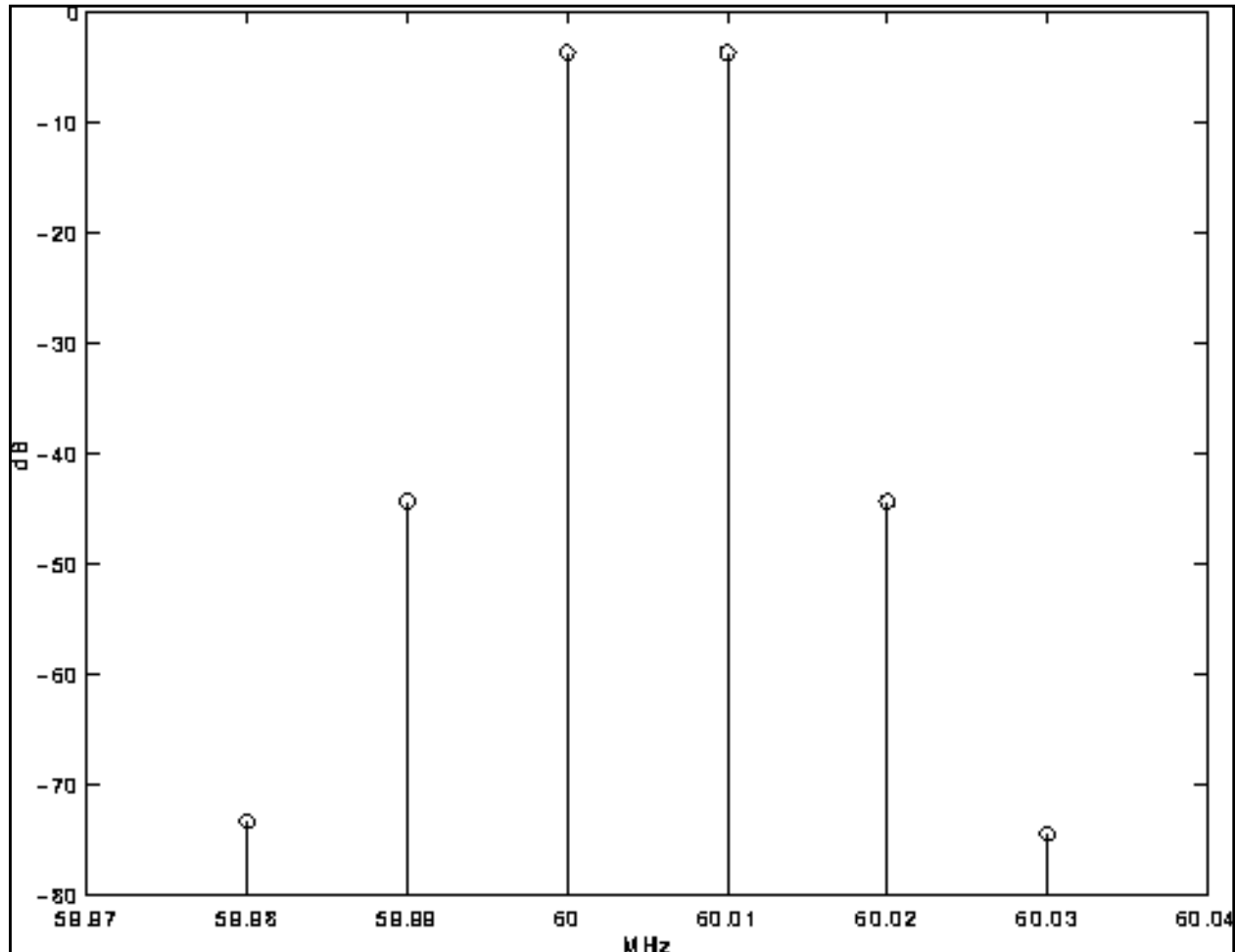
To determine the intermodulation distortion characteristics, the circuit is driven by a 780 MHz LO and two 50 mV closely placed RF inputs, at 840 MHz and 840 MHz+10 KHz, respectively. Three harmonics are used to model each of the RF signals, and 200 time points are used in each transient clock-cycle integration, considered to be a conservative accuracy specification for this circuit. As a consequence, about 10 million unknowns are generated

$$987 \times (2 \times 3 + 1)^2 \times 200 = 9,672,600$$

The simulation requires 55 CPU minutes on a Sun UltraSparc10 workstation with 128 megabytes of physical memory and a 300 MHz CPU clock. A swap file is used because the analysis cannot be finished in core. For more information, see [“Memory Management”](#) on page 931. [Figure F-3](#) on page 926 shows the 3<sup>rd</sup> and 5<sup>th</sup> order distortion products.

To appreciate the efficiency of the MFT method, consider that traditional transient analysis needs at least 80,000 cycles of the LO to compute the distortion, a simulation time of over two days. Additionally, the results might be inaccurate because of the large numerical error accumulated by integrating over so many cycles. In contrast, the MFT method is able to resolve very small signal levels, such as the 5<sup>th</sup> order distortion products shown in [Figure F-3](#).

Figure F-3 Intermodulation Distortion of a High-Performance Receiver



## Running a QPSS Analysis

The following sections describe how to set up and run a QPSS analysis. They also present ways to promote convergence.

### Picking the Large Fundamental

Your first task is to select the large fundamental, called the clock or the LO. Below are a few guidelines for selecting the large fundamental.

- Choose the one that is not sinusoidal.

- Choose the one that causes the most nonlinearity.
- Choose the one that causes the largest response.
- Choose the one that has the highest frequency.

## Setting Up Sources

You can specify the clock input using any type of source. However, other fundamentals can only be sinusoidal sources.

In addition to specifying the waveform parameters such as `type`, and `amp1`, you must also use the parameter `fundname` to specify a name for each non-DC source. Each fundamental can be a composition of several input sources with the same name. This is a difference between QPSS and other analyses.

Example F-1 shows how to set up the sources for the “[Switched Capacitor Filter Example](#)” on page 923.

An eight-phase clock:

### Example F-1 Setting Up the Sources for the Switched Capacitor Filter

```
//*****BEGIN NETLIST*****
.
.
.
// Clocks
Phil (phil gnd) vsource type=pulse period=2.5us delay=0.25us\ width=1us val0=-
VDD vall=VDD rise=10ns fundname="Clock"
Phi2 (phi2 gnd) vsource type=pulse period=2.5us delay=1.5us\ width=1us val0=-
VDD vall=VDD rise=10ns fundname="Clock"
Phi8 (phi8 gnd) vsource type=pulse period=5.0us delay=1.5us\ width=2.25us val0=-
VDD vall=VDD rise=10ns fundname="Clock"
Phi9 (phi9 gnd) vsource type=pulse period=5.0us delay=1.25us\ width=2.75us
val0=VDD vall=-VDD rise=10ns fundname="Clock"
Phi10 (phi10 gnd) vsource type=pulse period=10.0us delay=3.75us\ width=5.25us
val0=VDD vall=-VDD rise=10ns fundname="Clock"
Phi11 (phi11 gnd) vsource type=pulse period=10.0us delay=4.0us\ width=4.75us
val0=-VDD vall=VDD rise=10ns fundname="Clock"

// Input source
Vin (pin gnd) vsource type=sine freq=100_Hz amp1=1 sinephase=0\
fundname="Input"
// QPSS Analysis
harmDisto QPSS funds=["Clock" "Input"] maxharms=[3 3] +swapfile="SomeFileName"
//*****END NETLIST*****
```

Example F-2 shows how to set up the sources and analysis for the “[High-Performance Receiver Example](#)” on page 925.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

#### Example F-2 Setting Up the Sources for the High Performance Receiver

```
//*****BEGIN NETLIST*****
:
:
// LO input
Vlo (8 7) vsource type=sine ampl=400mV freq=780MegHz fundname="LO"
//RF inputs
Prf1 (15 44) port type=sine ampl=50mV freq=840MegHz fundname="F1"
+ ampl2=50mV freq2=840.01MegHz fundname2="F2"
//Analysis
InterModDisto QPSS funds=["LO" "F1" "F2"] maxharms=[5 3 3]
+ swapfile="SomeFileName"
//*****END NETLIST*****
```

#### Sweeping a QPSS Analysis

You can combine a QPSS analysis with a Spectre circuit simulator Sweep analysis to create a powerful tool for a wide variety of applications, such as IP3 and IP5 calculations. For example, you might want to calculate the distortion for input power ranging from -60 dBm to 0 dBm. The netlist for this task is shown in Example [F-3](#).

#### Example F-3 Calculating the Input Power Distortion from -60 to 0 dBm

```
//*****BEGIN NETLIST*****
:
:
parameters inputpower=-60
// Clock
Vlo (clock 0) vsource type=pulse val0=-1 val1=1 period=1n delay=0\ rise=10p
fall=10p width=460.00p fundname="carrier"
// Input port
Prf (input 0) port type=sine r=50.0 num=1.0
+freq=+9.0E+08 dbm=inputpower fundname="RF1"
+freq2=+9.05E+08 dbm2=inputpower fundname2="RF2"
// Analysis
swp sweep param=inputpower start=-60 stop=0 lin=6 {
distortion QPSS funds=["LO" "RF" "RF2"] maxharms=[5 3 3]
}
//*****END NETLIST*****
```

Always arrange the sweep values so that analyses that converge more easily are performed first. When you sweep QPSS, it automatically uses the converged steady-state solution of the previous QPSS analysis. As discussed in [“Convergence Aids”](#) on page 929, this practice can also aid convergence.



## Convergence Aids

Normally QPSS analysis converges with default parameter settings, but occasionally you might need to adjust some parameter settings in order to achieve convergence.

Normally, giving a sufficiently large `tstab` parameter value or a looser `steadyratio` value resolves convergence problems during the initial QPSS stages. For convergence problems during the QPSS iterations, try the following procedures.

- Increase the `tstab` parameter value.
- The `tstab` parameter controls both the length of the initial transient integration, with only the clock tone activated, and the number of stabilizing iterations, with the moderate tones activated. The stable iterations are run before Newton iterations begin.
- Increase the `steadyratio` parameter value.
- The `steadyratio` parameter guards against false convergence. Its default values, 1.0 for `liberal`, 0.1 for `moderate`, and 0.01 for `conservative`, are derived from the `errpreset` parameter. Tighten `steadyratio` only if you suspect false convergence.
- Sometimes `steadyratio` must be loosened (for example to `steadyratio = 1`) particularly with a tight `reltol` setting. Also loosen `steadyratio` when convergence stagnates. An indication of stagnation is that the `convNorm` value, which you can see on the screen, fluctuates within a certain range and never decreases further.
- The convergence tolerance of QPSS is determined by the product of `steadyratio` and `reltol`. Normally, you do not set `steadyratio` to a value higher than 10.
- Severe trapezoidal rule ringing can prevent convergence.
- If you suspect trapezoidal rule ringing, use `method = gear2` or `method = gear2only`.
- Avoid using unnecessarily tight `reltol` settings.
- Excessively tight `reltol` significantly reduces efficiency besides causing convergence problems.
- Do a continuation on a parameter, such as input power, of moderate fundamentals.
- When the circuit is behaving in a highly nonlinear manner at a certain input power level, the PSS plus PAC approach might not compute a good enough estimate of the initial condition. One effective strategy is to ramp up the input power gradually by carefully arranging a sweep as described in [“Sweeping a QPSS Analysis”](#) on page 928. Because it is usually much easier to achieve convergence at a low power input level where the

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

circuit behaves in a more linear manner, start with a low input power. Once the simulation converges, save the steady-state solution as the initial condition for the next input power level. This process repeats automatically as QPSS is swept. You can achieve convergence at the desired input power level if the sweep steps are sufficiently small.

- In general, avoid using an excessively high number of harmonics for the moderate fundamentals.
- Using too many harmonics lengthens the simulation time and uses a lot of memory. However, if the number of harmonics you use is not high enough for a particular fundamental to adequately model its nonlinear effects, convergence problems might also occur.

An important indication of convergence or divergence is the `CONV` value printed to the screen. There are a few typical scenarios shown in [Figure F-4](#) on page 931.

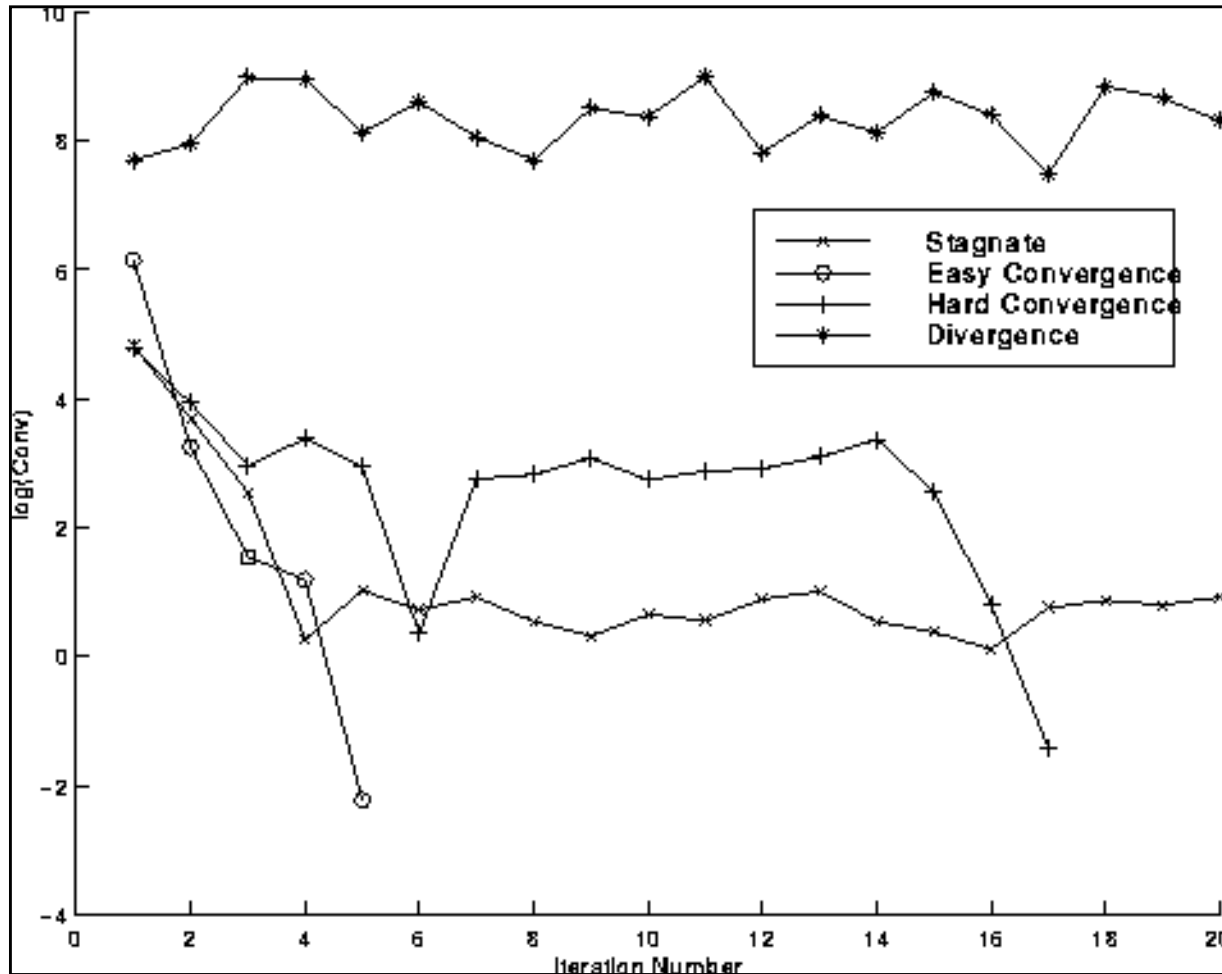
For most QPSS runs, the *Easy Convergence* scenario occurs.

A few simulations follow the *Hard Convergence* scenario.

If QPSS iterations *Stagnate* (the `CONV` value fluctuates close to but above 1.0), loosen `steadyratio`. Loosening `steadyratio` solves the stagnation problem.

If the iterations show *Divergence*, you usually must improve the initial condition. Do a continuation on input power level as described within the preceding list.

Figure F-4 QPSS Convergence Scenarios



## Memory Management

QPSS is a memory-intensive analysis. If QPSS cannot be finished in core with real physical memory, use a swap file residing on a local disk. The simulator manages swapping much more efficiently than the operating system. The examples found in “[Setting Up Sources](#)” on page 927 depict how to set up the `swapfile` parameter. Typically, 80% CPU utilization is achieved.

### Dealing with Sub-harmonics

One advantage of QPSS analysis over PSS is that you need only provide a name for each fundamental frequency. The actual beat frequency value associated with the name is calculated automatically by the simulator. For each unique fundamental name, the simulator

first finds all the frequencies associated with it. Then the greatest common factor is calculated among these frequencies, which is used as the beat frequency associated with the fundamental name.

However, if the fundamental frequency has sub-harmonics (circuit responses at some fraction of a driven frequency, typically 1/2 or 1/3), as with a divider, for example, the simulator currently cannot detect them. As a workaround, add a dummy source that tells the simulator of the existence of a sub-harmonic associated with a fundamental frequency.

## Understanding the Narration from the QPSS Analysis

The examples in this section describe information that is typically printed to the screen during a QPSS analysis run.

After initialization, the Spectre RF simulator confirms the fundamental tone names that were read and their beat frequencies. For this example, this circuit has

- A 1 GHz fundamental tone named `flo` as the large or clock signal.
- Two fundamental tones named `frf` and `fund2`, as moderate input signals. The frequency for `frf` is 900 MHz and the frequency for `fund2` is 920 MHz.

```
■
.
.
Fundamental flo: period = 1 ns, freq = 1 GHz.
Fundamental fund2: period = 1.08696 ns, freq = 920 MHz.
Fundamental frf: period = 1.11111 ns, freq = 900 MHz.
*****
Quasi-Periodic Steady State Analysis 'qpss': largefund = 1 GHz
*****
=====
'qpss': time = (1 ns -> 2 ns)
=====
.
.
.
```

QPSS prints the following message and begins the initial transient iteration.

Starting qpss analysis iterations.

The initial transient iteration runs with only the clock tone (the large signal) active and all moderate input signals suppressed. Unlike PSS analysis (where each iteration performs a single transient integration), each QPSS iteration performs a number of transient integrations.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

In this example, 25 transient integrations are performed. The `tstab` parameter controls the length of the initial transient integration.

For the 1<sup>st</sup> transient integration, QPSS prints data as it steps through the integration. For the 2<sup>nd</sup> through 25<sup>th</sup> integrations, QPSS prints a countdown timer for each integration.

At the end of the 1<sup>st</sup> transient iteration, QPSS prints out a summary that includes `tstab`, the iteration number, its convergence norm, the node with the maximum deviation, and the amount of CPU time spent in the iteration.

```

.
.
.
1st_Transient_Integration:
  qpss: time = 2.026 ns   (2.59 %), step = 1.792 ps   (179 m%)
  qpss: time = 2.077 ns   (7.66 %), step = 1.64 ps    (164 m%)
  qpss: time = 2.127 ns  (12.7 %), step = 2.572 ps   (257 m%)
  qpss: time = 2.176 ns  (17.6 %), step = 1.714 ps   (171 m%)
  qpss: time = 2.227 ns  (22.7 %), step = 2.492 ps   (249 m%)
  qpss: time = 2.277 ns  (27.7 %), step = 2.869 ps   (287 m%)
  qpss: time = 2.325 ns  (32.5 %), step = 3.514 ps   (351 m%)
  qpss: time = 2.375 ns  (37.5 %), step = 3.535 ps   (353 m%)
  qpss: time = 2.428 ns  (42.8 %), step = 3.566 ps   (357 m%)
  qpss: time = 2.477 ns  (47.7 %), step = 4.151 ps   (415 m%)
  qpss: time = 2.526 ns  (52.6 %), step = 4.464 ps   (446 m%)
  qpss: time = 2.576 ns  (57.6 %), step = 2.36 ps    (236 m%)
  qpss: time = 2.626 ns  (62.6 %), step = 2.378 ps   (238 m%)
  qpss: time = 2.675 ns  (67.5 %), step = 3.07 ps    (307 m%)
  qpss: time = 2.726 ns  (72.6 %), step = 4.936 ps   (494 m%)
  qpss: time = 2.777 ns  (77.7 %), step = 2.705 ps   (271 m%)
  qpss: time = 2.826 ns  (82.6 %), step = 3.1 ps     (310 m%)
  qpss: time = 2.875 ns  (87.5 %), step = 3.366 ps   (337 m%)
  qpss: time = 2.928 ns  (92.8 %), step = 3.661 ps   (366 m%)
  qpss: time = 2.976 ns  (97.6 %), step = 4.372 ps   (437 m%)

2nd_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
.
.
.
25th_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
'tstab' iter = 1, convNorm = 64.3, maximum dI(rif:p) = 51.5933 uA, took 5.49 s.
.
.
.

```

One or more stabilizing transient iterations run with all signals active; the clock tone (the large signal) and all moderate input signals. As for the first transient iteration, each QPSS iteration performs a number of transient integrations. This example performs 25 transient integrations. The `tstab` parameter controls the number of stabilizing iterations run when all tones are active.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

At the end of each stabilizing transient iteration, QPSS prints out a summary that includes `tstab`, the iteration number, its convergence norm, the node with the maximum deviation, and the amount of CPU time spent in the iteration.

```

.
.
.
1st_Transient_Integration:
qpss: time = 2.026 ns      (2.59 %), step = 1.792 ps      (179 m%)
qpss: time = 2.077 ns      (7.66 %), step = 1.64 ps       (164 m%)
qpss: time = 2.127 ns     (12.7 %), step = 2.572 ps     (257 m%)
qpss: time = 2.176 ns     (17.6 %), step = 1.714 ps     (171 m%)
qpss: time = 2.227 ns     (22.7 %), step = 2.492 ps     (249 m%)
qpss: time = 2.277 ns     (27.7 %), step = 2.869 ps     (287 m%)
qpss: time = 2.325 ns     (32.5 %), step = 3.514 ps     (351 m%)
qpss: time = 2.375 ns     (37.5 %), step = 3.535 ps     (353 m%)
qpss: time = 2.428 ns     (42.8 %), step = 3.566 ps     (357 m%)
qpss: time = 2.477 ns     (47.7 %), step = 4.151 ps     (415 m%)
qpss: time = 2.526 ns     (52.6 %), step = 4.464 ps     (446 m%)
qpss: time = 2.576 ns     (57.6 %), step = 2.36 ps      (236 m%)
qpss: time = 2.626 ns     (62.6 %), step = 2.378 ps     (238 m%)
qpss: time = 2.675 ns     (67.5 %), step = 3.07 ps      (307 m%)
qpss: time = 2.726 ns     (72.6 %), step = 4.936 ps     (494 m%)
qpss: time = 2.777 ns     (77.7 %), step = 2.705 ps     (271 m%)
qpss: time = 2.826 ns     (82.6 %), step = 3.1 ps       (310 m%)
qpss: time = 2.875 ns     (87.5 %), step = 3.366 ps     (337 m%)
qpss: time = 2.928 ns     (92.8 %), step = 3.661 ps     (366 m%)
qpss: time = 2.976 ns     (97.6 %), step = 4.372 ps     (437 m%)

2nd_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
.
.
.

25th_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
'tstab' iter = 2, convNorm = 15.2, maximum dI(rif:p) = -43.8125 uA, took 5.41 s.
.
.
.

```

The Newton iterations run after the stabilizing iterations.

The QPSS analysis employs the Mixed Frequency Time (MFT) algorithm extended to multiple fundamental frequencies. The large tone is resolved in the time domain and the moderate tones are resolved in the frequency domain (hence the name mixed frequency time algorithm). The QPSS analysis uses the shooting Newton method as its backbone. However, unlike PSS analysis where each Newton iteration performs a single transient integration, for each Newton iteration the QPSS analysis performs a number of transient integrations.

When you set up a QPSS analysis, you determine the number of integrations performed by the number of moderate fundamental harmonics you select. The efficiency of the shooting

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

Newton algorithm depends significantly on the number of harmonics required to model the responses of moderate fundamentals. The number of harmonics of the large fundamental does not significantly affect the efficiency of the Newton algorithm. The boundary conditions of a shooting Newton interval are such that the time domain integrations are consistent with a frequency domain transformation with a shift of one large signal period.

The shooting Newton iterations run with all signals active. As for the stabilizing transient iterations, each shooting Newton iteration performs a number of transient integrations. This example performs 25 transient integrations.

At the end of each shooting Newton iteration, QPSS prints out a summary that includes `Newton iter`, the iteration number, its convergence norm, the node with the maximum deviation, and the amount of CPU time spent in the iteration.

```

:
:
1st_Transient_Integration:
qpss: time = 2.026 ns      (2.59 %), step = 1.792 ps      (179 m%)
qpss: time = 2.077 ns      (7.66 %), step = 1.64 ps       (164 m%)
qpss: time = 2.127 ns     (12.7 %), step = 2.572 ps     (257 m%)
qpss: time = 2.176 ns     (17.6 %), step = 1.714 ps     (171 m%)
qpss: time = 2.227 ns     (22.7 %), step = 2.492 ps     (249 m%)
qpss: time = 2.277 ns     (27.7 %), step = 2.869 ps     (287 m%)
qpss: time = 2.325 ns     (32.5 %), step = 3.514 ps     (351 m%)
qpss: time = 2.375 ns     (37.5 %), step = 3.535 ps     (353 m%)
qpss: time = 2.428 ns     (42.8 %), step = 3.566 ps     (357 m%)
qpss: time = 2.477 ns     (47.7 %), step = 4.151 ps     (415 m%)
qpss: time = 2.526 ns     (52.6 %), step = 4.464 ps     (446 m%)
qpss: time = 2.576 ns     (57.6 %), step = 2.36 ps      (236 m%)
qpss: time = 2.626 ns     (62.6 %), step = 2.378 ps     (238 m%)
qpss: time = 2.675 ns     (67.5 %), step = 3.07 ps      (307 m%)
qpss: time = 2.726 ns     (72.6 %), step = 4.936 ps     (494 m%)
qpss: time = 2.777 ns     (77.7 %), step = 2.705 ps     (271 m%)
qpss: time = 2.826 ns     (82.6 %), step = 3.1 ps       (310 m%)
qpss: time = 2.875 ns     (87.5 %), step = 3.366 ps     (337 m%)
qpss: time = 2.928 ns     (92.8 %), step = 3.661 ps     (366 m%)
qpss: time = 2.976 ns     (97.6 %), step = 4.372 ps     (437 m%)
2nd_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
:
:
25th_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
Newton iter = 1, convNorm = 28.3, maximum dI(rif:p) = 9.16928 uA, took 7.58 s.
:
:

```

In this example, four Newton iterations were performed to reach the steady-state solution. Information about the QPSS analysis including the steady-state solution print at the end.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using QPSS Analysis Effectively

---

```
.
.
.
1st_Transient_Integration:
.
.
25th_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
Newton iter = 2, convNorm = 1.35, maximum dI(L1:1) = 927.427 nA, took 8.29 s.
1st_Transient_Integration:
.
.
25th_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
Newton iter = 3, convNorm = 4.21e-03, maximum dI(q56:i_extra) = -20.204 nA,
took 8.24 s.
1st_Transient_Integration:
.
.
25th_Transient_Integration:
.....9.....8.....7.....6.....5.....4.....3.....2.....1.....0
Newton iter = 4, convNorm = 1.05e-06, maximum dI(q56:i_excess) = -478.108 fA,
took 5.78 s.
qpss: The steady-state solution was achieved in 6 iterations.
Number of accepted qpss steps = 360 each in 25 time intervals.
Starting spectrum calculation.
Total time required for qpss analysis 'qpss' was 42.83 s.
.
.
.
```

Occasionally, you might see warning messages such as the following

```
Minimum time step used. Solution might be in error.
```

or

```
Junction current exceeds 'imelt'. The results computed by Spectre are now incorrect
because the junction current model has been linearized.
```

You can ignore these warning messages if they appear in the early stage of QPSS iterations. They might be caused by bad starting integration conditions and do not affect the final solution. However, if they appear in the final iteration, the solution might be in error.



## References

- [1] A. Allgower and K. Georg, *Numerical Continuation Methods*, Springer-Verlag, New York, 1990.
- [2] L. O. Chua and A. Ushida, "Algorithms for computing almost periodic steady-state response of nonlinear systems to multiple input frequencies," *IEEE Transactions on Circuits and Systems*, vol. 28, pp. 953-971, 1981.
- [3] D. Feng, J. Phillips, K. Nabors, K. Kundert, and J. White, "Efficient computation of quasi-periodic circuit operating conditions via a mixed frequency/time approach," Submitted to *Proceedings of the 36th Design Automation Conference*, June 1999.
- [4] K. Kundert, J. White, and A. Sangiovanni-Vincentelli, "A mixed frequency-time approach for distortion analysis of switching filter circuits," *IEEE Journal of Solid State Circuits*, vol 24, pp. 443-451, 1989.
- [5] P. Lancaster and M. Tismenetsky, *The Theory of Matrices*, Academic Press, second ed., 1985.
- [6] Y. Saad, *Iterative methods for sparse linear systems*, PWS Publishing Company, 1996.
- [7] R. Telichevesky, J. White, and K. Kundert, "Efficient steady-state analysis based on matrix-free krylov-subspace methods," in *Proceedings of 32rd Design Automation Conference*, June 1995.
- [8] \_\_\_\_\_, "Efficient AC and noise analysis of two-tone RF circuits," in *Proceedings of the 1996 Design Automation Conference*, June 1996.

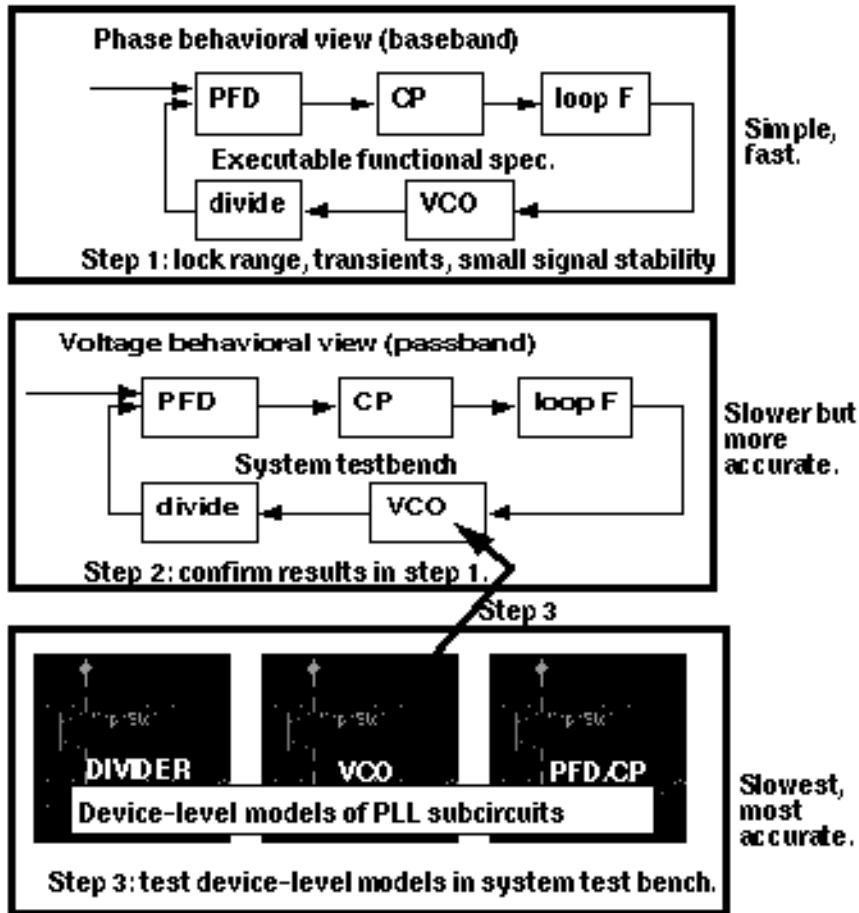
**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Using QPSS Analysis Effectively

---

# Introduction to the PLL library

The models in the phase lock loop (PLL) library support top-down design of PLLs. Figure shows the three steps of the design flow. This appendix describes the first step in detail; all three steps are described briefly.

## PLL Top-Down Design Flow



1. The first step in Figure is to develop an executable specification. The executable specification is an arrangement of fast behavioral models that permits fast architectural studies to separate specification and implementation issues. The executable specification contains baseband models [1,2,3,4,5,6,7,8,9]. (Reference [1] uses the terms “baseband” and “bandpass” explicitly.)

These baseband models suppress clocks and RF/IF carriers. Some literature refers to PLL baseband models as “relative phase” or “phase-domain” models [2]. This appendix uses the latter term. Phase-domain PLL models are exceptionally fast, capture the important non-linear mechanisms, and can be linearized directly for AC analysis.

2. The second step in Figure is to translate the executable specification into a system testbench. The system testbench, unlike the executable specification, is composed of passband models [1]. This Appendix refers to passband models as voltage-domain models because they simulate voltages you can observe in a laboratory.
3. Comparing voltage- and phase-domain voltage-controlled oscillator (VCO) models highlights the difference between the two models. The output of a voltage-domain VCO model is a clock voltage, a periodic signal. The output of a phase-domain VCO model is a voltage numerically equal to phase. If you unwrap the VCO phase, in steady state, it ramps up indefinitely. Unwrapped phase is not periodic. Voltage-domain models describe non-linear effects related to the shapes of the actual RF waveforms. Such waveform effects include spurs and harmonic locking. Harmonic locking occurs when the PLL locks on to a harmonic of the reference.
4. Phase-domain models do not simulate waveform effects. The system testbench is more accurate than the executable specification, but it is still behavioral. Equipped only with behavioral voltage-domain models, the testbench does not simulate device-level effects associated with specific implementations. Examples of such implementation effects are interstage loading, improper bias, and device parasitics.
5. The last step in Figure is to gradually replace the behavioral models in the system testbench with device-level models, one or two blocks at a time. Device-level models check for the previously mentioned implementation problems. The entire PLL is simulated at the device-level only as a final verification step because such simulations are very lengthy.

## Models in the PLL library

The PLL library includes the following phase-domain models:

- Analog multiplier phase detector
- XOR phase detector with bipolar output

The XOR phase detector is not explicitly discussed here because it is very similar to the analog multiplier phase detector. The only difference is that the duty cycle-phase error transfer curve is triangular instead of sinusoidal.

- Three-state digital phase frequency detector (PFD)
- Charge pump (current source version)
- VCO tuning curve (analytic and tabular versions)
- Frequency divider
- Lock indicator

## Introduction to the PLL Library Documentation

The primary system-level specifications captured by this first set of phase-domain models are acquisition time, lock and capture ranges [12], and phase margin. The PFD model also simulates backlash[8]. Backlash is sometimes called “deadband” effect. It is a limit cycle caused by the phase-frequency detector’s inability to linearly reduce its output pulse width to zero as phase error goes to zero.

The remainder of this appendix is divided into two main sections.

The first section introduces phase-domain modeling, describes a feature included to prevent DC convergence problems, and then shows you some examples of using phase-domain models.

The second section explains how to assemble a more complex PLL and discusses an example. The examples are introductory and are not a comprehensive discussion of all applications of phase-domain models.

## Phase-Domain Model of a Simple PLL

### Description

This PLL example, which is built around the simplest phase detector in the library, introduces the fundamentals of phase-domain modeling. [Figure G-1](#) on page 942 shows a voltage-domain model of the example and also some selected waveforms. The phase detector in this case is an ideal analog multiplier.

Figure G-1 Voltage Domain Model

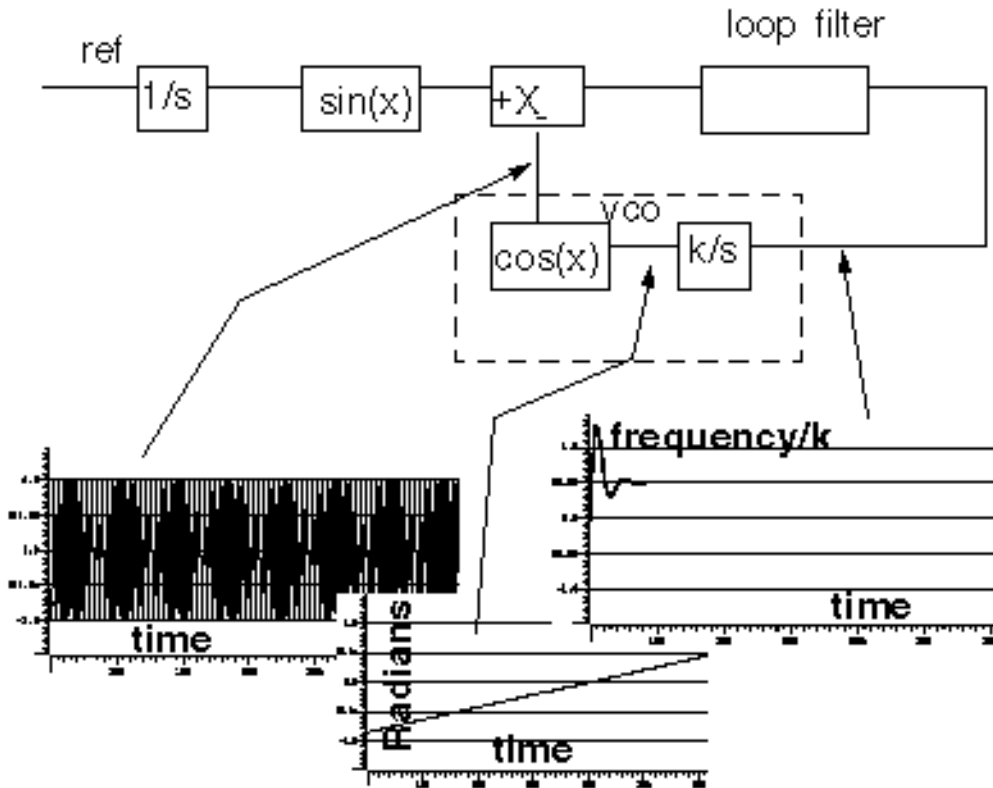


Figure G-2 on page 943 shows the equivalent phase-domain model. The phase-domain model is based on the following trigonometric identity:

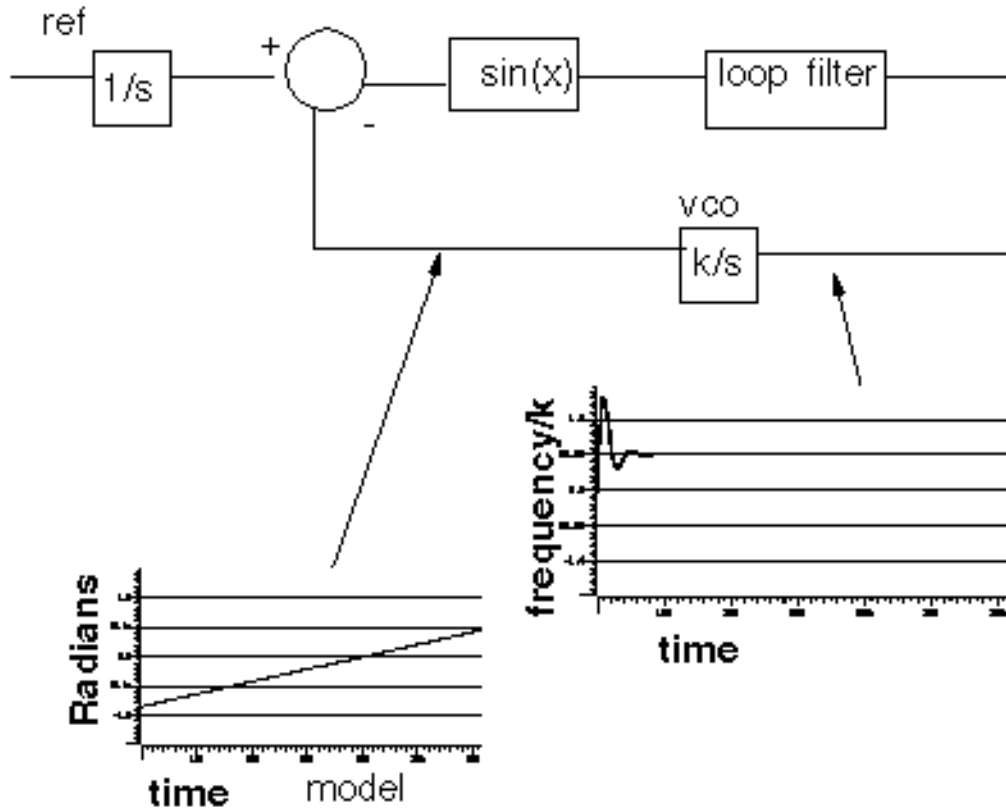
$$\sin(\theta_1) \cdot \cos(\theta_2) = (1/2) \cdot [\sin(\theta_1 + \theta_2) + \sin(\theta_1 - \theta_2)]$$

which, after filtering is approximately

$$(1/2) \cdot \sin(\theta_1 - \theta_2)$$

**Note:**  $\theta_1 + \theta_2 = (w_1 + w_2) \cdot t$  and  $w_1 + w_2$  usually lie far beyond the filter's corner frequency.

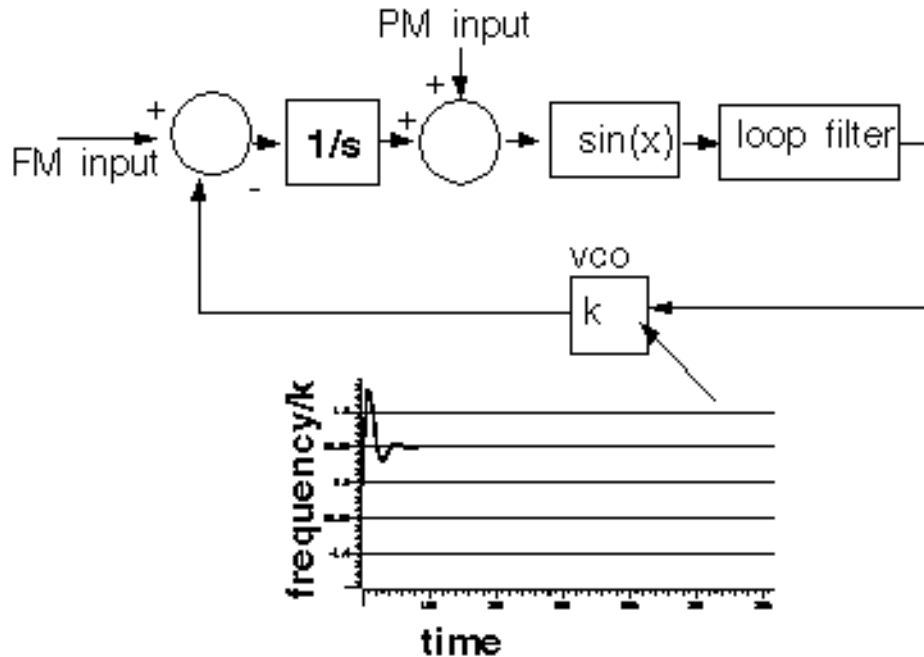
Figure G-2 Phase Domain Model



The phase and frequency waveforms from the phase-domain model match their voltage-domain counterparts, but the simulation runs faster because the oscillatory waveform is not explicitly simulated.

Combining the integrators, as shown in [Figure G-3](#) on page 944, eliminates the integrator outside the feedback loop which might cause a problem if you forget to specify an initial condition. Combining the integrators is also necessary if you build phase-domain models of phase-frequency detectors because, in this case, the non-linearity has memory (hysteresis).

Figure G-3 A More Practical Phase Domain Model

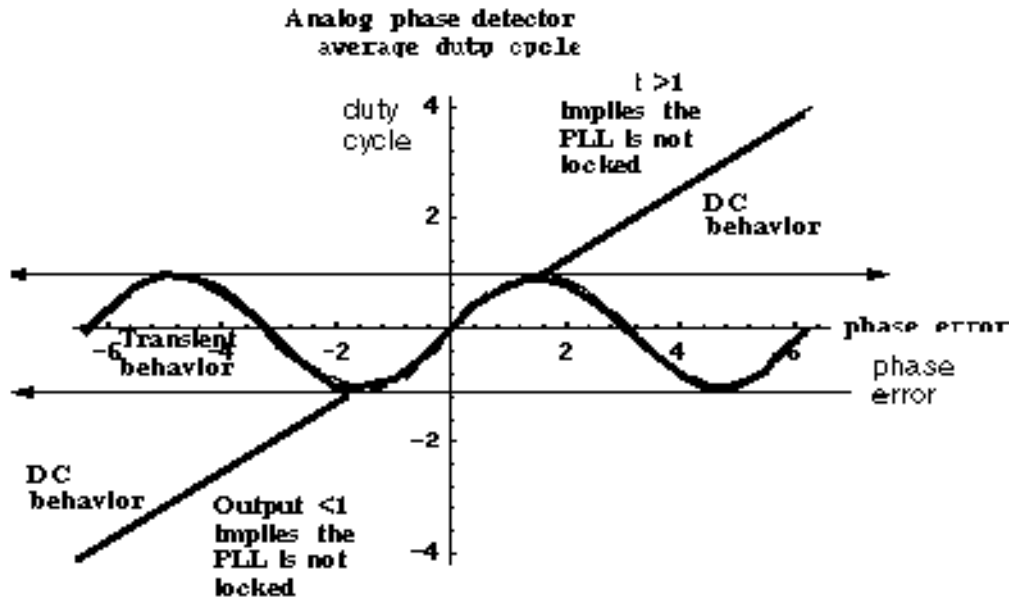


The models in both Figure G-2 and Figure G-3 can fail to achieve DC convergence because the phase detector model either has no DC operating point or because it has an infinite number of operating points.

The sinusoidal function in Figure G-4 on page 945 is the phase detector transfer curve. The phase detector is the only non-linear element in this PLL model. For reasons associated with phase-frequency detectors, the phase detector output is called the *duty cycle*.



Figure G-4 Phase Detector Transfer Curves



If the required duty cycle lies outside  $[-1, 1]$ , the loop is not locked in steady state. If the required duty cycle lies within  $[-1, 1]$ , there are an infinite number of possible phase errors. In either case, a Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) simulation might not converge. The ability of Verilog-A® to perform different tasks for different analyses provides an elegant solution to the DC convergence problems and a quick way to map out lock range. Lock range is the range of input frequencies for which the PLL can maintain lock. (Some literature refers to lock range as hold-in range [8].)

The phase detector model uses the monotonic transfer curve for DC analysis and the true periodic transfer curve for transient analysis. The two transfer curves coincide when the phase error lies in the interval  $[-\pi/2, \pi/2]$ . If the required duty cycle lies within  $[-1, 1]$ , the monotonic transfer curve forces the steady-state phase error to the interval  $[-\pi/2, \pi/2]$ , where the two curves coincide. The equilibrium point is *open-loop-stable*, meaning that at DC the loop gain is a positive real number. This is true because the slope of the transfer curve is positive over  $[-\pi/2, \pi/2]$ . The Nyquist stability criterion is therefore easier to apply. The DC analysis is general enough because only the phase error modulo  $2\pi$  is of interest, and you usually care only about the open-loop-stable operating points. When the loop is not locked, the DC analysis computes a duty cycle with a magnitude greater than one. A duty cycle greater than one is clearly incorrect, but it is much easier to interpret than a convergence error. DC duty cycle is a lock indicator which can be used in a parametric DC analysis to sweep out lock range.

### Example 1: Dynamic Test for Capture Range and Lock Range

The circuit used to dynamically test for capture range and lock range is *example\_analog\_PD* in the *pllLib* library. *Capture range* is the range of input frequencies that the PLL can acquire from an unlocked state. Since acquisition of frequencies near the edge of the capture range involves a pull-in mechanism [1-12], measuring the capture range requires a transient analysis. You can measure capture and lock ranges by slowly sweeping the input frequency and observing the frequency at which the duty cycle begins and ends a long ramp [12]. You must skip the DC analysis to observe the capture limits. Figure G-5 on page 946 plots the VCO control voltage against the input frequency voltage. The input frequency first ramps up and then down. A buffered auxiliary circuit responds to the duty cycle and adds 2.5 volts when the input frequency changes direction. This technique makes the plot easier to read because the forward and reverse sweeps occupy different parts of the vertical scale. In this example, lock range is from 1.36Khz to 3.4Khz, and the capture range is from 1.8Khz to 3Khz.

**Figure G-5 Lock Range and Capture Range**

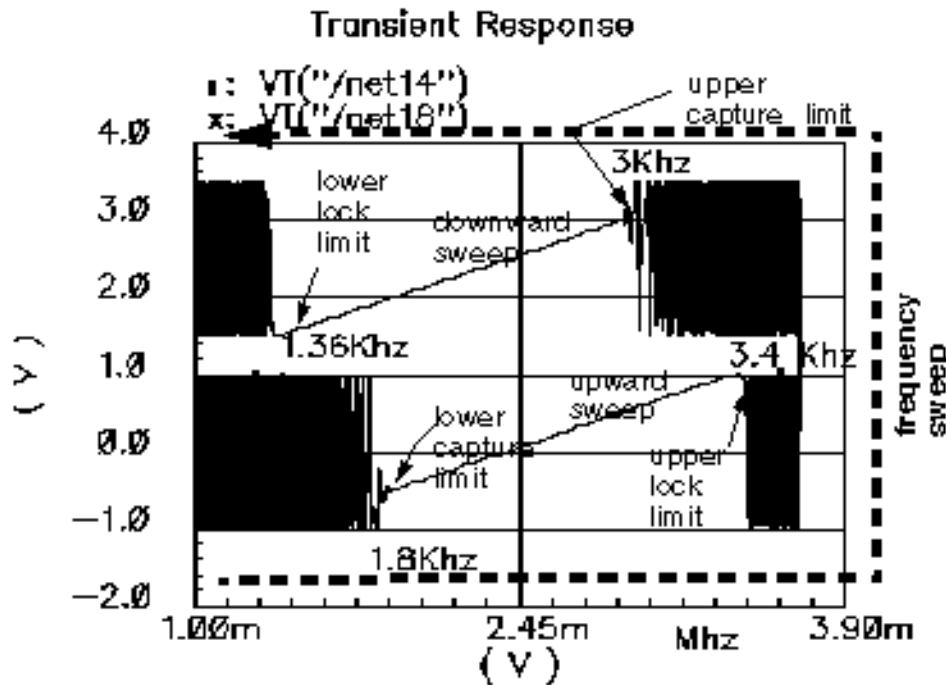
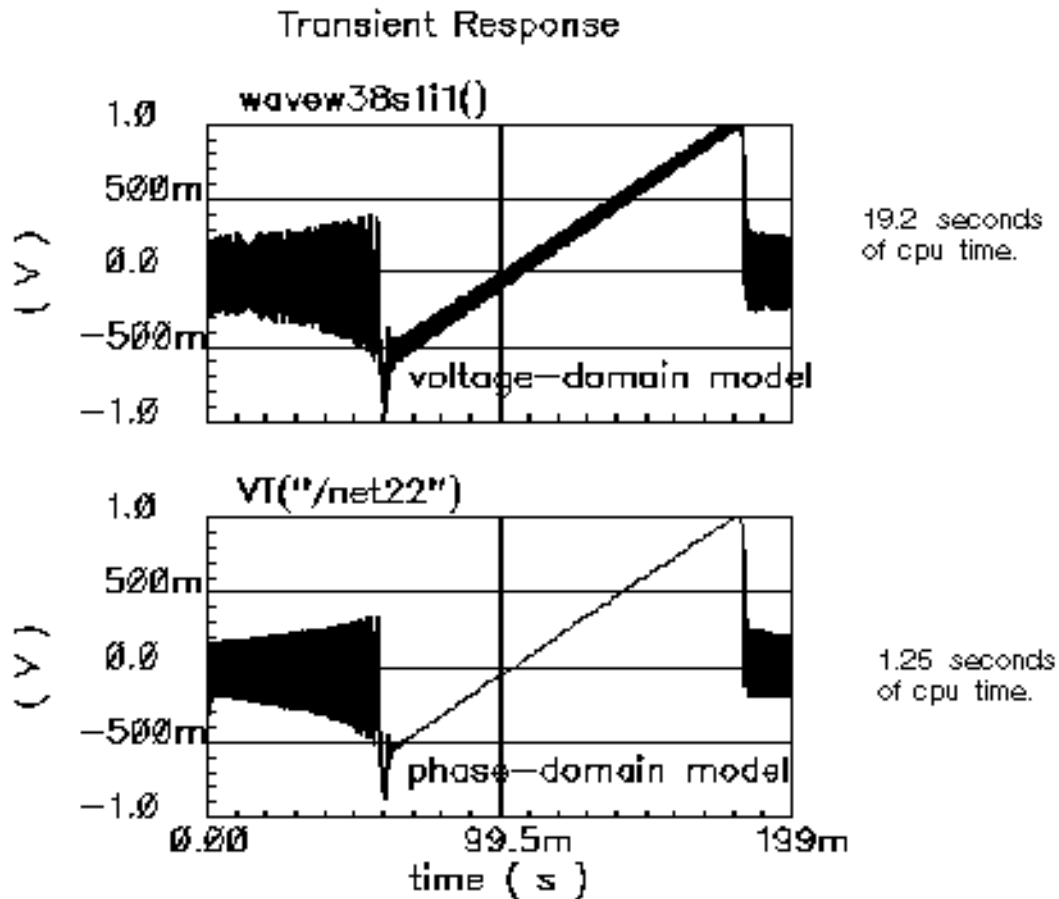


Figure G-6 on page 947 compares VCO control voltages in the forward sweep when computed with voltage- or phase-domain models. The models produce similar results. In this example (2.5Khz center frequency), the phase-domain model is only about 20 times faster than the voltage-domain model.

Figure G-6 VCO Control Voltages Computed with Voltage- and Phase-Domain Models

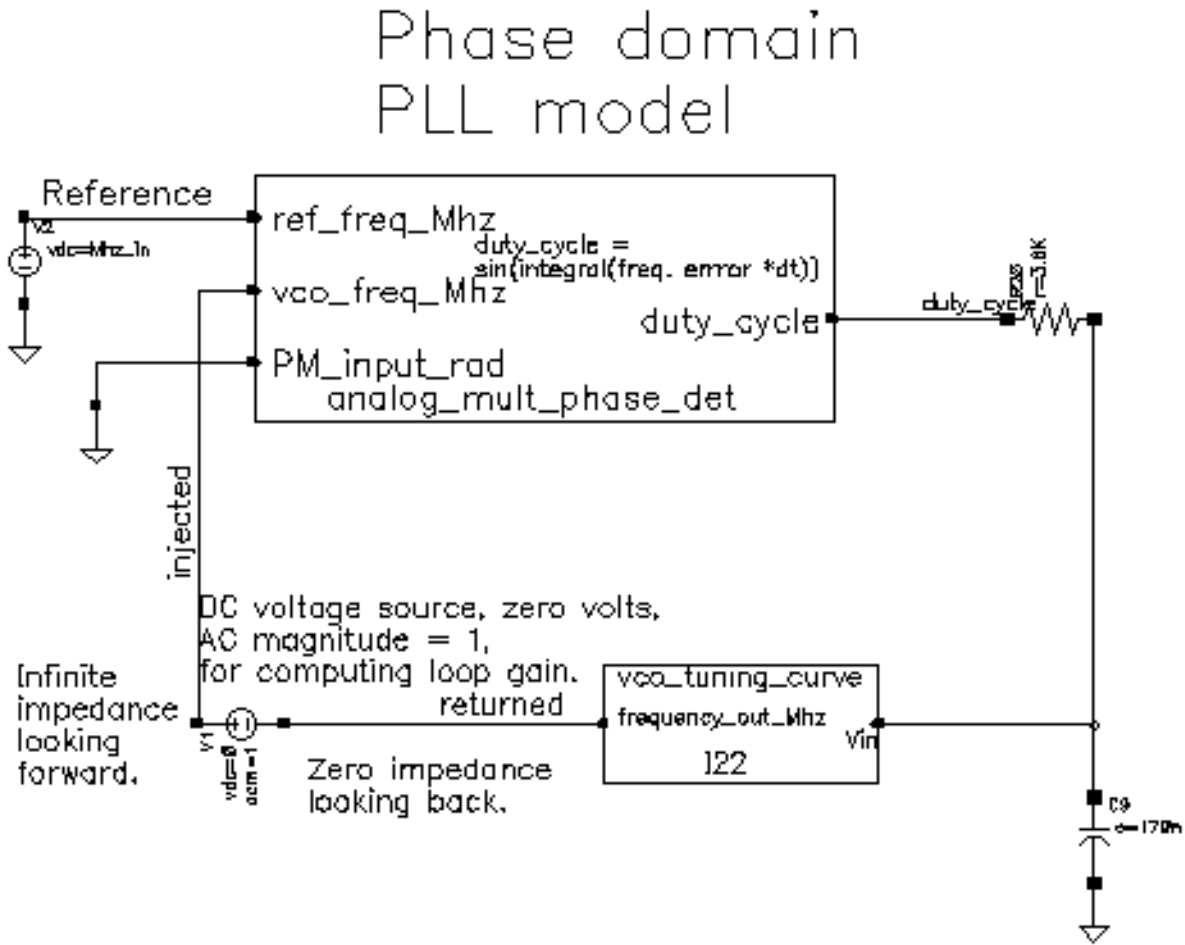


## Example 2: Loop Gain Measurement

Spectre RF cannot perform a useful AC analysis on a voltage-domain model because by design, a voltage-domain PLL model has no DC operating point. However, because Spectre RF linearizes phase-domain models about phase error, and phase error is a meaningful DC quantity, subsequent AC analyses are valid.

This example describes how to compute loop gain with a phase-domain model. [Figure G-7](#) on page 948 shows an analog design environment version of the model shown in [Figure G-3](#) on page 944. The circuit used to measure loop gain is *example\_loop\_gain* in the *pllLib* library.

Figure G-7 Set Up for Loop Gain Measurement



The phase-domain model in Figure G-7, *example\_loop\_gain*, includes a voltage source inserted after the VCO. The DC voltage is zero volts, and the AC magnitude is 1 volt. The new voltage source inserts a test signal without changing the DC operating point. You must insert this source at a point where the impedance looking back is much smaller than the impedance looking forward. The accuracy of the resulting loop gain computation depends on how well this condition is met.

Use the following procedure to compute the loop gain.

Open the *example\_loop\_gain* Schematic

1. In the CIW, choose *File – Open*.

The Open File form appears.

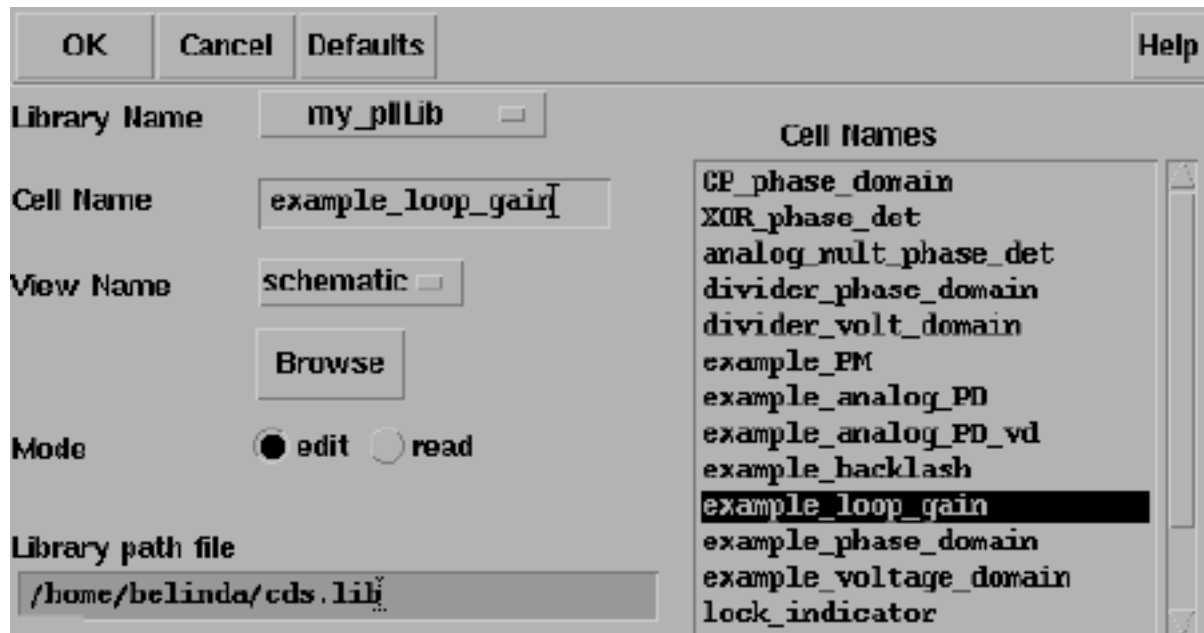
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Introduction to the PLL library

---

2. In the Open File form, choose *my\_pllLib* in the *Library Name* cyclic field. Choose the editable copy of the pllLib library you created. (You can create an editable copy of the *pllLib* in the same way as is described for the *rfExamples* library in [Chapter 3, “Setting Up for the Examples.”](#))
3. Choose *example\_loop\_gain* in the *Cell Names* list box.

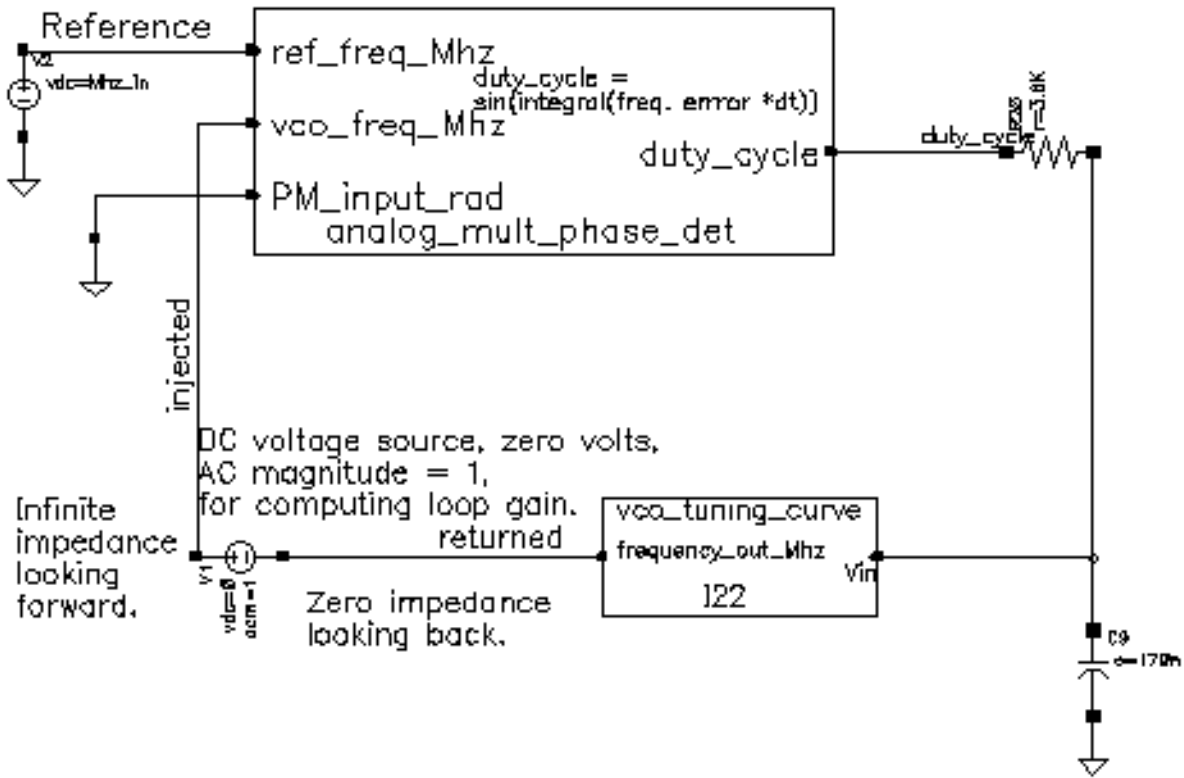
The completed Open File form appears like the one below.



4. Click *OK*.

The Schematic window for the *example\_loop\_gain* circuit appears.

## Phase domain PLL model



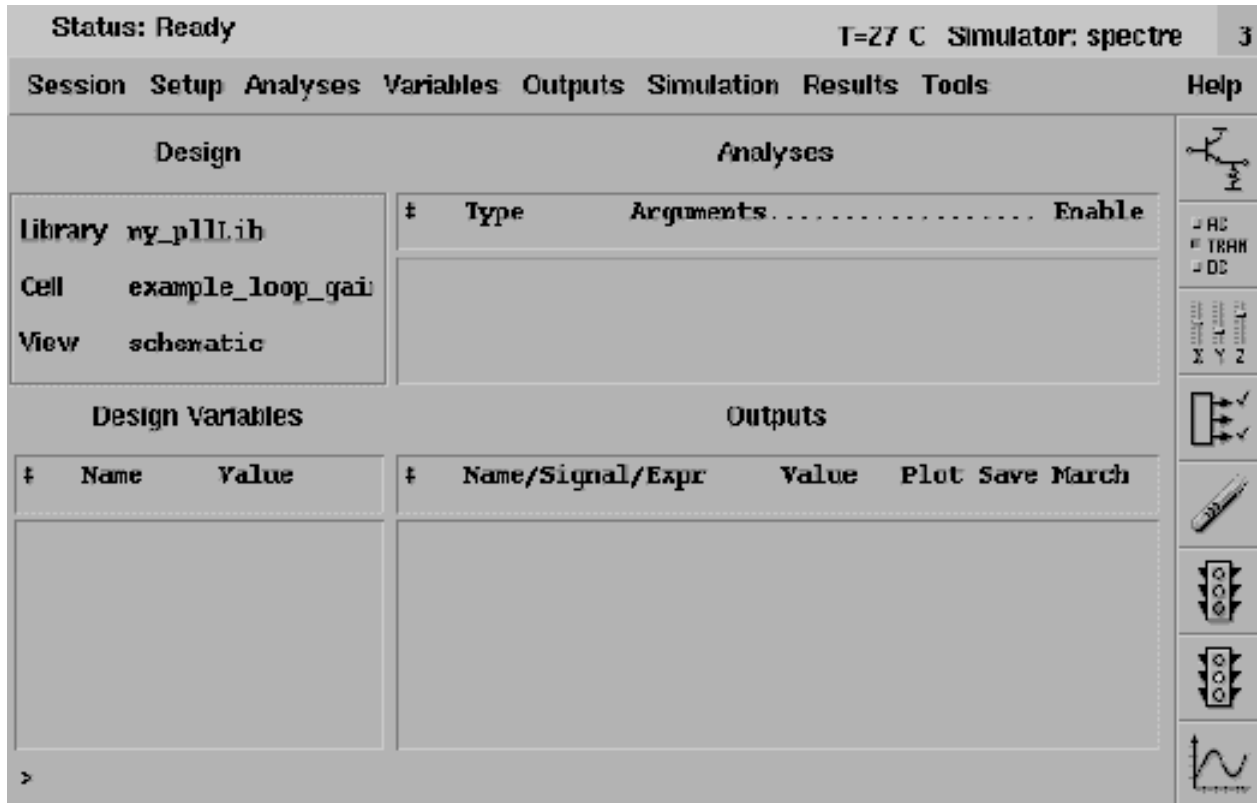
5. In the Schematic window, choose *Tools– Analog Environment*.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Introduction to the PLL library

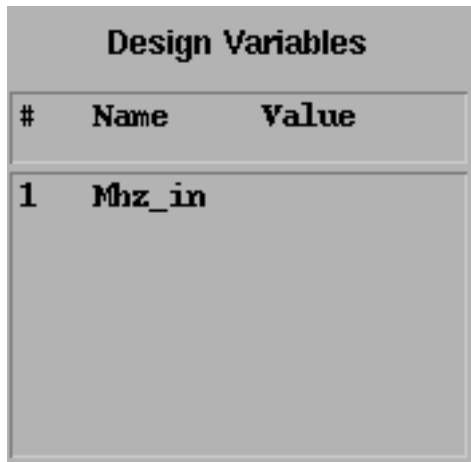
---

The Simulation window opens. This window is also called the Cadence® Analog Circuit Design Environment.



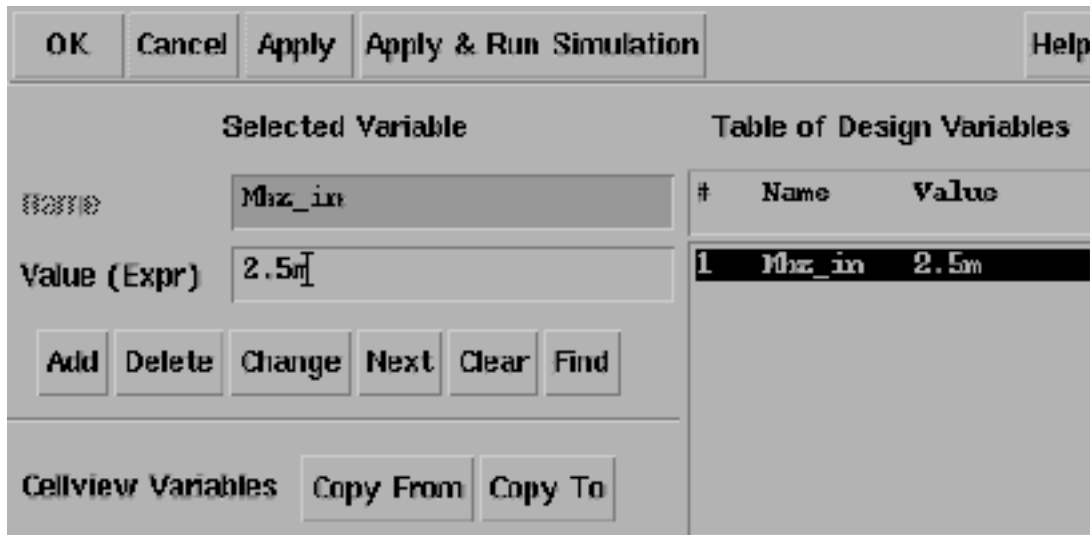
## Setting Up the Design Variables

1. In the Simulation window, choose *Variables—Copy From Cellview* to copy variables from the schematic to the Simulation window. *Mhz\_in* displays in the *Design Variables* area of the Simulation window.



#	Name	Value
1	Mhz_in	

2. In the Simulation window, choose *Variables—Edit*, to provide a value for the *Mhz\_in* variable.
3. The Editing Design Variables form appears.



Selected Variable		Table of Design Variables		
Name	Value (Expr)	#	Name	Value
Mhz_in	2.5m	1	Mhz_in	2.5m

4. In the *Value (Expr)* field, type `2.5m` for the value of *Mhz\_in* and click *Change*.
5. In the Editing Design Variables form, click *OK*.

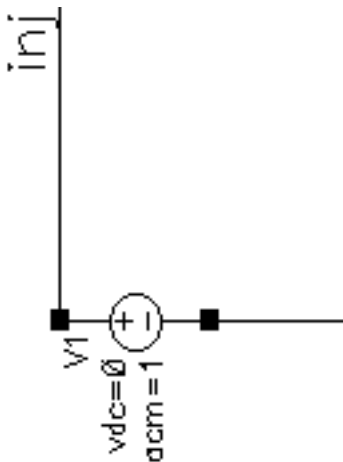


- The new value for *Mhz\_in* displays in the *Design Variables* area of the Simulation window.

Design Variables		
#	Name	Value
1	Mhz_in	2.5m

### Setting Up the AC and DC Simulations

Set up both AC and DC analyses. The zero-voltage *vdc* source must be the only source with a non-zero AC magnitude.



When you set up the DC analysis, save the DC data so you can annotate the schematic with DC node voltages.

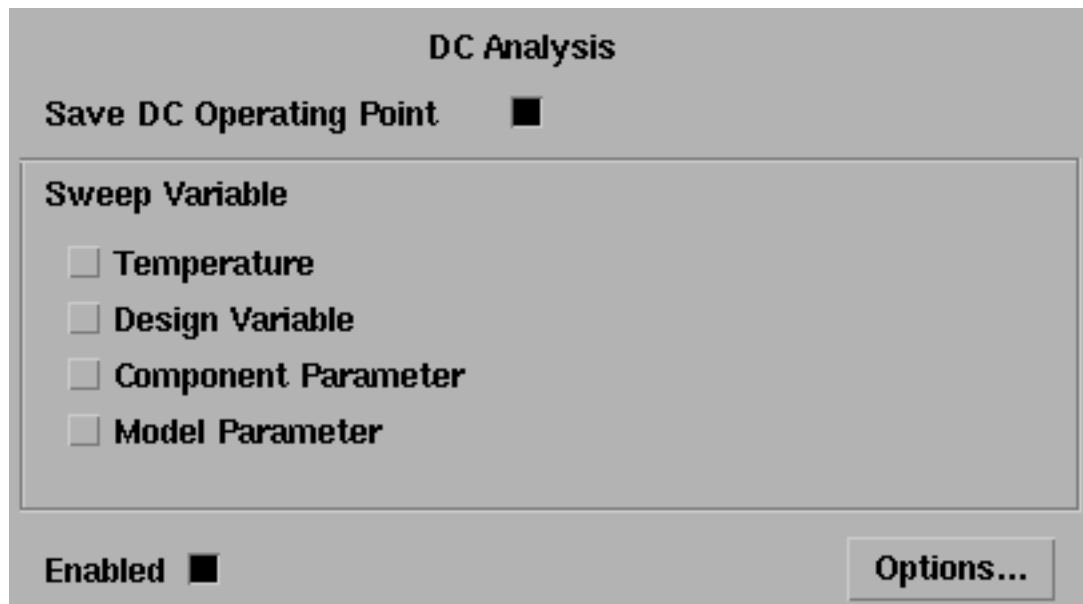
- In the Simulation window, choose *Analysis - Disable* to disable any analyses you ran previously. (Check the Simulation window to verify whether or not any analysis is enabled.)

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Introduction to the PLL library

---

2. In the Simulation window, choose *Analysis - Choose* to display the Choosing Analyses form.
3. Click *dc* to set up the DC analysis.
4. In the DC Analysis area
  - a. Highlight *Save DC Operating Point*.
  - b. Highlight *Enabled*.



5. Click *ac* to set up the AC analysis.
6. In the AC Analysis area
  - c. Highlight *Frequency* for *Sweep Variable*.
  - d. Highlight *Start - Stop* for *Sweep Range*. Type 10 in the *Start* field and 20k in the *Stop* field.
  - e. Select *Automatic* in the *Sweep Type* cyclic field.

f. Highlight *Enabled*.

**AC Analysis**

**Sweep Variable**

Frequency  
 Design Variable  
 Temperature  
 Component Parameter  
 Model Parameter

**Sweep Range**

Start-Stop      Start       Stop   
 Center-Span

**Sweep Type**

Automatic ▾

**Add Specific Points**

**Enabled**      

7. Click *OK* in the Choosing Analyses form.

8. Both analyses are displayed in the Simulation window.

Analyses					
#	Type	Arguments.....			Enable
1	dc	t			yes
2	ac	10	20K	Auto.. Star..	yes

### Run the Analyses

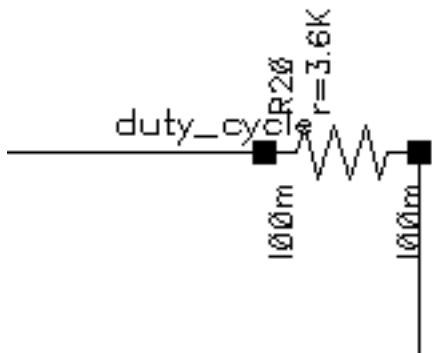
1. To run the analyses, choose *Simulation – Netlist and Run* in the Simulation window.

The output log file appears and displays information about the simulation as it runs.

Look in the CIW for a message that says the simulation completed successfully.

### Displaying the DC Voltages on the Schematic Nodes

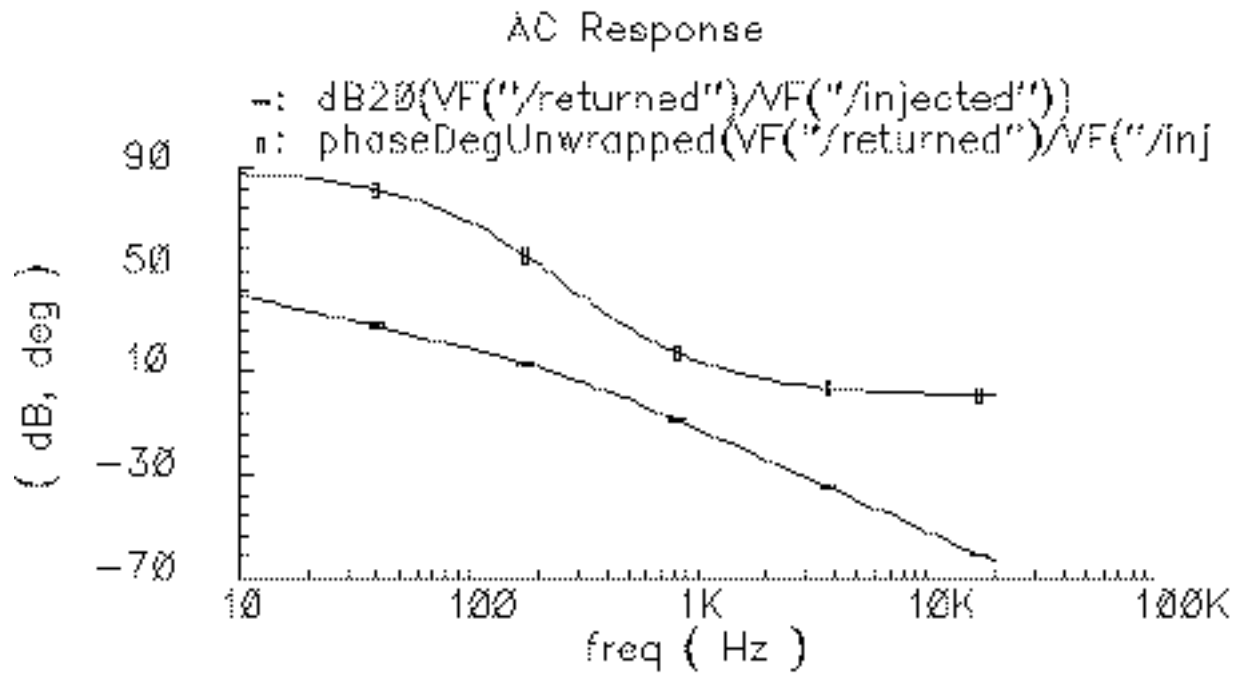
- In the Simulation window, choose *Results – Annotate – DC Node Voltages* to display the node voltages on the schematic. The DC operating point for the net called *duty\_cycle* must remain between -1 volt and 1 volt.



If the operating point falls outside the interval [-1, 1] volt, the loop is not locked and the AC analysis is invalid.

### AC Response as Gain and Phase

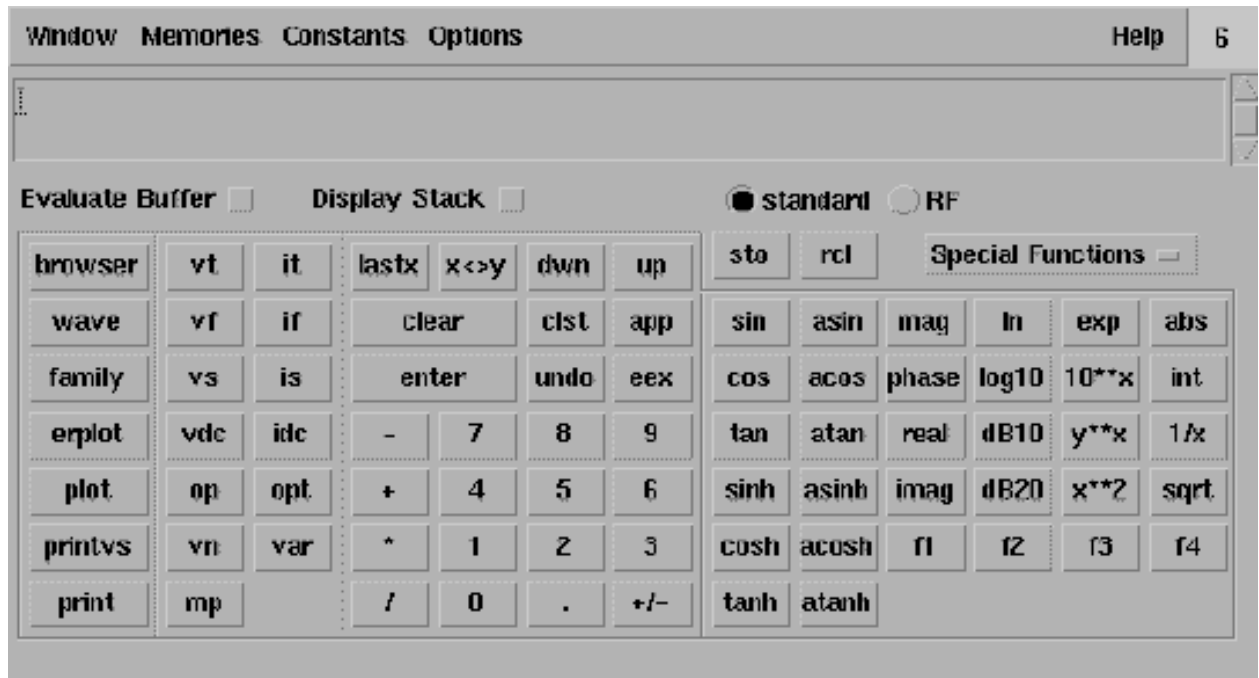
1. In the Simulation window, choose *Results – Direct Plot – AC Gain & Phase* and follow the prompts at the bottom of the Schematic window.
2. *Select first point*—Select the net labeled *returned* in the schematic.
3. *Select second point*—Select the net labeled *injected* in the schematic.
4. The Waveform window displays two curves.
5. The top curve plots phase.
6. The bottom curve plots gain.



# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Introduction to the PLL library

7. In the Waveform window, choose *Tools—Calculator* to open the Waveform Calculator.



8. In the Calculator, click the *wave* button (on the left).

9. Then, in the Waveform window, select the phase curve (on the top).

```
wavew5s1i1()
```

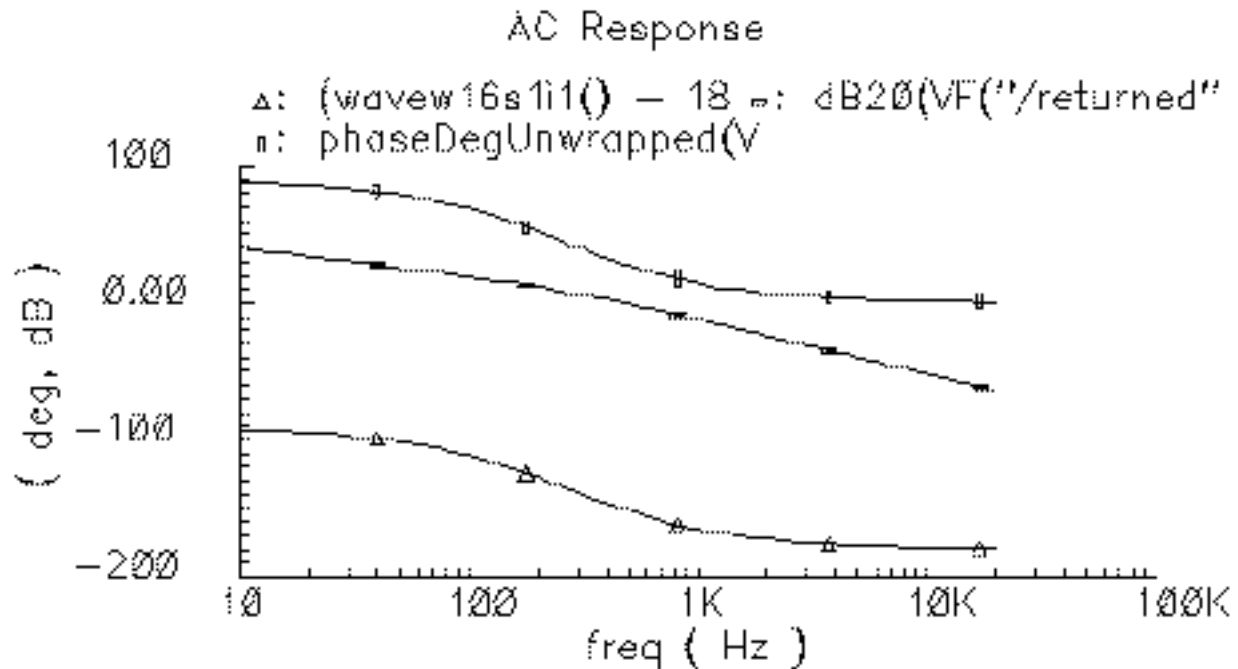
10. In the Calculator, to perform the calculations algebraically, choose *Options—Set Algebraic*.

11. Subtract 180 from the phase waveform—In the Calculator, click the subtraction symbol (-) followed by the numbers 180.

12. The Calculator buffer should look similar to the following

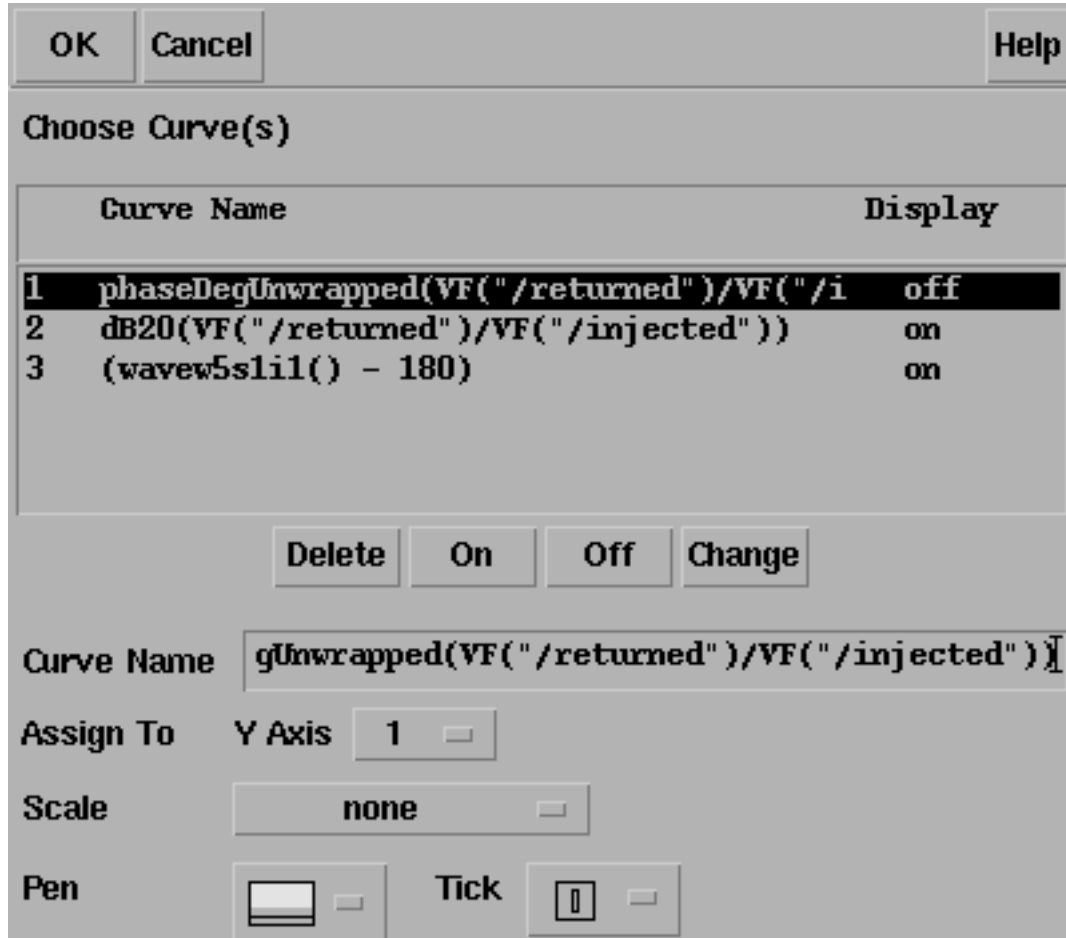
```
wavew5s1i1()-180
```

13. Click the *plot* button to plot the calculated waveform.



14. To remove the original phase curve from the Waveform window, in the Waveform window
- Choose *Curves—Edit* to display the Curves form.
  - In the Choose Curves list box, highlight the original phase curve.
  - Click *Off*.

d. The Curves form looks similar to the following.



OK Cancel Help

**Choose Curve(s)**

	Curve Name	Display
1	phaseDegUnwrapped(VF("/returned")/VF("/i	off
2	dB20(VF("/returned")/VF("/injected"))	on
3	(wavew5slil() - 180)	on

Delete On Off Change

Curve Name

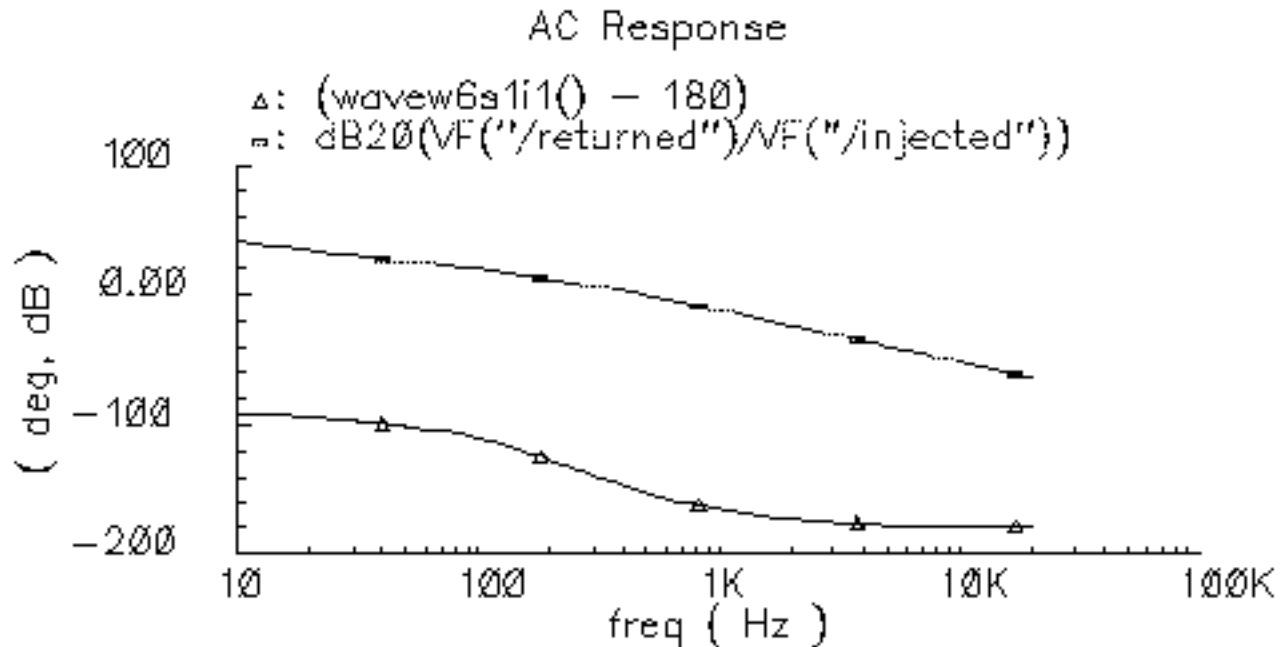
Assign To Y Axis

Scale

Pen  Tick



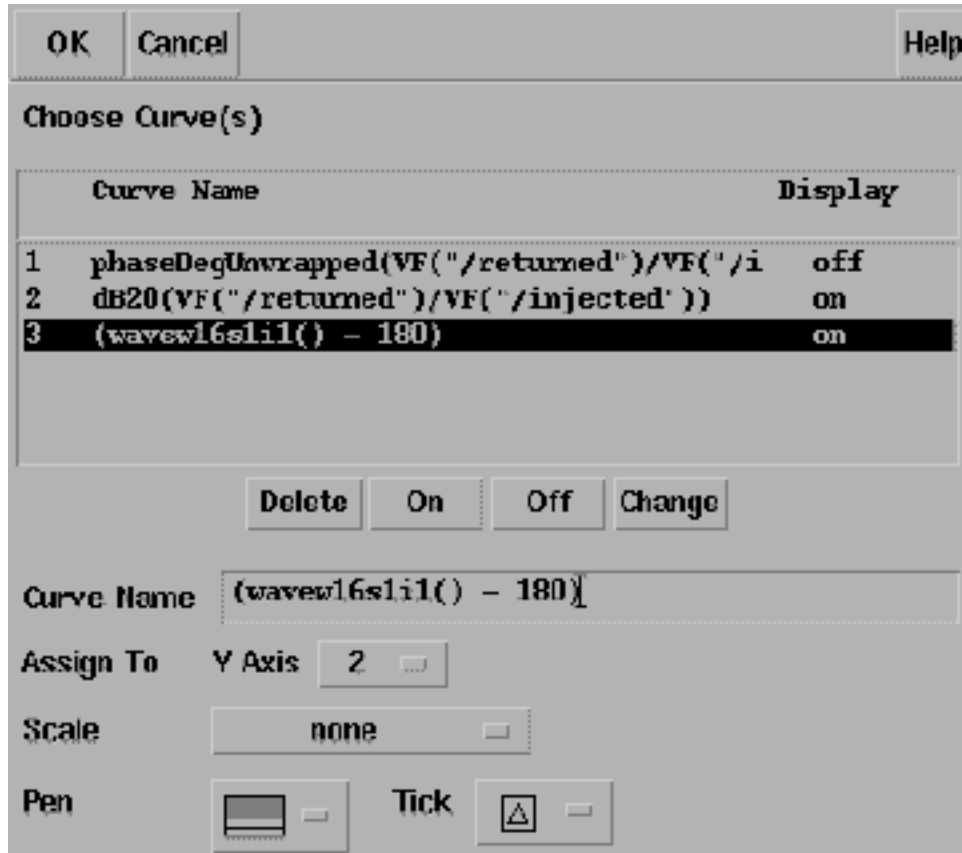
The original phase waveform is no longer displayed in the Waveform window.



15. To create the Bode plot

- a. If necessary, in the Waveform window, choose *Curves – Edit* to display the Curves form.
- b. In the Curves form, select the shifted (by 180 deg) phase curve
- c. In the Curves form *Assign to Y Axis* cyclic field, select 2.

d. The Curves form looks similar to the following.





	Curve Name	Display
1	phaseDegUnwrapped(VF("/returned")/VF("/i	off
2	dB20(VF("/returned")/VF("/injected' ))	on
3	(wavew16sli1() - 180)	on

Curve Name: (wavew16sli1() - 180)

Assign To Y Axis: 2

Scale: none

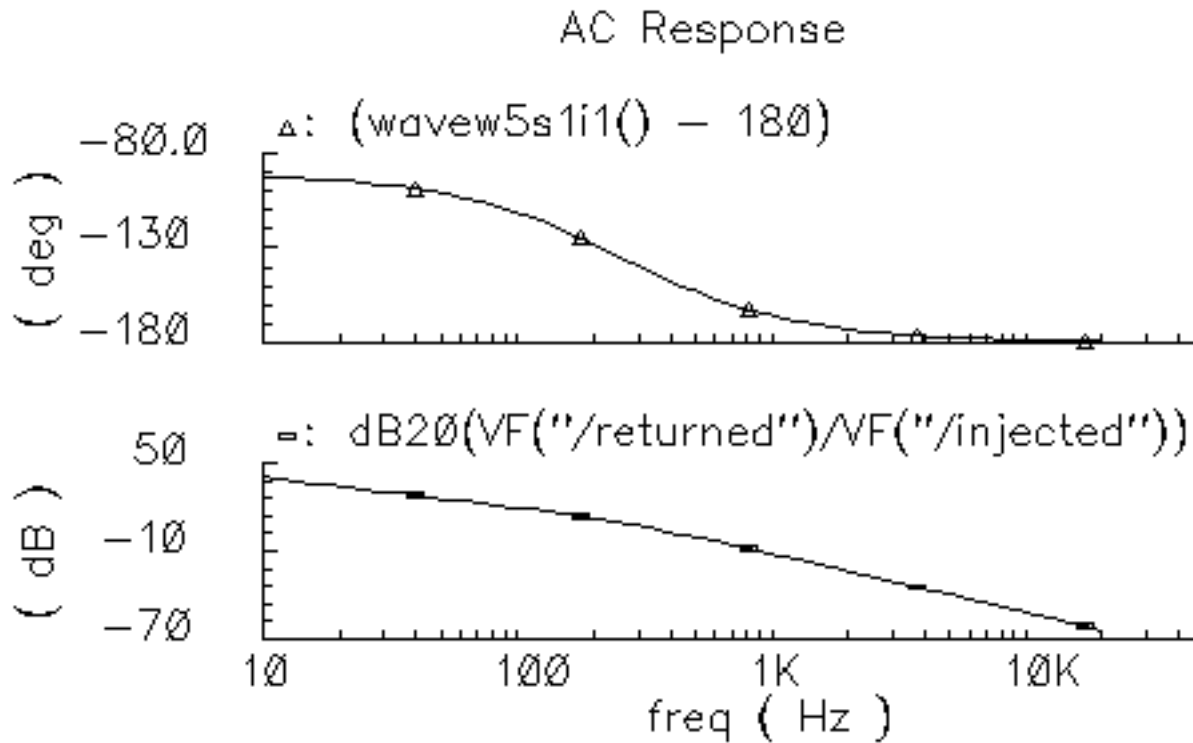
Pen:  Tick: 

16. Click *OK* in the Edit Curves form.

17. In the Waveform window, choose *Axes—To Strip* to change the display as follows.

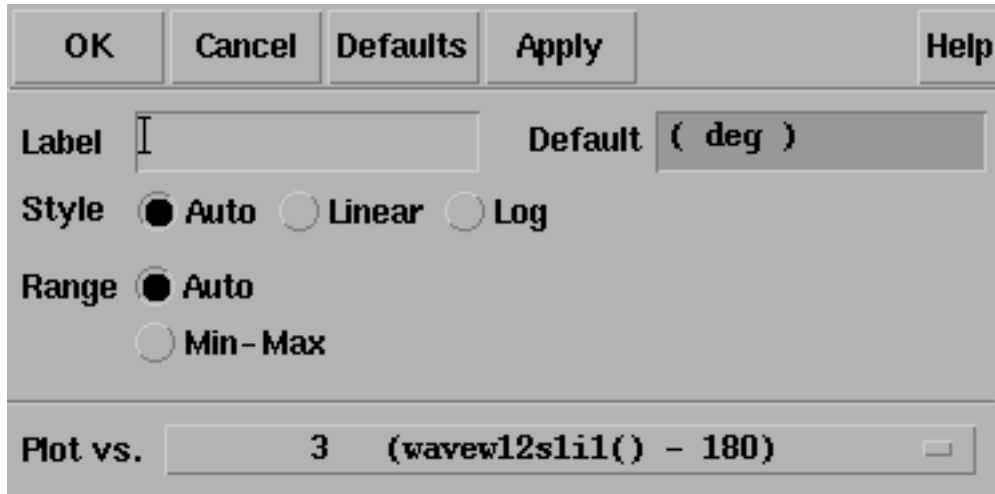
This produces a Bode plot (magnitude and phase) of the loop gain in the Waveform window Shown in Figure G-8.

Figure G-8 Bode Plots, Magnitude and Phase of Loop Gain



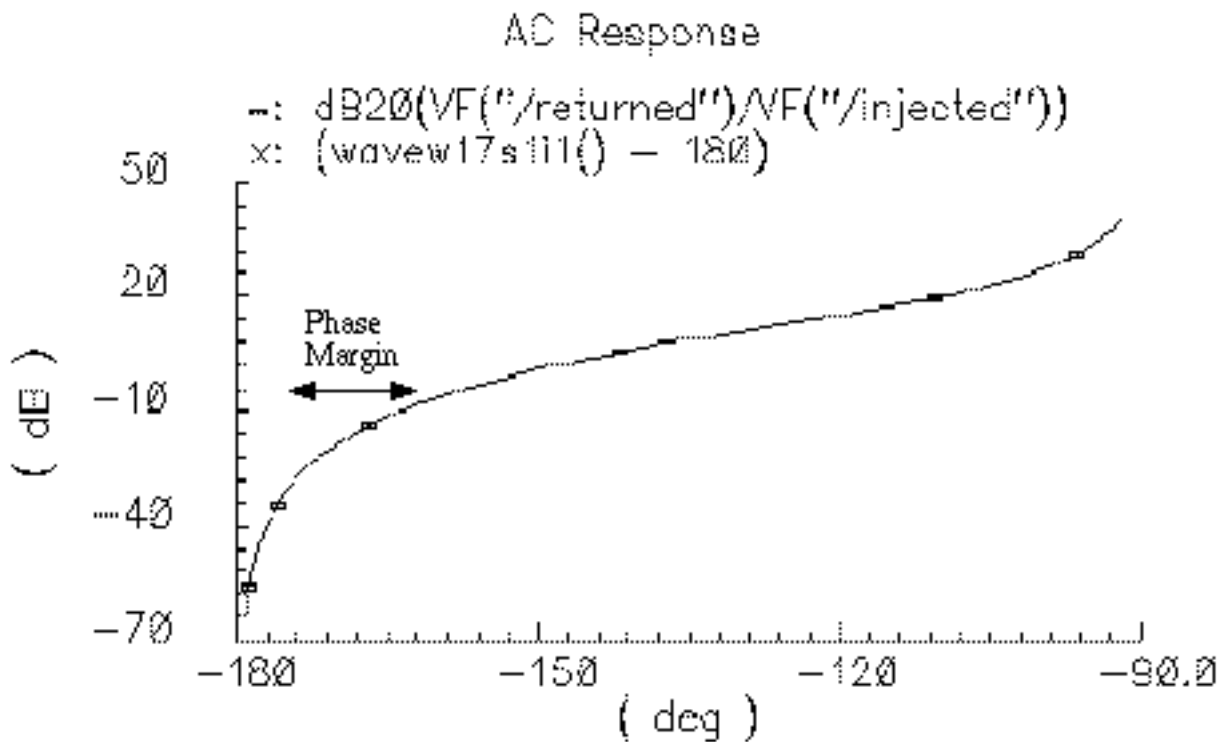
18. To generate a Nichols chart (dB versus degrees) from which you can pick off phase and gain margins,
  - a. In the Waveform window, choose *Axes – X Axes*
  - b. In the *Plot vs. cyclic* field select the phase curve you created that is 180 degrees out of phase.

c. Click *OK*.



You now have a Nichols chart like the one in Figure G-9. The phase margin is 30 degrees.

Figure G-9 Nichols Chart of Loop Gain



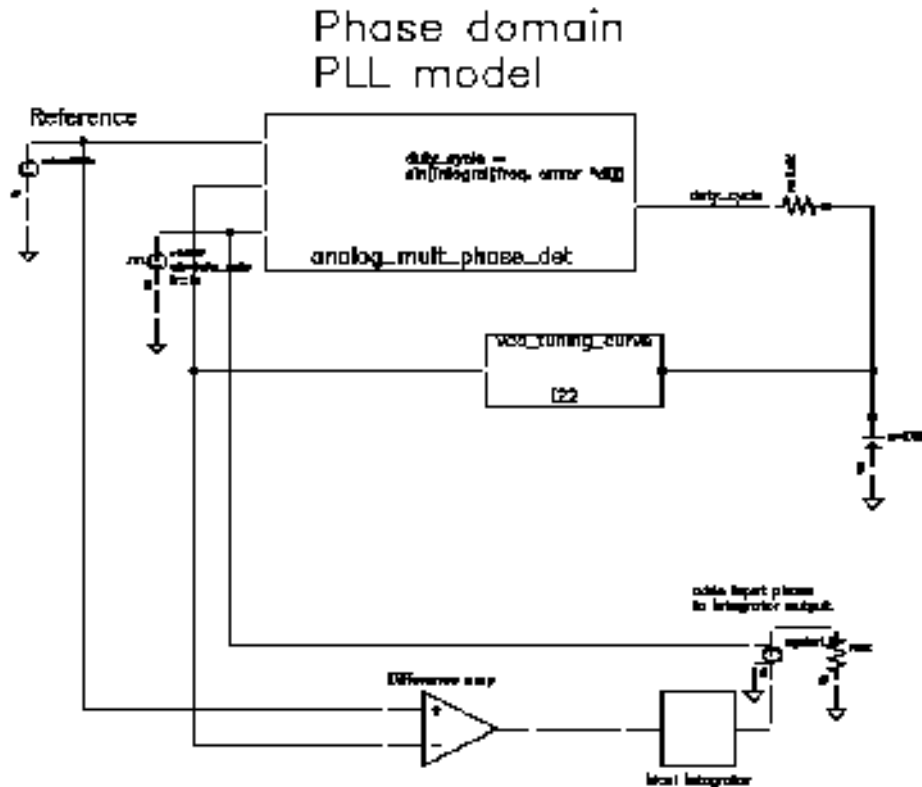
To compute phase margin directly do the following:

1. In the Waveform Calculator, click *vf*.
2. In the Schematic window, click first on the return node and then on the injected node.
3. In the Waveform Calculator, click the *divide* button.
4. In the Waveform Calculator Special Functions menu, choose *phase margin* followed by *print*.
5. Add 180 degrees to the expression in the waveform calculator then choose *print* from the Special Functions menu.
6. The Results Display Window displays the phase margin.

### **Example 3: PM Input**

The circuit used to test for PM input is *example\_PM* in the *pllLib* library. The PM (phase modulation) input pin is useful if the PLL is used as a modulator or demodulator, but it also provides a convenient place to perturb the PLL to assess large signal stability. [Figure G-10](#) on page 966 shows a test circuit for such a stability check.

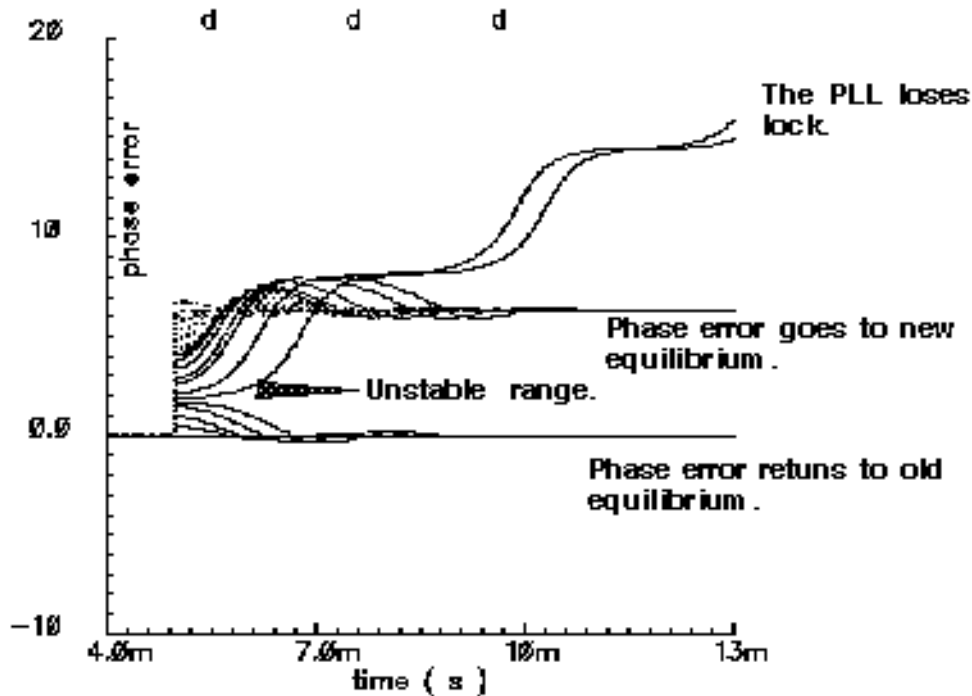
Figure G-10 Stability Check Using the PM Input



This PLL is very simple but yet different from the previous example. It was modified to produce more interesting results. The lower circuitry requires explanation. The difference amplifier computes frequency error and converts it from Mhz to rad/sec. The integrator computes VCO and reference contributions to phase error. The voltage-controlled-voltage-source at the end adds the input phase stimulus to compute total phase error. The difference amplifier and integrator are from the *ahdLib*.

The input phase is a delayed step. The delay makes the initial phase error easy to read. A parametric analysis on the phase error's step response with respect to the size of the input phase step reveals some interesting behavior. Figure G-11 on page 967 shows the family of phase error step responses produced by the parametric analysis. The external integrator is intentionally not a circular integrator like the one inside the phase detector model. For large and small steps in input phase, the PLL settles into equilibrium, possibly a new one. However, a narrow intermediate range of steps puts the PLL into an unstable mode. The references examine this behavior in mathematical detail [1,4,6,10]. This example shows one way to assess large-signal stability and to demonstrate that the phase-domain models capture the major non-linear mechanisms.

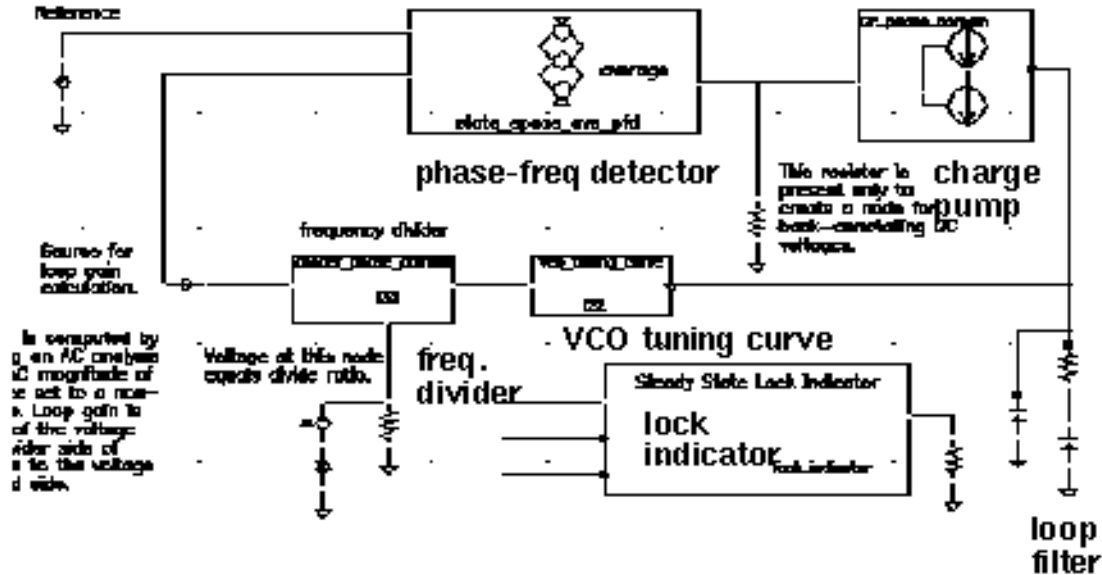
Figure G-11 Phase Error Response to Step PM Input



## Modeling a PFD-Based PLL

Figure G-12 on page 968 shows a block diagram of a typical PLL with a phase-frequency detector. This section describes how to specify each component in Figure G-12 and briefly explains what each model does.

Figure G-12 PLL Block Diagram



## VCO

The VCO is modeled by its tuning curve. The tuning curve characterizes the relationship between the input voltage and the output frequency. The input to the VCO model is the loop filter output voltage, also called the VCO control voltage. The VCO output is a voltage representing the VCO's instantaneous frequency in Mhz. Therefore, when the VCO operates at 2 Mhz, the model output is 2 Volts.

The VCO tuning curve is generally nonlinear and can be specified in one of two ways:

- With the coefficients of a fourth order polynomial
- With a look-up table

**Polynomial tuning curve:** The input voltage is internally clamped to the nearest end point if it moves outside the interval  $[min-vco-input-voltage, max-vco-input-voltage]$ . Although the input voltage may fall outside the interval, the output behaves as though the input voltage value is at the end points. Within the interval, the output is a fourth order polynomial in the quantity,  $V_{input}$  minus the free running voltage. When the input voltage equals the free running voltage, the output frequency equals the free running frequency. The scale factor scales the entire polynomial and has a default value of 1. The scale factor is useful in converting data in Khz/volt, for example, to the required Mhz/volt. The parameters are the coefficients of the polynomial.



# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Introduction to the PLL library

---

**Table look-up tuning curve:** The two parameters are the *scale factor* and the *path to the look-up data*. The look-up model linearly interpolates between data points and linearly extrapolates outside the data interval. The data format is two columns of data delimited by spaces. There is no header, and there are no extra lines at the end. The first column is input voltage. The second column is output frequency. The path to the data can be absolute or relative to the netlist. The netlist is usually stored at

```
<home>/simulation/ckt_name/spectre/schematic/netlist/input.scs
```

but you can choose a different location.

If the data is at

```
<home>/data/table
```

and the netlist is as shown above, the relative path is

```
../../../../../../../../data/table.
```

```
Frequency Divider
```

The frequency divider is essentially a simple gain element. It takes an input voltage that represents frequency in Mhz, and then scales it by the divide ratio to generate an output voltage that represents the divided frequency. The divide ratio is numerically equal to the voltage on the control pin. If the divide ratio drops below 0.001, the model assigns it to 0.001 and issues a warning. This assignment prevents division by zero during simulation.

## Charge Pump

The charge pump transforms the duty cycle into the expected average current sourced or sunk by the charge pump. You define the maximum source and sink currents, and they can be different from each other. If the charge pump output voltage exceeds the rails you define, the output voltage is clamped to the rail through a 0.001 Ohm resistance. The other parameters are the leakage resistance and open circuit voltage. These last two parameters specify the Thevenin equivalent circuit of a leakage path. The leakage path can source or sink current depending on the open circuit voltage.

## Loop Filter

The loop filter is entered component-for-component.

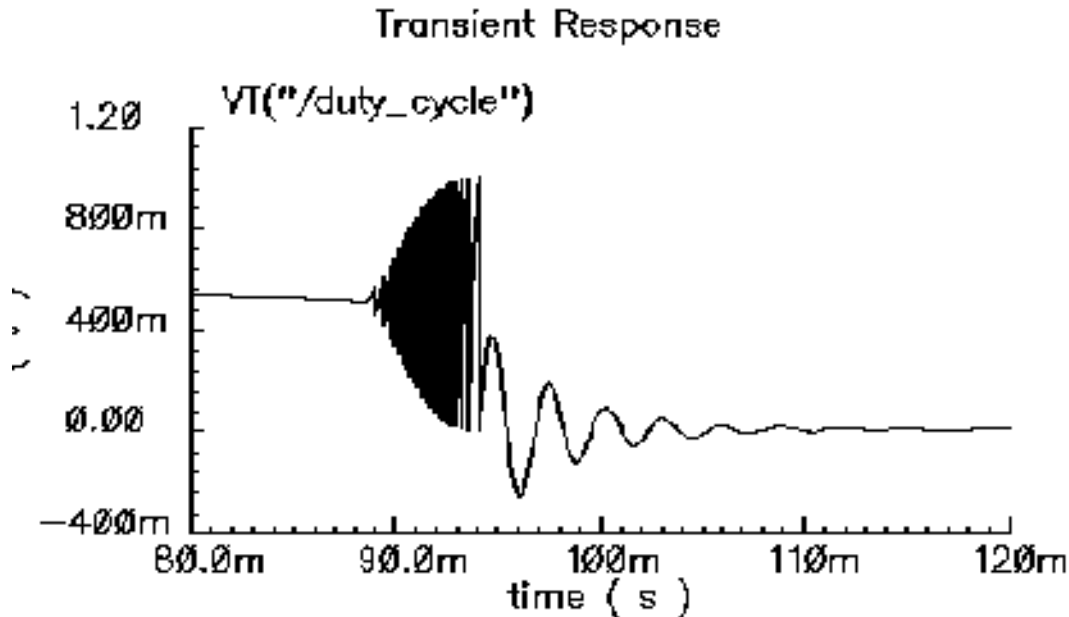
### State-Space Averaged PFD (Phase-Domain Phase-Frequency Detector Model)

The phase-frequency detector (PFD) model approximates average behavior of a digital, three-state, phase frequency detector. This is the most complicated model in the PLL library. The term *state-space averaged* is borrowed from the power electronics field [14]. The charge pump currents for the three PFD states are averaged together with a duty cycle much like voltages are averaged together with a duty cycle in a switch-mode power supply model. The PFD inputs are voltages representing the reference and divider output frequencies in Mhz. The output is a voltage that when multiplied by the maximum charge pump current, numerically equals the average charge-pump output current. The PFD output is a duty cycle. When the frequency error is large, the duty cycle is a smooth waveform directly related to the normalized frequency error [1,4]. When the frequency error is small, an integrator inside the PFD model converts frequency error to phase error, and the duty cycle is proportional to the phase error. The duty cycle starts jumping to zero (or resets) as it changes from frequency-mode to phase-mode.

As phase error enters a deadband determined by the minimum-on-time parameter and reference frequency, the model computes a duty cycle pulse with magnitude one and duration equal to the minimum-pulse-width. After the pulse expires, the duty cycle drops to zero until the phase error exits the deadband. As the phase error exits the deadband, the duty cycle increases to a non-zero value. The deadband and fixed-width unity pulse simulate what some texts call *backlash* [8].

The PFD model has two parameters. The first is a numerical option that controls the trade-off between execution speed and accuracy. The *speed\_vs\_accuracy* parameter controls the number of times the internal integrator is reset during the transition from frequency-mode to phase-mode. Too few resets can cause error. Too many resets can needlessly slow run time. The default value of this parameter is 50k. To reduce the number of resets in a slow PLL, and thereby reduce run time, increase the *speed\_vs\_accuracy* parameter to 70k or 100k. To increase the number of resets in fast PLLs, and thereby increase the accuracy, reduce the *speed\_vs\_accuracy* parameter to 10k or 20k. A reasonable setting for the *speed\_vs\_accuracy* parameter produces a duty cycle step response that resets approximately to zero at least 3-10 times before entering the final transient. [Figure G-13](#) on page 971 shows reasonable duty cycle step response.

Figure G-13 Reasonable Duty Cycle Waveform



The other parameter is the *minimum\_on\_time* which controls the backlash. This is the minimum pulse width the PFD can generate. As the phase error decreases, the pulse width drops discontinuously from the minimum pulse width to zero. This effect creates a deadband in the duty cycle versus phase error curve.

Figure G-11 on page 967 was generated with the default *minimum\_on\_time* parameter value of zero  $\mu\text{s}$ . The default value of the *minimum\_on\_time* parameter produces no deadband and no unity pulses. The default deactivates the backlash mechanism.

Figure G-14 on page 972 was generated with a *minimum\_on\_time* parameter value of 0.2  $\mu\text{s}$ . Figure 15a illustrates that the pulses only occur as the phase error enters the deadband. Figure 15b shows the limit cycle created by the backlash. The limit cycle is primarily determined by leakage on the loop filter and the minimum pulse width. Some references suggest biasing the duty cycle away from the deadband or loading the filter down to force the limit cycle frequency to a value in which the loop filter attenuates it. The phase-frequency detector model can help quantify the problem and check the solution.

A pulse is not kicked out upon exiting the deadband because that behavior causes convergence problems for Spectre RF. If phase error is entering the deadband, a pulse at that moment pushes phase error in the same direction it was going, into the deadband. If a pulse occurs as phase error exits the deadband, the pulse drives phase error back into the deadband and Spectre RF has trouble figuring out whether phase error should leave the deadband at all. Fortunately, no significant error is introduced by implementing the pulse only when phase error enters the deadband. In a backlash limit cycle, the feedback loop quickly

drives phase error back into the deadband and a pulse occurs on the way in. The error is in the time the feedback loop takes to return phase error to the edge of the deadband and that is usually small when PLL is in a backlash limit cycle.

(The PLL model that generated Figure G-14 had a center frequency of about 1Mhz and the simulation ran out to 130ms. A voltage-domain model might easily simulate 10 points per carrier cycle. The voltage-domain model would require 1.3 million points to simulate the same amount of action. I did not attempt it. The phase-domain simulation that generated Figure G-14 ran in a matter of seconds!)

**Figure G-14 Duty Cycle Waveforms With Pulses and Backlash**

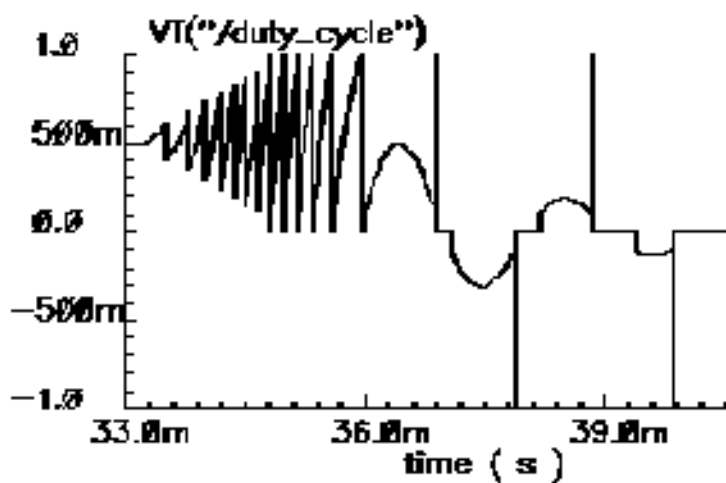


Figure 15a  
Duty cycle between  
33 ms and 40 ms

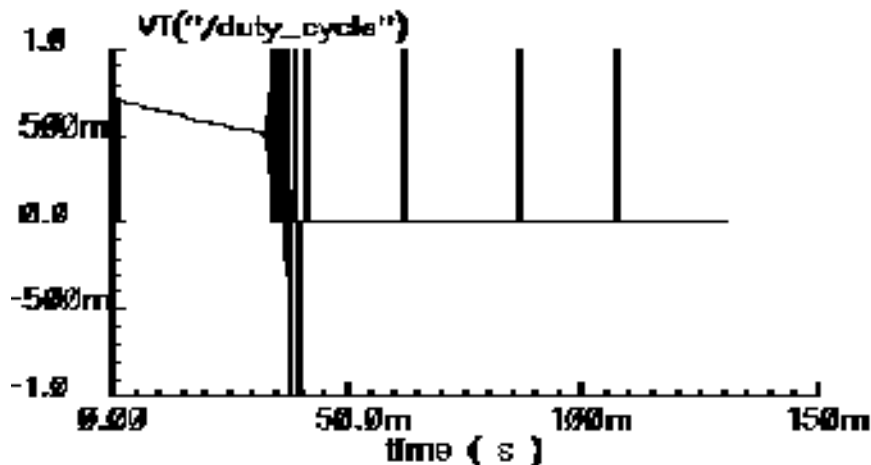


Figure 15b  
Duty cycle to  
130 ms

### Lock Indicator

All real components have limited outputs and the limits of any one component can keep the PLL from locking. Just like the simple phase detector model, all of the phase-domain models operate one way for DC analysis and another for transient analysis to prevent DC convergence errors. The lock indicator monitors three signals. In the example, the lock indicator monitors the phase detector output, the VCO control, and the charge pump output. If any of those signals exceeds its limit, the lock indicator output is zero, signifying that the loop is not locked in steady state. If all signals are within their limits, the output is 1 volt, specifying that the loop is locked. The lock indicator is only valid for DC analysis. Use node names to tie the lock indicator inputs to the right nodes and use variables for the component limits. You must specify the units manually twice, once for each component and once for the lock indicator. With variables, the lock indicator parameters are linked to the proper component parameters, and you specify changes in only one place.

### Example 4: Modeling Acquisition Transients

The circuit used to model acquisition transients is the *example\_phase\_domain* in the *pllLib* library. [Figure G-15](#) on page 974 shows the duty cycle and VCO frequency response to a momentary change in the divider ratio. When the divider ratio changes, the PFD enters the frequency-mode and slews the VCO frequency toward the new value. As the VCO frequency approaches the final value, the PFD model gradually changes from frequency-mode to phase-mode. When the frequency error is small, but still large enough to slew phase error, the duty cycle waveform looks like a sawtooth waveform. The model gradually increases the amplitude of the sawtooth component of the duty cycle, and always maintains the correct average, as frequency error reduces to zero. The final duty cycle transient is the sawtooth that depends mostly on phase error. [Figure G-11](#) on page 967, modifies the x-axis of the graph to show the duty cycle in the first transition. [Figure G-16](#) on page 974 modifies it further to show the sawtooth waveform.

Figure G-15 Response to Momentary Change in Divider Ratio

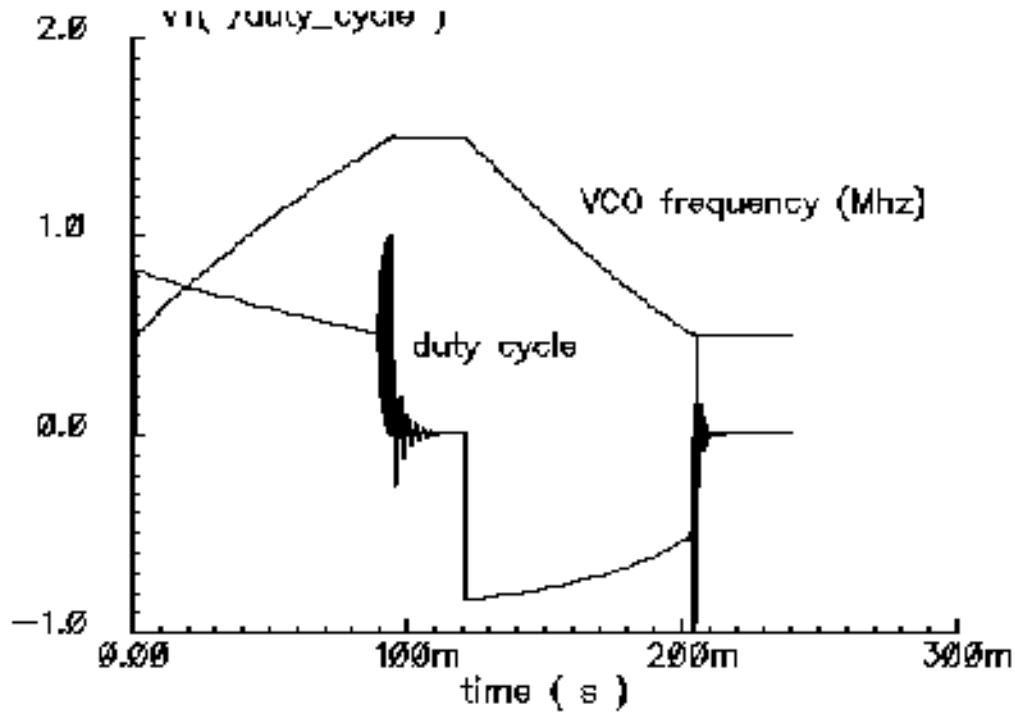
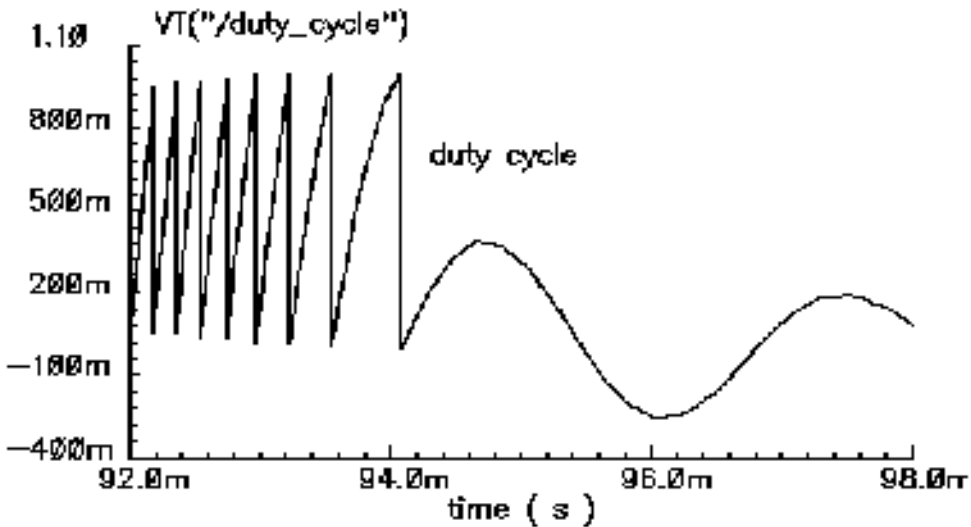


Figure G-16 Duty Cycle During Transition From Frequency to Phase Mode

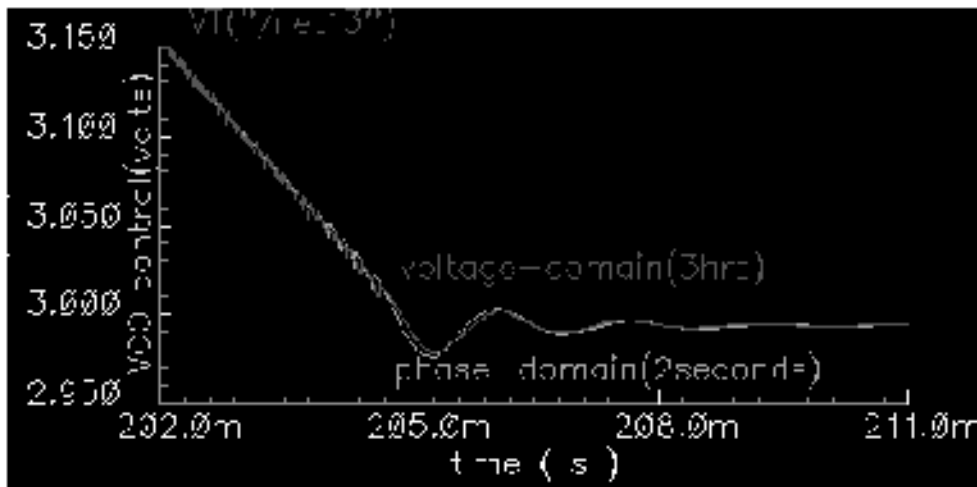
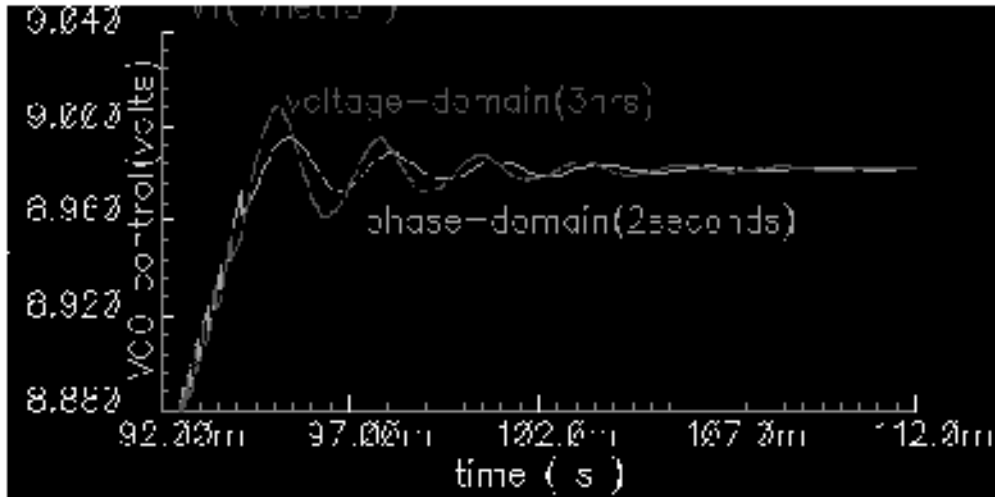


## Example 5: Comparison With a Voltage-Domain Model

Use the *example\_voltage\_domain* model in the *pllLib* library with the only node that can be directly compared between phase- and voltage-domain models, the VCO control node. At full scale, the difference between the two models is not visible. The differences occur at the transitions. [Figure G-17](#) on page 976 compares the two models at the transitions. In this example and on the same machine, the voltage-domain model simulates in three hours while the phase-domain model simulates in two seconds.

The error between the two models does not appear to be consistent. It is larger in the first transition. Furthermore, decreasing the *speed\_vs\_accuracy* parameter does not always increase the similarity of the waveforms. This is because the final transient, the one driven primarily by phase error, depends on the residual frequency error at the time the phase error last crossed  $2\pi$ . The frequency error at that moment, especially after a long frequency slewing period, is sensitive to, among other effects, initial conditions.

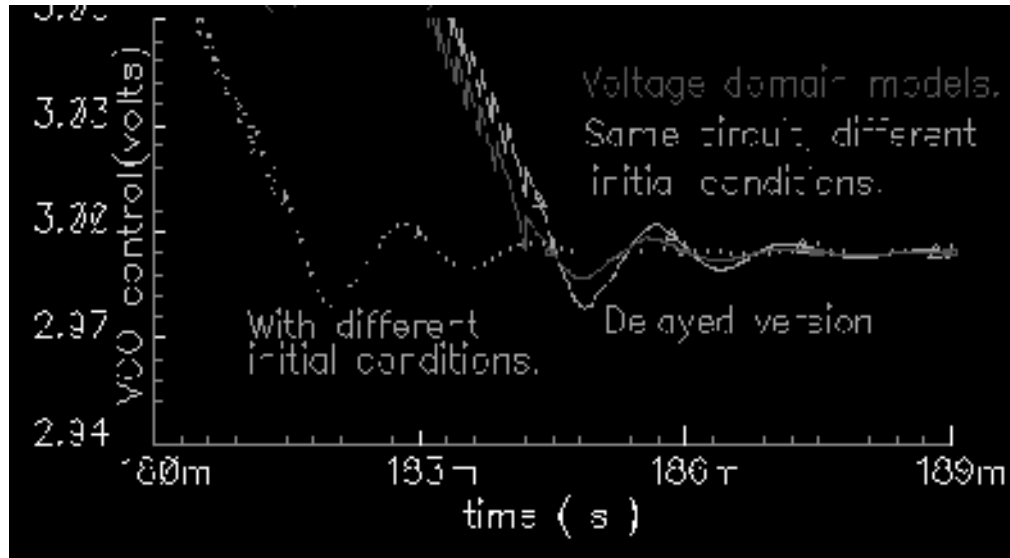
Figure G-17 Comparison with Voltage Domain Model



The voltage-domain model therefore shows the same level of variation for small differences in the initial conditions preceding the transition to the new equilibrium. [Figure G-18](#) on page 977 compares two voltage-domain simulations of the VCO control signal during the second transition. One of the voltage-domain simulations (the dotted waveform) used different initial conditions. The solid waveform is a delayed version of the dotted waveform. The delay overlays the two simulations for direct comparison. The error is comparable to the error between voltage-domain and phase-domain simulations.



Figure G-18 Effect of Initial Conditions



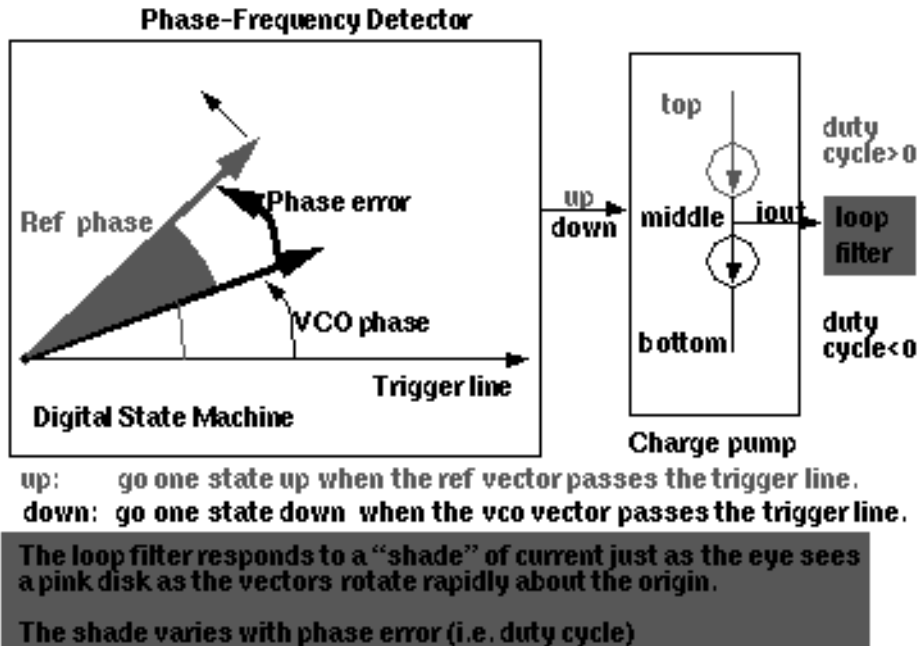
## How the PFD Model Works

The heart of the PFD is a digital state machine [8,9]. The model is for PFDs with three digital states. The output stage is usually a charge pump or pair of switches. It is convenient to model the PFD in two pieces. The first piece models the state machine and computes a duty cycle that is independent of the output stage. The second piece models the output stage. The charge pump (CP) is used here as an example.

## How the PDF/CP Pump Works

Let the three PFD states be stacked. In the top state, the PFD commands the CP to source current. In the middle state, the pump is off. In the bottom state, the CP sinks current. The PFD is edge triggered. [Figure G-19](#) on page 978 shows vectorial representations of the reference and VCO clocks [1,4]. Both vectors rotate counter-clockwise around the origin. The angle between the hands equals phase error. Phase error lies between  $\pm 2\pi$ . Whenever the reference passes a trigger line, like 3 o'clock, the state jumps to the next state up. If the PFD is already in the top state, the state does not change. Whenever the VCO passes the trigger line, the state jumps to the next state down. If it is already in the bottom state, it again does not change. For a fixed phase error, the state toggles as the hands rotate. State toggles between the middle and top, or between the middle and bottom states. The percentage of time spent in the top, or bottom, state is the duty cycle. The duty cycle is positive for top to middle toggling and negative for middle to bottom toggling.

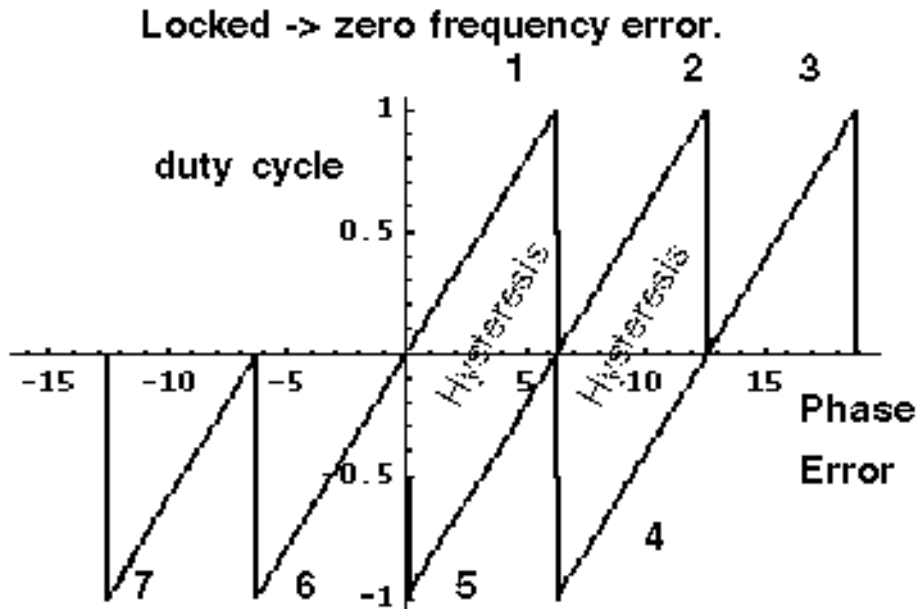
Figure G-19 PFD Operation



If the reference and VCO frequencies are identical, the vectors in Figure G-19 rotate together. Let the sector defined by phase error be red if the reference leads and blue if it lags. If the hands rotate once per minute and the reference leads, you see two colors, white and red. At two million revolutions per second, you see pink. The shade of pink depends linearly on the phase error. Although PFD output current toggles between two values, the loop filter and VCO respond mainly to the “shade” of current. The shade is proportional to the duty cycle. With zero frequency error, duty cycle equals phase error divided by  $2\pi$ . Existing literature uses one function to describe the *phase-error-to-duty-cycle* relationship and a different function to describe the *frequency-error-to-duty-cycle* relationship. These two functions are the *locked duty cycle function* and *averaged unlocked duty cycle function*, respectively. The new model combines these two functions into one practical model.

The locked duty cycle function is a multivalued sawtooth. For monotonic movements away from the origin of steady-state phase error, the duty cycle is a sawtooth in the upper-half plane. The duty cycle lies in the lower-half plane for negative movements. If a movement starts off positive, then changes direction, the duty cycle crosses zero and becomes a sawtooth in the lower half plane. The duty-cycle-phase-error trajectory encloses a nonzero area as shown by the {1-2-3-4-5-6-7} sequence of peaks in Figure G-20 on page 979. This is a good reason for putting the integrator next to the non-linearity—hysteresis involves memory and the integral supplies it.

Figure G-20 Duty Cycle Versus Phase Error With Zero Frequency Error



This can be modeled by a resettable integrator.

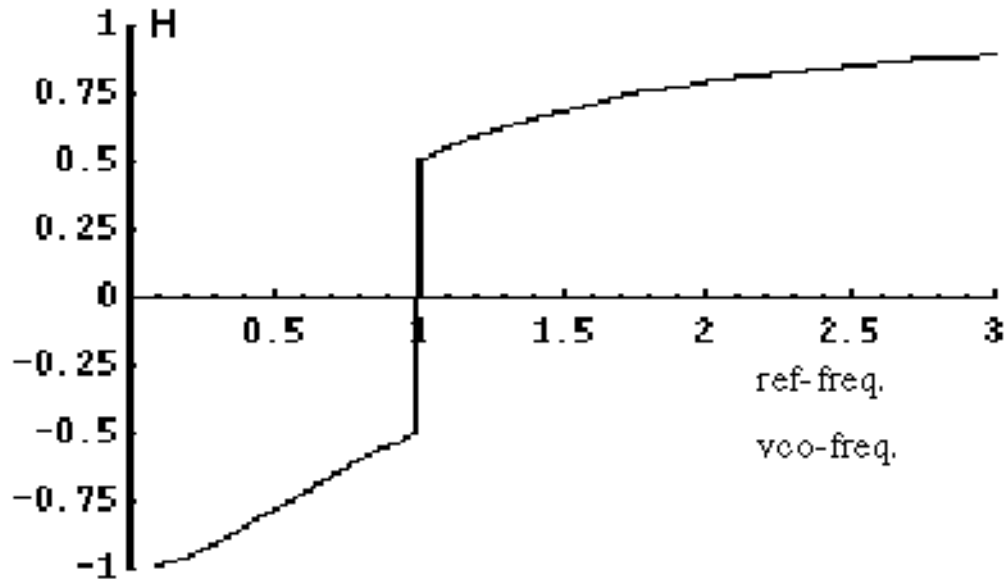
$$\text{duty cycle} = \int_R \frac{(\text{Frequency Error})}{2\pi} dt$$

The new model operates only on frequency error. For small-frequency errors, the duty cycle is indeed proportional to the phase error. The phase error is the integral, with respect to time, of the frequency error. The duty cycle therefore equals the integral of the frequency error divided by  $2\pi$ . Resetting the integral whenever it hits  $\pm 2\pi$  produces the multivalued sawtooth described above. If the frequency error changes sign, the resetting integrator ramps to zero, passes through zero, and generates a sawtooth in the lower-half plane. The phase-error-duty-cycle trajectory is precisely the multivalued sawtooth described above. The resettable integrator (RI) merges the integrator of a phase-domain model with the locked duty-cycle function.

For a sustained frequency error, the RI model predicts an average duty cycle of  $\pm 1/2$  regardless of error size. This is correct only for small-frequency errors. The true duty cycle goes to  $\pm 1$  for large-frequency errors. Let the reference frequency far exceed VCO frequency. Whenever the VCO passes the trigger line, the reference frequency passes shortly thereafter. The reference frequency might pass the trigger line several more times before the VCO passes again. In this case, the phase error is still a sawtooth, but the average duty cycle is nearly 1. This behavior lets the PLL acquire input signals faster. The function H, in

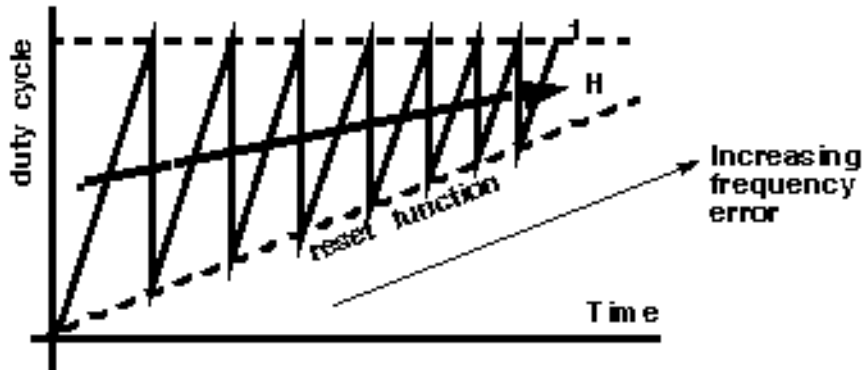
Figure G-21 on page 980, shows the averaged unlocked duty cycle. This cycle depends on the ratio of the reference to VCO frequencies and is discontinuous where frequency error is zero.

Figure G-21 Averaged Unlocked Duty Cycle



The RI is modified to include the frequency effect. For small-frequency errors, the predicted average duty cycle equals 1/2. This is true because the RI runs from the reset point ( $=0$ ) to the reset threshold ( $=2\pi$ ). It is not necessary to reset the integrator to zero. Resetting the integrator to a “reset” function gives the correct average duty cycle (Figure G-22). As the frequency ratio goes to  $+\infty$ , the reset point changes to  $+2\pi$ . Because the reset threshold is still  $+2\pi$ , the predicted average duty cycle changes to  $+1$ . As the frequency ratio approaches unity, the reset point returns to zero, and the predicted average duty cycle returns to 1/2.

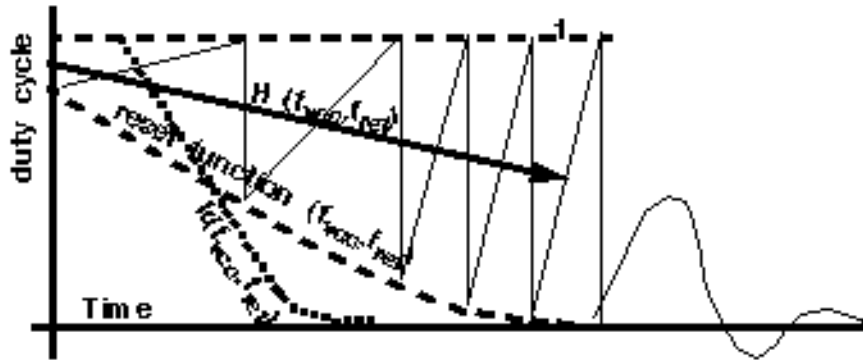
Figure G-22 Combining Averaged Locked and Unlocked Duty Cycles



$$\text{duty cycle} = \int \frac{R(\text{frequency error})}{2\pi} dt$$

The state space averaged PFD model requires one more addition to be practical. As the reset point nears  $\pm 2\pi$ , the integrator resets very frequently and execution stalls. The integrator must be deactivated for large-frequency errors. The new PFD model uses the weighted sum of a RI and the H. The weighting factors are  $k$  and  $(1-k)$ .  $k$  is a function of the ratio of the two input frequencies.  $k$  approaches 1 for large-frequency errors and approaches 0 as the frequency ratio approaches unity. A factor of  $(1-k)$  under the integral deactivates the integral for large-frequency errors. The resulting PFD model looks like H for large-frequency errors, and it looks like the reactivated RI for small-frequency errors.  $k$  determines how fast the RI reactivates and how gradually the model changes from H to the RI. A *speed\_vs\_accuracy* parameter controls  $k$ . If the model does not reset a few times before reaching frequency lock, you can improve the results by decreasing the *speed\_vs\_accuracy* parameter. If the model resets so often that the simulation is too slow, you can speed execution by increasing the parameter. The default setting of 50000 covers a wide range of loop speeds. [Figure G-23](#) on page 982 shows the transition from frequency-mode to phase-mode.

Figure G-23 Complete Model



$$\text{duty cycle} = k \cdot H + (1-k) \int \frac{(1-k) \cdot \text{FrequencyError}}{R \cdot 2\pi} dt$$

## References

[1]“Loops, Theory and Application,” J. L. Stensby.

[2]“Relative -phase modeling speeds Spice simulation of modulated systems,” John Kesterson, November 11, 1993 issue of EDN

[3]“Simulation of Communication Systems,” M. Jeruchim, P. Balaban and K. Shanmugan. Plenum, 1992.

[4]“Synchronization in Digital Communications,” Heinrich Meyr and Gerd Ascheid, Published by John Wiley and Sons.

[5]“Phase-Locked and Frequency-Feedback Systems,” Klapper and Frankle, Published by Academic Press

[6]“Phaselock Techniques,” Floyd Gardner, Published by John Wiley and Sons

[7]“Non-Linear Relative Phase Models of Phaselock Loops,” Jess Chen, Cadence Technical Conference, 1996.

[8]“Phase-Locked Loops: Theory, Design, and Applications,” Roland Best, Published by McGraw-Hill

[9]“Phase-Locked Loop Circuit Design,” Dan Wolaver, Published by Prentice Hall

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Introduction to the PLL library

---

[10]“Phase-Lock Basics,” William F. Egan. Published by John Wiley and Sons.

[11]“Non-Linear State Space Averaged Modeling of a 3-State Digital Phase-Frequency Detector,” Jess Chen, Cadence Technical Conference, 1997

[12] “Analog and Mixed-Signal Hardware Description Languages,” Edited by A. Vachoux, J. Berge, O. Levia, and J. Rouillard. Kluwer Academic Publishers.

[13] “Macromodeling with SPICE,” J.A. Connelly and P. Choi. Prentice Hall. Pages 168-169.

[14] S. Cuk, California Institute of Technology, Ph.D. Thesis, “Modelling, Analysis, and Design of Switching Converters.” 1977

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Introduction to the PLL library

---



---

## Using the Port Component

---

You can use the *port* component, located in the *analogLib* library, in RF circuits for Virtuoso® Spectre® circuit simulator RF Analysis (Spectre RF) and Spectre S-parameter simulations.

The *port* component, located in the *analogLib* library, is similar to the existing *psin* component. The *port* component supports all the *Source types* of the Spectre *port* primitive: *pwl*, *pulse*, *sine*, *dc*, and *exp*.



The *port* component is an independent resistive source tied between positive and negative terminals. It is equivalent to a voltage source in series with a resistor, where the reference resistance of the *port* is the value of the resistor.

### Capabilities of the port Component

While the *port* component is most useful as a stimulus in high-frequency circuits, it also has the following unique capabilities.

It defines the ports of a circuit to the S-parameter analysis

- It has an intrinsic noise source that lets the noise analysis directly compute the noise figure of the circuit
- Is the only source for which you can specify the amplitude in terms of power

## Terminating the port

When you specify the voltage on a *port*, Spectre RF assumes that the port is properly terminated in its reference resistance. The specified voltage value is not the voltage on the internal voltage source, which is actually set to twice the value specified on the `port`. If you use a `port` source to drive an open circuit, the voltage (for DC, transient, AC, and PAC signals) is double its specified value. However, you can alternatively specify the amplitude of the sine wave in the transient and PAC analyses as the power in dBm delivered by the *port* when terminated with the reference resistance.

The *port* component Edit Object Properties form is shown in [Figure H-1](#) on page 986 and [Figure H-2](#) on page 987.

**Figure H-1 Top of the port Component Edit Object Properties Form**

Property	Value	Display
Library Name	analogLib	off
Cell Name	port	off
View Name	symbol	off
Instance Name	PORT	off

User Property	Master Value	Local Value	Display
Ivignore	TRUE		off

CDF Parameter	Value	Display
Resistance	50 Ohms	off

## Parameters for the Port Component

The *port* component's CDF parameters described here are grouped by parameter types, rather than in the order they appear on the *port* Edit Object Properties form.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using the Port Component

Figure H-2 Bottom of the port Component Edit Object Properties Form

CDF Parameter	Value	Display
Resistance	50 Ohms	off
Reactance		off
Port number		off
DC voltage		off
Source type	sine	off
Frequency name 1		off
Frequency 1		off
Amplitude 1 (Vpk)		off
Amplitude 1 (dBm)		off
Phase for Sinusoid 1		off
Sine DC level		off
Delay time		off
Display second sinusoid	<input type="checkbox"/>	off
Display multi sinusoid	<input type="checkbox"/>	off
Display modulation params	<input type="checkbox"/>	off
Display small signal params	<input type="checkbox"/>	off
Display temperature params	<input type="checkbox"/>	off
Display noise parameters	<input type="checkbox"/>	off
Multiplier		off
Number of FM Files	◆ none ◇ one ◇ two	off

### Port parameters

- Resistance
- Reactance
- Port number
- Multiplier
- Number of FM Files

### General waveform parameters

- Source type
- Delay time

## **DC Waveform parameters**

DC voltage

## **Pulse waveform parameters**

Zero value

One value

Period of waveform

Rise time

Fall time

Pulse width

## **PWL waveform parameters**

Waveform Entry Method

File name

Number of PWL/Time pairs

DC offset

Amplitude scale factor

Time scale factor

Breakpoints

Period

Transition Width

## **Sinusoidal waveform parameters**

Sine DC level

Frequency name 1

Frequency 1

Amplitude 1 (Vpk)

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

Amplitude 1 (dBm)

Phase for Sinusoid 1

Frequency name 2

Frequency 2

Amplitude 2 (Vpk)

Amplitude 2 (dBm)

Phase for Sinusoid 2

Sinusoid Frequency 1

Sinusoid Ampl 1 (Vpk)

Sinusoid Ampl 1 (dBm)

Sinusoid Phase 1

Sinusoid Maxharm 1

### **Amplitude and Frequency modulation parameters**

AM modulation index 1

AM modulation frequency 1

AM modulation phase 1

FM modulation index 1

FM modulation frequency 1

Damping factor 1

### **Exponential waveform parameters**

Delay time

Zero value

One value

Rise time start

Rise time constant

Fall time start

Fall time constant

### **Small-signal parameters**

PAC magnitude

PAC magnitude (dBm)

PAC phase

AC magnitude

AC phase

XF magnitude

### **Temperature effect parameters**

Linear temperature coefficient

Quadratic temperature coefficient

Nominal temperature

### **Noise parameters**

Noise Entry Method

Noise file name

Number of noise/freq pairs

### **Port Parameters**

Port parameters include *Resistance*, *Reactance*, *Port Number*, *Multiplier*, and *Number of FM Files*.

#### **Resistance**

The reference resistance of the system. The value must be a real number, but not 0. The default value is 50  $\Omega$  (50 Ohms).

## Reactance

The imaginary part of impedance, used for harmonic balance analyses only. The value must be a real number. Default: 0  $\Omega$

Units: Ohms

## Port number

The number associated with the *port*. The value must be a nonzero integer. Each *port* in a schematic must have a unique *Port number*. The *Port number* is not automatically indexed when you place a new *port* on your schematic.

## Multiplier

The multiplicity factor specifies a number of *ports* in parallel. The value must be a nonzero real number and the default is 1. For example, if you set *Resistance* to 50 and *Multiplier* to 2, you specify two *ports* in parallel, with an effective reference resistance of 25  $\Omega$ .

## Number of FM Files

The number of files that contain the data for frequency modulated waveforms. FM I/Q signals can be written to one file or to two files with the I file first.

## General Waveform Parameters

The General Waveform parameters include *Source type* and *Delay time*.

### Source type

The *Source type* parameter lets you select a wave shape for the port from the *Source type* cyclic field: *dc*, *pulse*, *exp*, *pwl*, *sine*, or blank. Each *Source type* has different parameter settings associated with it. You can define several different wave shapes and quickly switch between them without losing the wave shape settings. The wave shape settings are described in detail in the *DC*, *Pulse*, *Piecewise Linear*, *Sinusoidal*, and *Exponential Waveform Parameters* sections.

The typical *Source types* used in Spectre RF analyses are *dc*, *pulse*, and *sine*. For example, you can quickly switch from a sinusoid level (for PSS analysis) to a DC level (for PAC analysis) by changing the *Source type* from *sine* to *dc*.

When you set *Source type* to the blank value, the *port* acts as a resistive load.

### **Delay time**

The *Delay time* is the amount of time that the source stays at the DC level before it starts generating waveforms (assuming the *Source type* is set to *sine*, *pulse*, *pwl*, or *exp*). The value must be a real number. The default value is 0 and the units are seconds.

### **DC Waveform Parameters**

To generate a dc waveform from the *port* component, select *dc* in the *Source type* cyclic field, as illustrated in [Figure H-3](#) on page 993.

When the *Source type* is set to *dc*, the *dc* and *temperature effect* parameters are active.

The *dc* setting sets the DC level for all analyses. In DC analysis, this setting also determines the DC level generated by the source, regardless of what *Source type* you specify.

### **DC voltage**

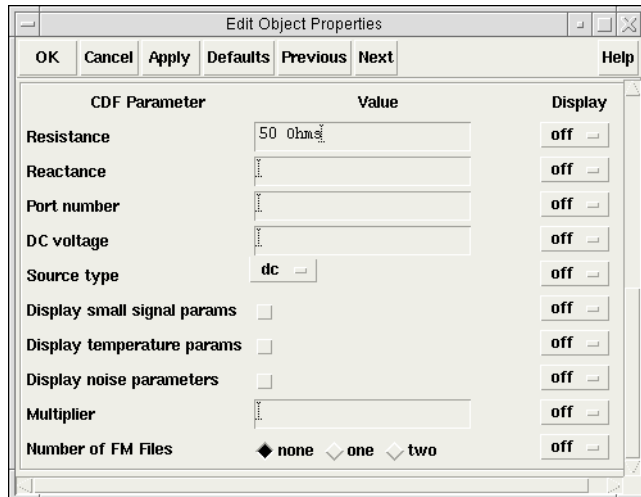
The *DC voltage* parameter sets the port's DC level for DC analysis. The value must be a real number. If you do not specify the DC value, it is assumed to be the *time=0* value of the waveform. The default value is 0 and the units are Voltage.

The *DC voltage* parameter specifies the DC voltage across the *port* when it is terminated in its reference resistance. In other words, the *DC voltage* of the internal voltage source is double the specified DC value, *dc*. The same is true for the values for the *transient*, *AC*, and *PAC* signals from the `port`. However, you can alternatively specify the amplitude of the sine wave in the transient and PAC analyses as the power delivered in dBm by the *port* when it is terminated with the reference resistance.

Because all small signal analyses (AC, XF, and Noise) use DC analysis results, the *DC voltage* level also affects the small-signal analyses. Transient analysis is not affected unless you specify *Source type=dc* or use *dc* as a default for the other waveform types.



Figure H-3 Source type=dc in the Edit Object Properties form



The *Display small signal params*, *Display temperature params*, and *Display noise parameters* fields are discussed in [“Small-Signal Parameters”](#) on page 1011, [“Temperature Effect Parameters”](#) on page 1013, and [“Noise Parameters”](#) on page 1009.

## Pulse Waveform Parameters

To generate a pulse waveform from the *port* component, select *pulse* in the *Source type* cyclic field.

When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Therefore, the voltage on the internal source is set to twice the value specified on the *port*.

Figure H-4 Source type=pulse in the Edit Object Properties form

CDF Parameter	Value	Display
Resistance	50 Ohms	off
Reactance		off
Port number		off
DC voltage		off
Source type	pulse	off
Frequency name 1		off
Delay time		off
Zero value		off
One value		off
Period of waveform		off
Rise time		off
Fall time		off
Pulse width		off
Display small signal params	<input type="checkbox"/>	off
Display temperature params	<input type="checkbox"/>	off
Display noise parameters	<input type="checkbox"/>	off
Multiplier		off
Number of FM Files	<input checked="" type="radio"/> none <input type="radio"/> one <input type="radio"/> two	off

### Frequency name 1

The *Frequency name 1* parameter is described in [“Frequency name 1”](#) on page 999.

### Delay time

The *Delay time* parameter is described in [“Delay time”](#) on page 992

### Zero value

The *Zero value* parameter (*val0*) is used with the *pulse* and *exp* waveforms. The default value is 0 and the units are Voltage.

### One value

The *One value* parameter (*val1*) is used with the *pulse* and *exp* waveforms. The default value is 1 and the units are Voltage.

### Period of waveform

The *period* parameter of the *pulse* waveform. The default value is infinity and the units are seconds.

### Rise time

The *Rise time* parameter of the *pulse* waveform is the time for the transition from the *Zero value* to the *One value*. The units are seconds.

### Fall time

The *Fall time* parameter of the *pulse* waveform is the time for the transition from the *One value* to the *Zero value*. The units are seconds.

### Pulse width

The *Pulse width* parameter of the *pulse* waveform is the width, or duration of the *One value*. The default value is infinity and the units are seconds.

## PWL Waveform Parameters

To generate a piecewise linear waveform from the *port* component, select *pwl* in the *Source type* cyclic field. This sets the *Source type* CDF parameter to *pwl* and displays additional fields for the PWL CDF parameter settings.

When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Therefore, the voltage on the internal source is set to twice the value specified on the *port*.

Figure H-5 Source type=pwl in the Edit Object Properties form

CDF Parameter	Value	Display
Resistance	50 Ohms	off
Reactance		off
Port number		off
DC voltage		off
Source type	pwl	off
Frequency name 1		off
Waveform Entry Method	File	off
File name		off
Delay time		off
DC offset		off
Amplitude scale factor		off
Time scale factor		off
Breakpoints		off
Period		off
Transition width		off
Power of PWL waveform	dBm	off
Display small signal params	<input type="checkbox"/>	off
Display temperature params	<input type="checkbox"/>	off
Display noise parameters	<input type="checkbox"/>	off
Multiplier		off
Number of FM Files	none	off

## Waveform Entry Method

With the *Waveform Entry Method* buttons, select how you enter piecewise-linear data,

- By specifying a *File name*.
- By entering a series of *Voltage/Time points*.

### File name

When you select *File* as the *Waveform Entry Method*, type the name of the file containing your piecewise-linear data in the *File name* field,. The *File name* must be a string. There is no default.

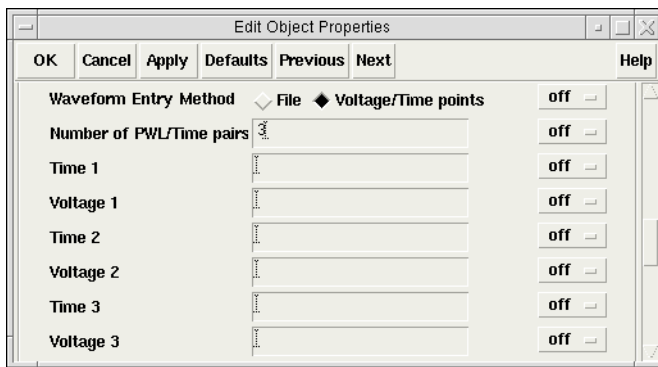
In your file, list the piecewise-linear data in the form of time-value pairs. Enter one pair per line with a space or tab between the time and voltage values. The numbers in the file must be simple numbers. You cannot use SI scale factors (p, n, u, m, k, M, G, etc.).

### Number of PWL/Time pairs

When you select *Voltage/Time points* as the *Waveform Entry Method*, the *Number of PWL-Time pairs* field (tvpairs) opens. Enter the number of time-value pairs you plan to enter. The form expands to let you enter the designated number of *Time* and *Voltage* values. Units are seconds and volts. The default is 0 and the maximum value is 50.

In this example, the number of voltage-time pairs is 3.

**Figure H-6 Waveform Entry Method=Voltage/Time points**



The *DC offset*, *Amplitude scale factor*, and *Time scale factor* parameter fields let you quickly adjust the amplitude, frequency, and offset of your piecewise-linear data pairs without editing each individual time-value pair in the *pwl* waveform.

### ***DC offset***

*DC offset (offset)* for the *pwl* waveform. Default: 0 Units: V

### ***Amplitude scale factor***

*Amplitude scale factor (scale)* for the *pwl* waveform. Default: 1

### ***Time scale factor***

*Time scale factor (stretch)* for the time given for the *pwl* waveform. Default: 1

### ***Breakpoints***

Possible values are *no*, *yes*, or blank. If you set *Breakpoints* to *yes*, you force Spectre RF to place time points at each point specified in a *pwl* waveform during a transient analysis. This can be very expensive for waveforms with many points. If you set *Breakpoints* to *no*, Spectre RF inspects the waveform, looking for abrupt changes, and forces time points only at those changes. If you set *Source type = pwl* and set *Breakpoints* to blank, the default is *yes* if the number of points you specify is less than 20.

### ***Period***

The *pwl* waveform is periodic if you specify *Period (pwlperiod)*. Units: seconds

If the value of the waveform you specify is not exactly the same at both its beginning and its end, then you must provide a nonzero value for *Transition Width*.

### ***Transition Width***

*Transition width (twidth)* is used when making *pwl* waveforms periodic. Default:  $PWL\ period/1000$ . Units: seconds

Before repeating, the waveform changes linearly in an interval of *Transition Width* from its value at  $(Period - Transition\ Width)$  to its value at the beginning of the waveform. Thus the *Transition Width* must always be less than the *Period*.

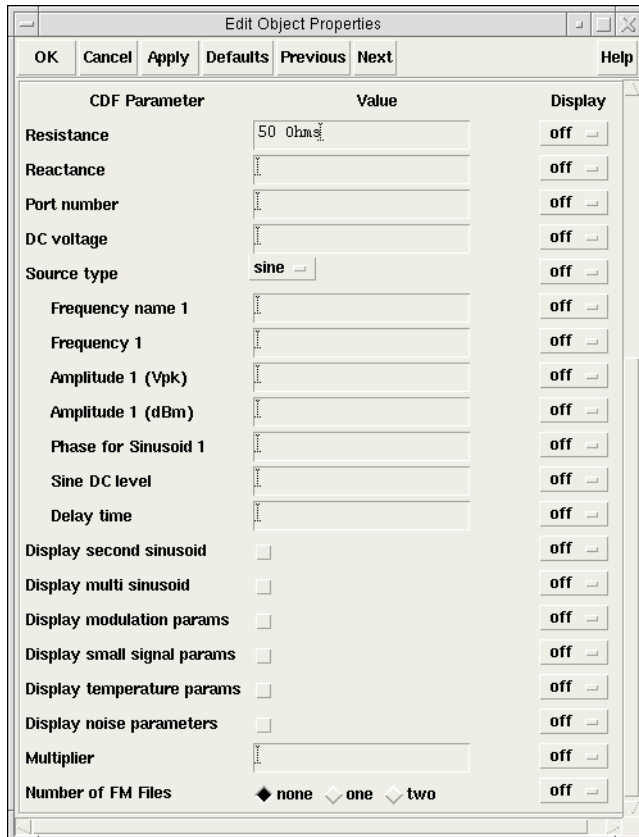
## **Sinusoidal Waveform Parameters**

The *port* component can generate up to two sinusoids simultaneously. They are denoted as 1 and 2. You can set the amplitude, frequency, and phase for both individually. The amplitude can be set to either a voltage or a power level. You can also specify sinusoidal AM or FM modulation of sinusoid 1.

The Edit Object Properties form for *Source type=sine* is shown in [Figure H-7](#) on page 999.

The first sinusoid is described by the parameters *Frequency name 1*, *Frequency 1*, *Amplitude 1 (Vpk)*, *Amplitude 1 (dBm)*, *Phase for Sinusoid 1*, *Sine DC level*, *Damping factor 1*, and by AM or FM modulation terms.

Figure H-7 Source type=sine in the Edit Object Properties form



### Frequency name 1

Names the fundamental tones of sinusoid 1. After you save the schematic, the names you assign appear in the *Fundamental Tones* list box on the Choosing Analyses form.

*Frequency 1* is the frequency of the first sinusoidal waveform (carrier frequency). You typically use unmodulated signals in Spectre RF analyses. The value must be a real number. Default: 0 Units: Hz

### Amplitude 1 (Vpk)

The peak amplitude of the first sinusoidal waveform that you generate. The value specified is the voltage delivered into a matched load. You can select either *Amplitude 1 (Vpk)* or *Amplitude 1 (dBm)*, but not both. If *Amplitude 1 (Vpk)* has a value, the *Amplitude 1 (dBm)* field is grayed out. The value must be a real number. Default: 1 Units: V

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *port*.

#### Amplitude 1 (dBm)

*Amplitude 1 (dBm)* is the amplitude of the first sinusoidal waveform, in dBm. The value specified is the power delivered into a matched load. You can select either *Amplitude 1 (Vpk)* or *Amplitude 1 (dBm)*, but not both. If *Amplitude 1 (dBm)* has a value, the *Amplitude 1 (Vpk)* field is grayed out. The value must be a real number. Units: dBm

#### Phase for Sinusoid 1

The phase at the specified *Delay time*. To achieve a specified phase and still remain continuous, the sinusoidal waveform might start before the given *Delay time*. For example, if you want to generate a cosine wave, set this parameter to 90°. The value must be a real number. Default: 0 Units: degrees

#### Sine DC level

Sets the DC level for sinusoidal waveforms in transient analyses. This parameter is used when the sinusoid has a different average level than the one specified for the DC analyses. If not specified, the average value of the sinusoid is the same as that of the DC level of the source. The value must be a real number. Default: dc Units: V

### Modulation Parameters

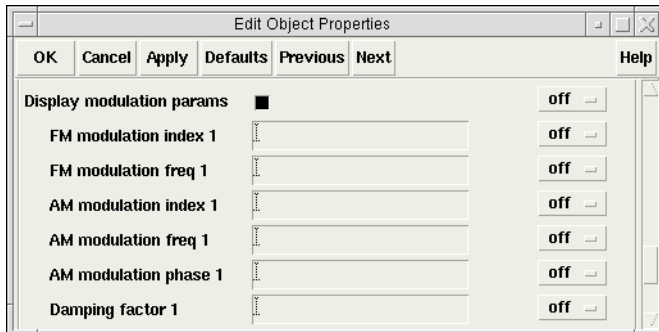
#### Display Modulation Parameters

When selected, the form expands and the following modulation parameters are displayed: *FM modulation index 1*, *FM modulation freq 1*, *AM modulation index 1*, *AM modulation freq 1*, *AM modulation phase 1*, and *Damping factor 1*.

Only the first sinusoid can be modulated.



**Figure H-8 Display modulation params**



### FM Modulation (Background Information)

The frequency modulation for the sinusoidal case is defined as

$$v_{FM}(t) = A \sin(2\pi f_c t + \beta \sin(2\pi f_m t) + \phi)$$

where

- $A$  is the amplitude of sinusoid 1
- $\beta$  is the FM modulation index
- $\sin(2\pi f_m t)$  is the modulation signal
- $f_c$  is the carrier frequency
- $\phi$  is the *phase for sinusoid 1*

The frequency modulation parameters affect only the first sinusoid generated by *port*. They have no effect on the second sinusoid.

### FM modulation frequency 1

FM modulation frequency for the sinusoidal waveform ( $f_m$  in the previous equation). The value must be a real number. Default: 0 Units: Hz

### FM modulation index 1

FM index of modulation for the sinusoidal waveform, the ratio of peak frequency deviation divided by the center frequency ( $\beta$  in the above equations).

$$\beta = \Delta f / f_m$$

The value must be a real number. Default: 0

## Effect of Amplitude Modulation (Background Information)

The amplitude modulation (double sideband suppressed carrier, or DSB-SC) is defined as

$$v_{AM}(t) = A (1 + m \sin(2\pi f_m t + \phi)) \sin(2\pi f_c t)$$

where

- $A$  is the carrier amplitude (amplitude of sinusoid 1)
- $m$  is the AM modulation index
- $f_m$  is the AM modulation frequency
- $\phi$  is the AM modulation phase
- $\sin(2\pi f_c t)$  is the carrier signal

The amplitude modulation parameters affect only the first sinusoid generated by *port*. They have no effect on the second sinusoid.

### AM modulation frequency 1

AM modulation frequency for the first sinusoidal waveform ( $f_m$  in the previous equation). The value must be a real number. Default: 0 Units: Hz

### AM modulation phase 1

AM phase of modulation for the first sinusoidal waveform ( $\phi$  in the previous equation). The value must be a real number. Default: 0 Units: degrees

### AM modulation index 1

AM index of modulation for the first sinusoidal waveform ( $m$  in the previous AM equation). The *AM modulation index 1* is a dimensionless scale factor used to control the ratio of the sidebands to the carrier.

$$m = (\text{peak\_DSB-SC\_amplitude}) / (\text{peak\_carrier\_amplitude})$$

The value must be a real number. Default: 0

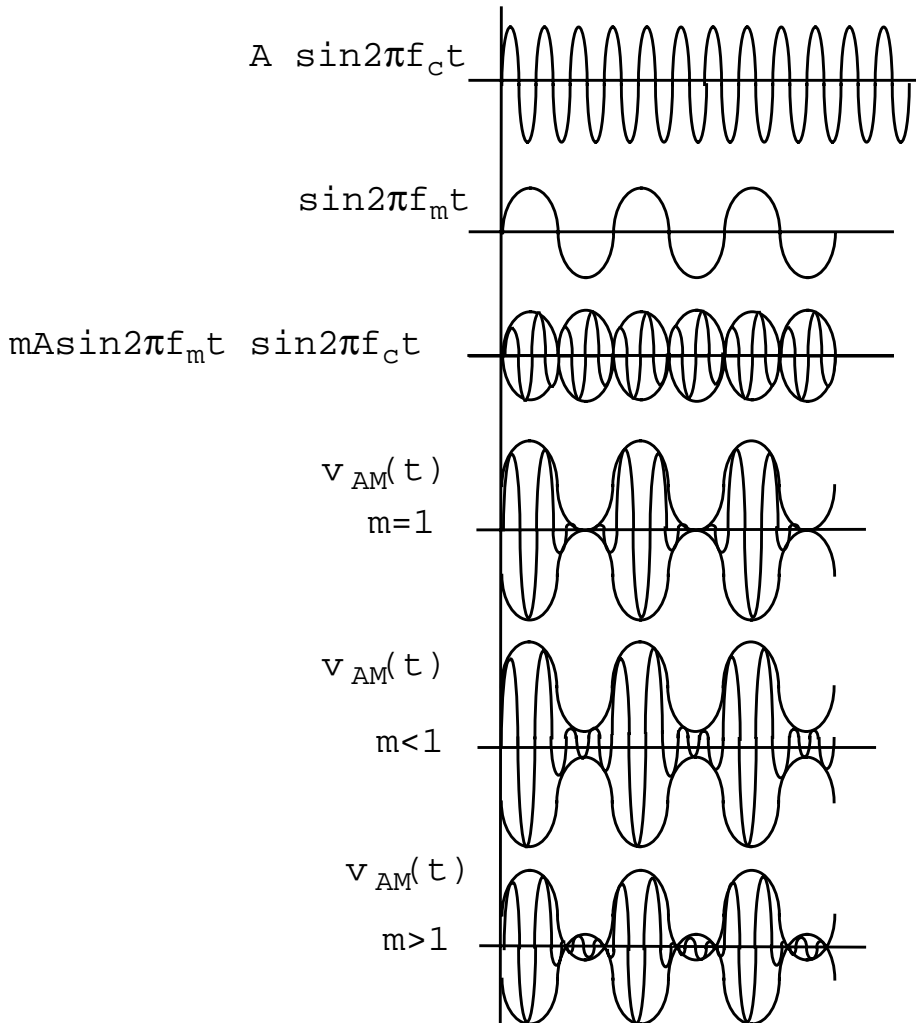
## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

The following figure shows the effect of varying modulation indexes for the following three cases:  $m < 1$ ,  $m = 1$ , and  $m > 1$ .  $f_c$  is the carrier frequency, and  $f_m$  is the modulation frequency.

**Figure H-9 Amplitude Modulation: Effects of Varying Modulation Indexes**



### Damping factor 1

Damping factor for the sinusoidal waveform. *Damping factor 1* specifies the time it takes to go from the envelope (full amplitude) at  $time=0$  to 63 percent of the full amplitude. For example, consider the following damped sinusoid:

$$v(t) = A e^{-\sigma t} \sin(2\pi f t + \phi)$$

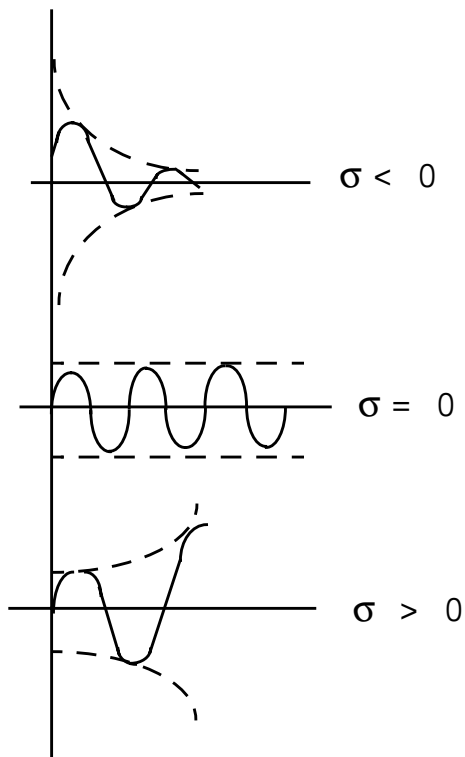
where

$\sigma$  = *Damping factor 1*,  $A$  is the amplitude of sinusoid 1, and  $\phi$  is the *Phase for sinusoid 1*

- If  $\sigma = 0$ , the waveform is a pure sinusoid (steady state).
- If  $\sigma < 0$ , the waveform exhibits decaying oscillations.
- If  $\sigma > 0$ , the waveform exhibits growing oscillations.
- It takes  $5\sigma$  to diminish to 1 percent of the peak amplitude. The value must be a real number. Default: 0. Units: 1/seconds

The following figure shows the effect of *Damping factor 1* on the first sinusoid for three values of  $\sigma$ .

**Figure H-10 Effect of Damping Factor 1 on the First Sinusoid**



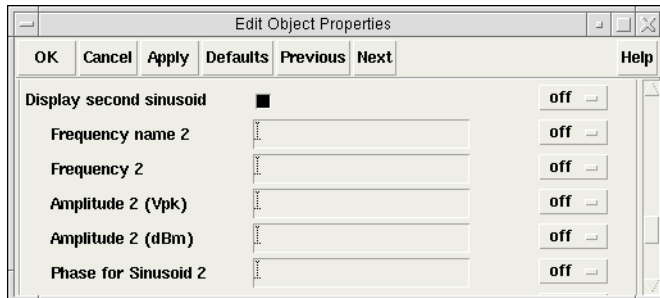
### Display second sinusoid

Displays the CDF parameters for the second sinusoid in the Edit Object Properties and Add Instance forms. When selected, the form expands to show the following CDF parameters:

*Frequency name 2, Frequency 2, Amplitude 2 (Vpk), Amplitude 2 (dBm), and Phase for Sinusoid 2.*

**Note:** The second sinusoid cannot be modulated.

**Figure H-11 Display second sinusoid**



### Frequency name 2

Name for the second sinusoid. After you save the schematic, the name you assign appears in the *Fundamental Tones* list box on the Choosing Analyses form.

*Frequency 2* is the frequency of the second sinusoidal waveform. The value must be a real number. Default: 0 Units: Hz

### Amplitude 2 (Vpk)

Peak amplitude of the second sinusoidal waveform. The value specified is the voltage delivered into a matched load. You can select either *Amplitude 2 (Vpk)* or *Amplitude 2 (dBm)*, but not both. If *Amplitude 2 (Vpk)* has a value, the *Amplitude 2 (dBm)* field is grayed out. The value must be a real number. Default: 1 Units: V

When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *port*.

### Amplitude 2 (dBm)

*Amplitude 2 (dBm)* is the amplitude of the second sinusoidal waveform, in dBm. The value specified is the power delivered into a matched load. You can select either *Amplitude 2 (Vpk)* or *Amplitude 2 (dBm)*, but not both. If *Amplitude 2 (dBm)* has a value, the *Amplitude 2 (Vpk)* field is grayed out. The value must be a real number. Units: dBm

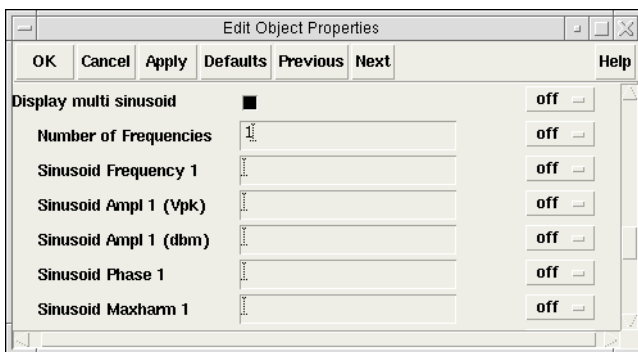
## Phase for Sinusoid 2

The phase at the specified *Delay time* for the second sinusoid. To achieve specified phase while still remaining continuous, the sinusoidal waveform might start before the given *Delay time*. The value must be a real number. Default: 0 Units: degrees

## Display multi sinusoid

Displays the CDF parameters for multiple sinusoids in the Edit Object Properties form. When selected, the form expands to show the *Number of Frequencies* field and for each one of the frequencies, a set of parameters patterned after: *Sinusoid Frequency 1*, *Sinusoid Ampl 1 (Vpk)*, *Sinusoid Ampl 1 (dBm)*, *Sinusoid Phase 1*, *Sinusoid Maxharm 1*.

Figure H-12 Display multi sinusoid



## Number of Frequencies

Number of sinusoid frequencies to be specified.

## Sinusoid Frequency 1

The frequency of the first sinusoidal waveform (carrier frequency). The value must be a real number. Default: 0 Units: Hz

## Sinusoid Ampl 1 (Vpk)

The peak amplitude of the sinusoidal waveform that you generate. The value must be a real number. Default:1 Units: V

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *port*.

Use either *Sinusoid Ampl 1 (Vpk)* or *Sinusoid Ampl 1 (dBm)* but do not set both.

#### Sinusoid Ampl 1 (dBm)

The amplitude of the first sinusoidal waveform when specified in dBm. The value must be a real number. Units: dBm.

Use either *Sinusoid Ampl 1 (Vpk)* or *Sinusoid Ampl 1 (dBm)* but do not set both.

#### Sinusoid Phase 1

The phase at the specified delay time. To achieve a specified phase and still remain continuous, the sinusoidal waveform might start before the given delay time. For example, to generate a cosine wave, set this parameter to  $90^\circ$ . The value must be a real number. Default: 0 Units: degrees.

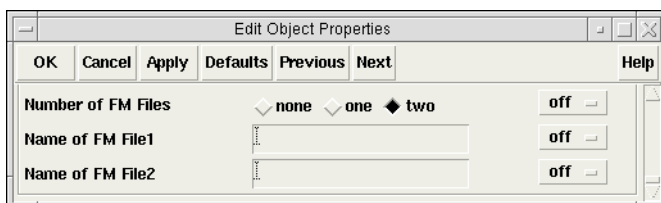
#### Sinusoid Maxharm 1

An array of the number of harmonics of each fundamental to consider for each fundamental.

#### Number of FM Files

Specifies the number of FM files used to hold FM waveform I/Q signals (none, one, or two) and expands the form to show the following parameters: *Name of FM File 1*, and *Name of FM File 2*.

**Figure H-13 Number of FM Files with Two Selected.**



### Name of FM File1

Name of an FM file containing data for frequency modulated waveforms for a sinusoidal source.

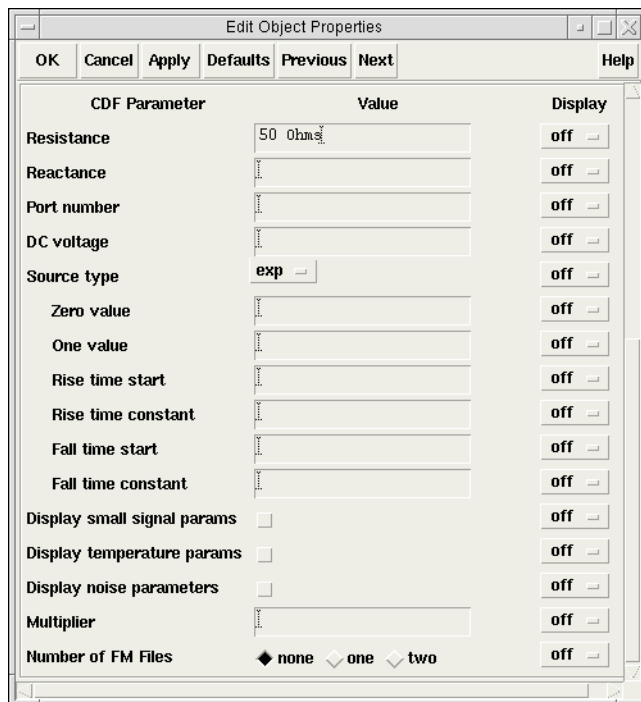
### Name of FM File2

Name of a second FM file containing data for frequency modulated waveforms for a sinusoidal source.

## Exponential Waveform Parameters

To generate an exponential waveform from the `port` component, set the CDF parameter *Source type=exp*, as shown in the next figure.

**Figure H-14 Source type=exp in the Edit Object Properties form**



When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *port*.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

#### Zero value

The *Zero value* (*val0*) used in *pulse* and *exp* waveforms. Default: 0 Units: V

#### One value

The *One value* (*val1*) used in *pulse* and *exp* waveforms. Default: 1 Units: V

#### Rise time start

The *Rise time start* (*td1*) for the *exp* waveform. Default: 0 Units: seconds

#### Rise time constant

The *Rise time constant* (*tau1*) for the *exp* waveform. Units: seconds

#### Fall time start

The *Fall time start* (*td2*) for the *exp* waveform. Units: seconds

#### Fall time constant

The *Fall time constant* (*tau2*) for the *exp* waveform. Units: seconds

### Noise Parameters

The noise parameters include *Noise temperature*, *Noise Entry Method*, *Noise file name*, and *Number of Noise Frequency Pairs*.

#### Noise temperature

The *Noise temperature* of the *port*. If not specified, the *Noise temperature* is assumed to be the actual temperature of the *port*. When you compute the noise figure of a circuit driven at its input by a *port*, set the *Noise temperature* of the *port* (Spectre parameter *noisetemp*) to 16.85C (290K). This setting matches the standard IEEE definition of noise figure. In addition, disable all other sources of noise in the *port*, such as the Spectre parameters *noisefile* and *noisevec*. If you want a noiseless *port*, set the *Noise temperature* to absolute zero or below, and do not specify a noise file or noise vector. Default: Actual temperature of the port. Units: °C

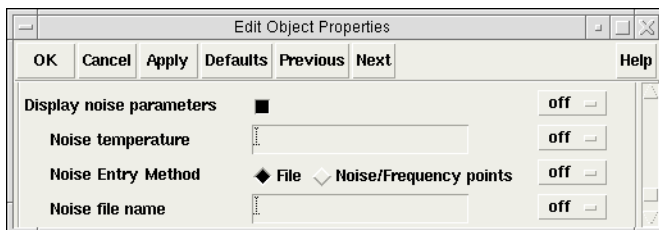
## Noise Entry Method

You can select one of two ways to enter noise data, either by specifying a *File* name or entering a series of *Noise/Frequency points*.

### Noise file name

If *Noise Entry Method = File* is selected, you enter the name of the file containing the excess spot noise data in the form of frequency-noise pairs. In your file, list the frequency-noise pairs as one pair per line with a space or tab between the frequency and noise values. The value must be a string. Default: none

**Figure H-15 Display noise parameters: Noise Entry Method=File**

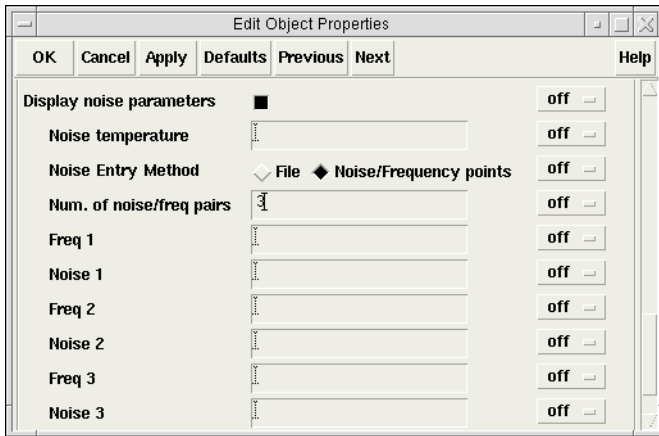


### Num. of noise/freq pairs

If the *Noise Entry Method=Noise/Frequency points*, you specify the *Number of noise/freq pairs*. The form expands to let you type in the designated *Freq* and *Noise* values. The noise values must be in  $V^2/Hz$ , and frequency in Hz. Default: 0 Maximum value: 10

The example in the next figure has the *Number of noise/freq pairs* set to 3.

Figure H-16 Display noise parameters: Noise Entry Method=Noise/Frequency Points

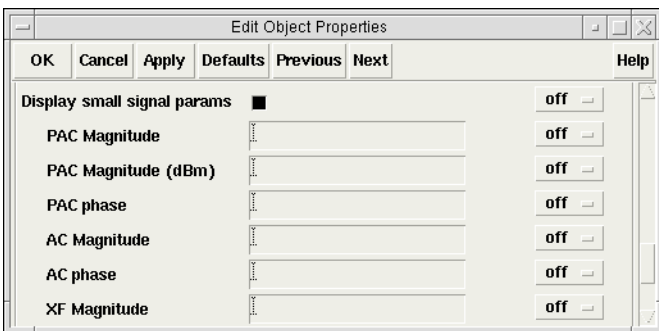


## Small-Signal Parameters

### Display small signal params

If selected, the Edit Object Properties/Add Instance form expands to show the small-signal parameters *PAC Magnitude*, *PAC Magnitude (dBm)*, *PAC phase*, *AC Magnitude*, *AC phase*, and *XF Magnitude*.

Figure H-17 Display small signal params



When you specify the voltage on a *port*, you are specifying the voltage when the port is properly terminated, and not the voltage on the internal voltage source. Thus, the voltage on the internal source is set to twice the value specified on the *port*. The same is true for the values for the transient, AC, and PAC signals. However, the amplitude of the sine wave in the PAC and transient analysis can alternatively be specified as the power in dBm delivered by the *port* when terminated with the reference resistance.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

#### **PAC Magnitude**

The peak periodic AC analysis magnitude. Setting this value to unity is a convenient way of computing the transfer function from this source to the output.

You can select either *PAC magnitude* or *PAC magnitude (dBm)*, but not both. If *PAC magnitude* has a value, the *PAC magnitude (dBm)* field is grayed out. The value must be a real number. Default: 0 Units V

#### **PAC Magnitude (dBm)**

The periodic AC analysis magnitude in dBm (alternative to *PAC magnitude*). You can select either *PAC magnitude* or *PAC magnitude (dBm)*, but not both. If *PAC magnitude (dBm)* has a value, the *PAC magnitude* field is grayed out. The value must be a real number. Units: dBm

#### **PAC phase**

The periodic AC analysis phase. The value must be a real number. Default: 0 Units: degrees

Typically, only one source in the circuit has a *PAC magnitude* set to a value other than zero, and usually it has a *PAC magnitude*=1 and *PAC phase*=0. However, there are situations where more than one source has a nonzero *PAC magnitude*. For example, applying a differential small-signal input could be done with two sources with the *PAC magnitudes* set to 0.5 and the *PAC phases* set to 0 and 180.

You do not specify the PAC frequency in the *port* Edit Object Properties form. Instead, you set the PAC frequency in the *PAC Choosing Analyses form*. For example, when making an IP3 measurement, you set the PAC frequency to a variable value in the Choosing Analyses Form. Then, you enter the same variable in the *PAC Amplitude* (or *PAC Amplitude dBm*) field of the *port* Edit Object Properties form.

#### **AC Magnitude**

The peak small-signal voltage. The value must be a real number. Default: 0 Units: V

#### **AC phase**

The small-signal phase. The value must be a real number. Default: 0 Units: degrees

Typically, only one source in the circuit has *AC Magnitude* set to a value other than zero, and usually it has an *AC magnitude*=1 and *AC phase*=0. However, there are situations where

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using the Port Component

---

more than one source has a nonzero *AC magnitude*. For example, you can apply a differential small-signal input with two sources with the *AC magnitudes* set to 0.5 and the *AC phases* set to 0 and 180.

### XF Magnitude

The transfer function analysis magnitude. Use *XF magnitude* to compensate for gain or loss in the test fixture. The value must be a real number. Default: 1 Units: V/V

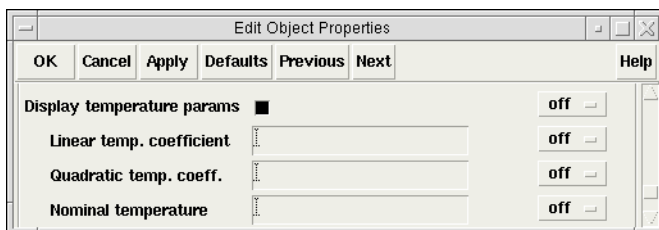
### Temperature Effect Parameters

Temperature effect parameters include the *Linear temperature coefficient*, *Quadratic temperature coefficient*, and *Nominal temperature*.

### Display temperature params

If selected, the Edit Object Properties/Add Instance form expands and the following three parameters appear in the form: *Linear temp. coefficient*, *Quadratic temp. coeff.*, and *Nominal temperature*.

**Figure H-18 Display temperature params**



### Linear temp. coefficient

First order (linear) temperature coefficient of the *DC voltage* ( $tc1$ ). The value must be a real number. Default: 0 Units:  $^{\circ}\text{C}^{-1}$

### Quadratic temp. coeff.

Second order (quadratic) temperature coefficient of the *DC voltage* ( $tc2$ ). The value must be a real number. Default: 0 Units:  $^{\circ}\text{C}^{-2}$

## Nominal temperature

The *Nominal temperature* for *DC voltage* (*tnom*). The value must be a real number.  
Default: Set by options specifications. Units: °C

## How Temperature Parameters Affect the Voltage Level (Background Information)

The value of the *DC voltage* can vary as a function of the temperature if you specify *tc1* and *tc2*. The variation is given by

$$V_{DC}(T) = dc * [1 + tc1 * (T - tnom) + tc2 * (T - tnom)^2]$$

where *T* is the analysis temperature specified in the analysis options, *tnom* is the *Nominal temperature* specified in the *Choosing Analyses* form, and *dc* is the *DC voltage*.

If the analysis temperature equals the nominal temperature, the result is the voltage amplitude that you specified,  $V(T)=dc$ .

If the nominal and analysis temperatures differ, the voltage amplitude is given by

$$V_{DC}(T) = dc * [1 + tc1 * (T - tnom) + tc2 * (T - tnom)^2]$$

where *T* is the analysis temperature you specify in the analysis options and *tnom* is the nominal temperature, *tc1* and *tc2* are the *Linear* and *Quadratic temperature coefficients*, and *dc* is the *DC voltage*.

For example, if the *Nominal temperature* is 27°C and the analysis temperature is 25°C, there is a 2° difference between the nominal and analysis temperature. The voltage amplitude is

$$V_{DC}(T) = dc * [1 + tc1 * (-2) + tc2 * (-2)^2]$$

## Additional Notes

### Active Parameters in Analyses

In DC analyses, the only active parameters are *dc*, *m*, and the temperature coefficient parameters.

In AC analyses, the only active parameters are *m*, *mag*, and *phase*.

In transient analyses, all parameters are active except the small-signal parameters and the noise parameters.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using the Port Component

---

In PAC, the only active parameters are  $m$ , *PAC magnitude* (amplitude or dBm), and *PAC phase*.

*XF magnitude* is active in XF and PXF analyses only.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using the Port Component

---



---

## Analyzing Time-Varying Noise

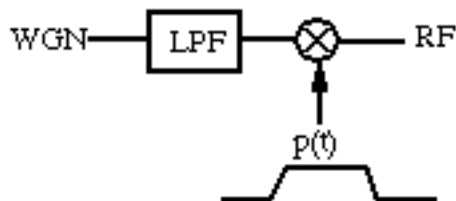
---

RF circuits are usually driven by periodic inputs. The noise in RF circuits is generated by sources that can therefore typically be modeled as periodically time-varying. Noise that has periodically time-varying properties is said to be cyclostationary.

### Characterizing Time-Domain Noise

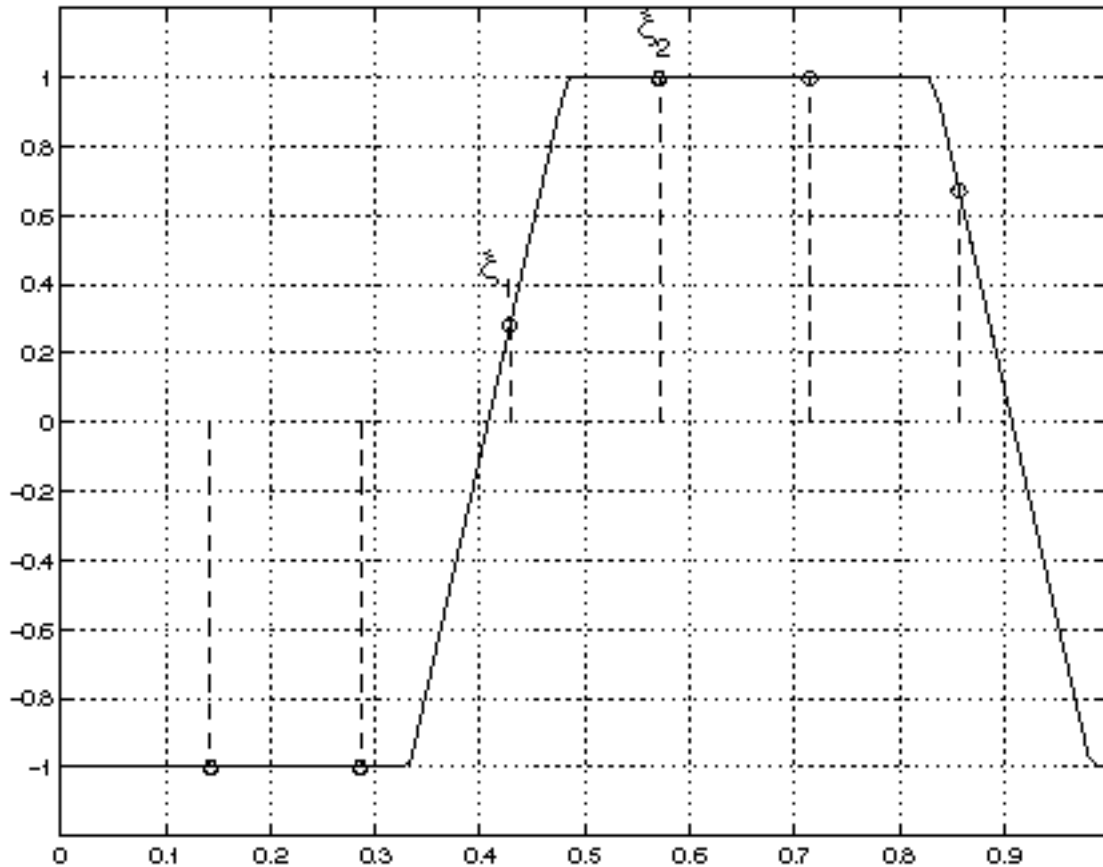
Noise in a circuit that is periodically driven, say with period  $T$ , exhibits statistical properties that also vary periodically. To understand time-domain characterization of noise, consider the simple circuit shown in Figure I-1.

**Figure I-1 Very Simple Mixer Schematic**



The amplitude of the noise measured at the RF output shown in Figure I-1 periodically varies depending on the magnitude of the modulating signal  $p(t)$ , as shown by the sample points in Figure I-2.

Figure I-2 Time-Varying Noise Process Analyzed at  $\xi_1$  and  $\xi_2$



In Figure I-2

- The solid line shows the envelope  $p(t)$  that modulates the noise process.
- The circles show possible phase points on the envelope where you might calculate the time-varying noise power.
- The circles marked  $\xi_1$  and  $\xi_2$  indicate the two phase points on the envelope where time-varying noise power is calculated.
- Noise in circuits that are periodically driven, say with period  $T$ , exhibits statistical properties that also vary periodically. To understand time-domain characterization of noise, consider the simple circuit shown in [Figure I-1](#) on page 1017. The amplitude of the noise measured at the RF output periodically varies depending on the magnitude of the modulating signal  $p(t)$ , as shown by the sample points (or circles on the signal envelope) in [Figure I-2](#) on page 1018.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

- Figure I-2 is a representation of periodically-modulated noise. It shows noise processes for two different phases in the periodic interval. Each process is stationary.

Virtuoso® Spectre® circuit simulator RF analysis (Spectre RF) can calculate the time-varying noise power at any point in the fundamental period. In fact, Spectre RF can calculate the full auto correlation function

$$R^{\xi}(p,q) = \langle x^{\xi}(p)x^{\xi}(p+q) \rangle = R^{\xi}(q)$$

and its spectrum for the discrete-time processes  $x^{\xi}$  obtained by periodically sampling the time-domain noise process at the same point in phase.

Figures I-3 and I-4 show two such noise processes for two different phases in the periodic interval. Each process is stationary. Figure I-3 shows the noise process for the phase marked  $\xi_I$  in Figure I-2 on page 1018.

**Figure I-3 Noise Process for Phase  $\xi_I$**

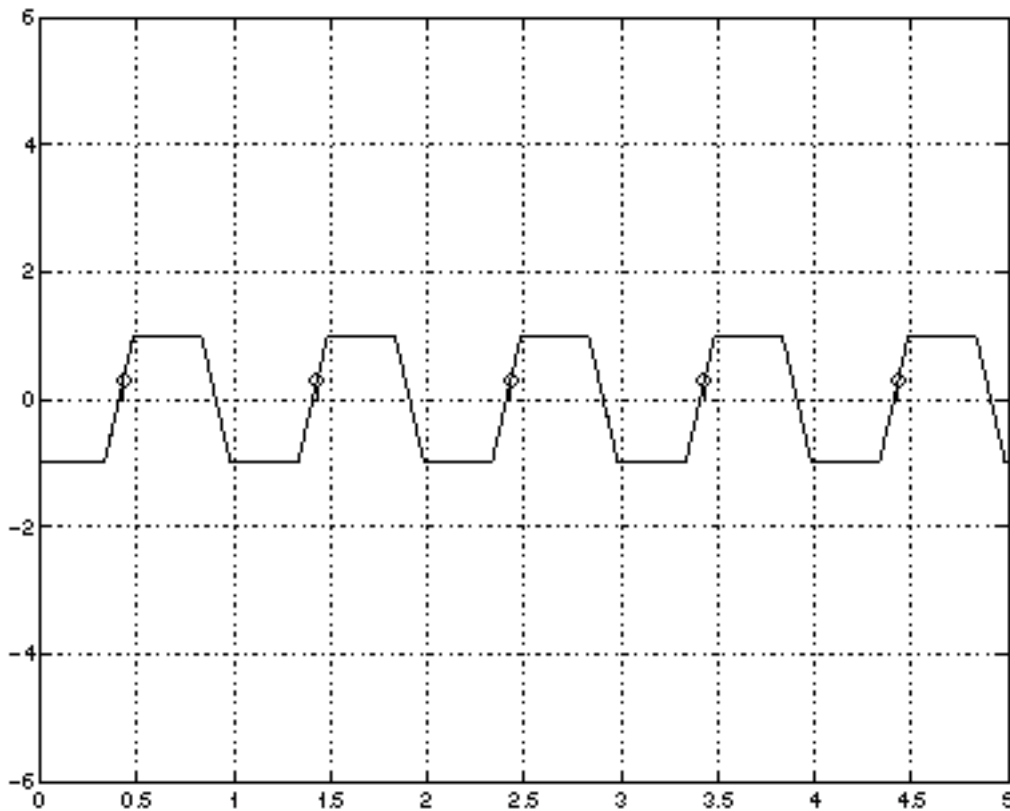
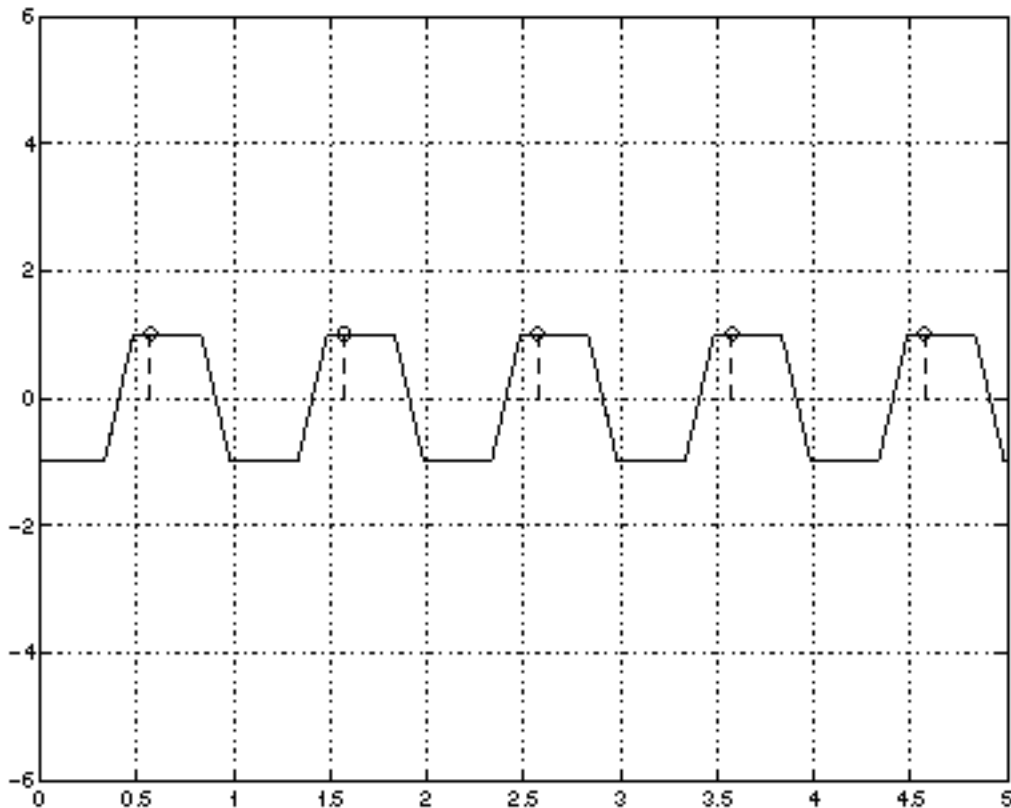


Figure I-4 shows the noise process for the phase marked  $\xi_2$  in Figure I-2 on page 1018.

**Figure I-4 Noise Process for Phase  $\xi_2$**



See the “[Reference Information on Time-Varying Noise](#)” on page 1032 for a more detailed introduction to noise in periodically time-varying systems.

#### Calculating Time Domain Noise

The following steps tell you how to calculate time-domain noise using Spectre RF.

1. In a terminal window, type `icms` to start the environment.
2. In the Simulation window, select *Analyses – Choose*.  
The Choosing Analyses form appears.
3. In the Choosing Analyses form, highlight `pss` and perform the PSS analysis setup.
4. In the Choosing Analyses form, highlight `pnoise`.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

5. The Choosing Analyses form changes to let you specify information for a Pnoise analysis.

6. In the Choosing Analyses form, perform the following:

a. Choose Noise Type *timedomain*.

b. Specify an appropriate frequency range and sweep for the analysis.

You might, for example, perform a linear sweep up to the fundamental frequency. Because each time point in the calculation is a separate frequency sweep, use the minimum number of frequency points possible to resolve the spectrum. This step minimizes computation time.

c. Specify a *noiseskipcount* value or specify additional explicit time points with *noisetimepoints*.

d. Specify an appropriate set of time points for the time-domain noise analysis.

e. Use *noiseskipcount* to calculate time-domain noise for one of every *noiseskipcount* time points.

If you set *noiseskipcount* to a value greater than or equal to zero, the simulator uses the *noiseskipcount* parameter value and ignores any *numberofpoints* parameter value. When *noiseskipcount* is less than zero, the simulator ignores the *noiseskipcount* parameter. The default is *noiseskipcount* = -1.

You can add specific points by specifying a time relative to the start of the PSS simulation interval. *noiseskipcount* = 5 performs noise calculations for about 30 time points in the PSS interval.

If you only need a few time points, add them explicitly with the *noisetimepoints* parameter and set *noiseskipcount* to a large value like 1000.

7. In the Simulation window, choose *Simulation – Netlist and Run*.

8. The simulation runs.

9. In the Simulation window, choose *Results – Direct Plot – PSS*.

The PSS Results form appears.

10. To calculate time-varying noise power, perform the following steps in the PSS Results form:

a. Click *tdnoise* and then select *Integrated noise power*.

b. Type 0 as the start frequency and the PSS fundamental frequency as the stop period.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

For example, type 1G if the PSS period is 1ns.

A periodic waveform appears that represents the expected noise power at each point in the fundamental period.

11. To display the spectrum of the sampled processes, perform the following steps in the PSS Results form:
  - a. Highlight *Output Noise*.
  - b. Highlight *Spectrum* for the type of sweep.
  - c. Clicking on *Plot*.

A set of curves appears, one for each sample phase in the fundamental period.

12. To calculate the autocorrelation function for one of the sampled processes, perform the following steps:
  - a. Display the spectrum using instructions from [step 11](#).
  - b. In the Simulation window, choose *Tools – Calculator*.
  - c. The calculator appears.
  - d. Click *wave* in the calculator and select the appropriate frequency-domain spectrum.

One of the sample waveforms is brought into the calculator

- e. Choose *DFT* from the list of special functions in the calculator. Then set 0 as the *From* and the PSS fundamental as the *To* value.
- f. Choose an appropriate window (e.g., *Cosine2*) and number of samples (around the number of frequency points in the interval  $[0, 1/T]$ ),
- g. Apply the *DFT* and plot the results.

Harmonic  $q$  of the DFT results gives the value of the discrete autocorrelation for this sample phase,  $R(q)$ .

Be sure the noise is in the correct units of power (e.g.,  $V^2/\text{Hz}$ ), not  $V/\text{square root of Hz}$ ) before performing the DFT to obtain the autocorrelation.

## Calculating Noise Correlation Coefficients

To characterize the noise in multi-input/multi-output systems, it is necessary to calculate both the noise power at each port and the correlation between the noise at various ports. The situation is complicated in RF systems because the ports may be at different frequencies. For example, in a mixer, the input port may be at the RF frequency and the output port at the IF frequency.

Denote the power spectrum of a signal  $x$  by  $S_{XX}(\omega)$ , that is

$$S_{XX}(\omega) = X^*(\omega) X(\omega)$$

where  $X(\omega)$  is the Fourier transform of the signal  $x(t)$ . For random signals like noise, calculate the expected value of the power spectrum  $S_{XX}(\omega)$ . To characterize the relationship between two separate signals  $x(t)$  and  $y(t)$ , you also need the cross-power spectrum

$$S_{XY}(\omega) = X^*(\omega) Y(\omega)$$

For random signals, the degree to which  $x$  and  $y$  are related is given by the cross-power spectrum. You can define a correlation coefficient  $\rho_{xy}(\omega)$  by

$$\rho_{XY}(\omega) = \frac{S_{XY}(\omega)}{\sqrt{S_{XX}(\omega)S_{YY}(\omega)}}$$

- A correlation coefficient ( $\rho_{xy}(\omega)$ ) of 0 indicates the signals are completely uncorrelated.
- A correlation coefficient of 1 indicates the signals are perfectly correlated. For example, a signal is always perfectly correlated with itself.
- You might also want to consider correlations between noise at different frequencies. The following quantity

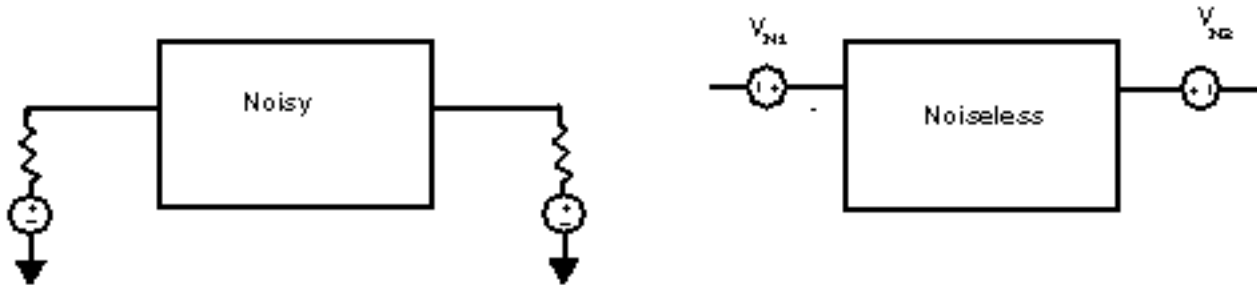
$$S_{XY}^{\alpha}(\omega)$$

expresses the correlation of a signal  $x$  at frequency  $\omega$  with the signal  $y$  at frequency  $\omega + \alpha$ . For example, white Gaussian noise is completely uncorrelated with itself for  $\alpha \neq 0$ . Noise in an RF system generally has  $S^{\alpha}(\omega)$  non-zero when  $\alpha$  is the fundamental frequency, for example, the LO frequency in a mixer.

Once you have measured the noise properties of a circuit, you can represent the circuit as a noiseless multiport with equivalent noise sources. For example, in Figure [I-5](#), first you

measure the noise voltage appearing at the excitation ports of the circuit on the left in Figure I-5. Then, you can express the noise properties of the circuit as two equivalent frequency-dependent noise voltages  $V_{N1}$  and  $V_{N2}$ , and a complex correlation coefficient  $\rho$ <sup>12</sup>.

**Figure I-5 Calculating Noise Correlations and Equivalent Noise Parameters**

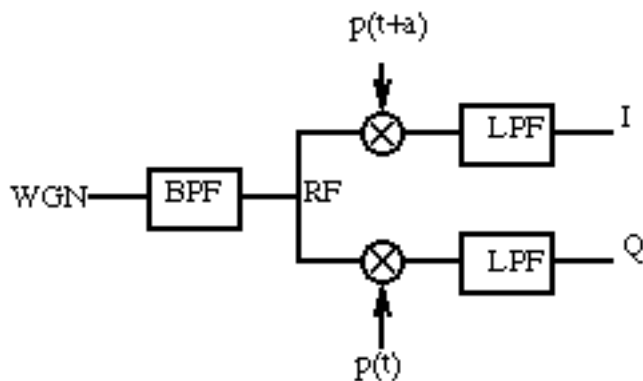


When you know the noise at each port and its correlation, you can obtain any of various sets of equivalent noise parameters. For example, you can express noise in an impedance representation as the equivalent correlated noise voltage sources shown in Figure I-5, as equivalent noise resistances and the correlation parameters, and as  $F_{min}$ ,  $R_N$ ,  $G_{opt}$ , and  $B_{opt}$ .

#### Cyclostationary Noise Example

As an example which illustrates the various aspects of cyclostationary noise, consider the simple mixer circuit shown in Figure I-6.

**Figure I-6 Simple Mixer Circuit**



In this simple mixer circuit, white Gaussian noise passes through a high-order band-pass filter with center frequency  $\omega_0$ . Then it is multiplied by two square-waves which have a phase shift



$a$  with respect to each other. Finally the output of the ideal multipliers is put through a one-pole low-pass filter to produce  $I$  and  $Q$  outputs.

The time-domain behavior of the noise is examined first. The most dramatic effect can be seen by looking directly at the mixer outputs in [Figure I-7](#) on page 1025. This figure shows the contributions to the time-varying noise power made by three separate source frequencies. Two of the source frequencies were selected around  $\omega_0$ , the third source frequency was selected away from  $\omega_0$ , slightly into the stop band of the band-pass filter. The sharp change in noise power over the simulation interval occurs because the mixers were driven with square-wave LO signals.

**Figure I-7 Time-Varying Noise Power Before Low-Pass Filter**

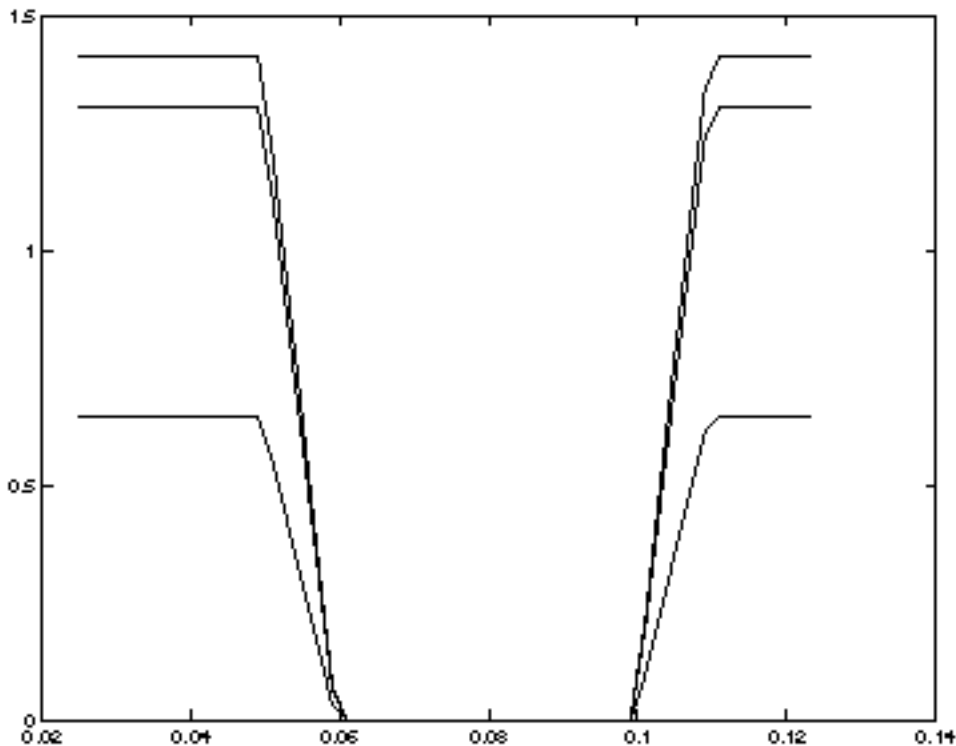
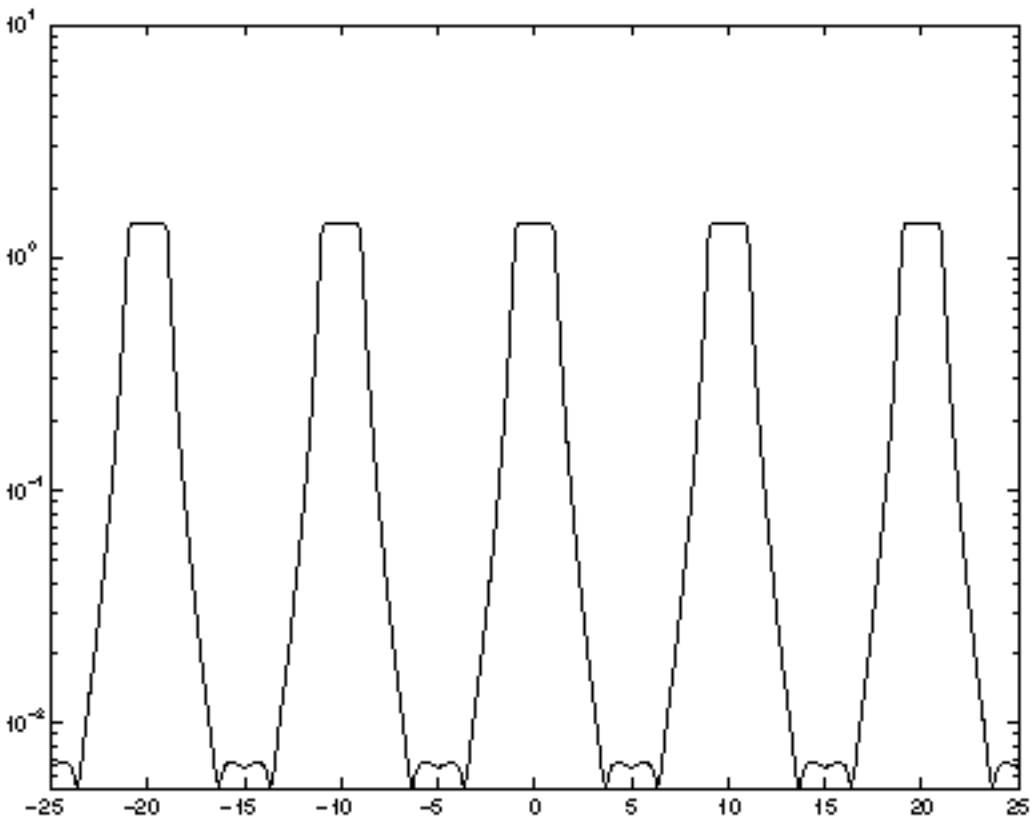


Figure I-8 shows the spectrum of a sampled noise process. Note the periodically replicated spectrum.

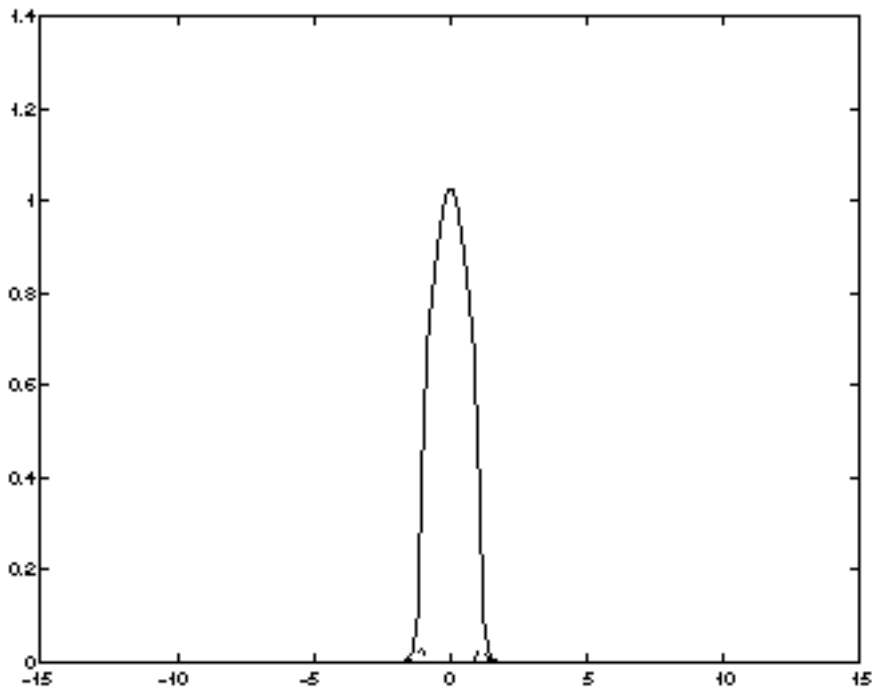
**Figure I-8 Spectrum of a Sampled Noise Process**



The noise behavior at the output ports is examined next. The output spectra at the  $I$  and  $Q$  outputs are shown in Figures I-9 and I-10. The noise density at  $I$  is concentrated around zero

because the noise at the RF input to the mixers (band-limited around  $\omega_0$ ) is shifted down to zero and up to  $2\omega_0$ , but components not around zero are eliminated by the low-pass filter.

**Figure I-9 Power Spectra With LO Tones 90<sup>deg</sup> Out of Phase**



More interesting is the cross-correlation spectrum of the  $I$  and  $Q$  outputs, shown as the dashed line in Figures [I-9](#) and [I-10](#). When the signals applied to the mixers are 90 degrees out of phase (as in Figures [I-9](#)), the cross-power spectral density of the noise at the separate  $I$  and  $Q$  outputs is small, indicating little noise correlation. If the tones are not quite out of phase (as in Figures [I-10](#)), the correlation is much more pronounced, though in neither case is it completely zero.

In Figures [I-9](#) and [I-10](#), the solid and dashed lines represent the following

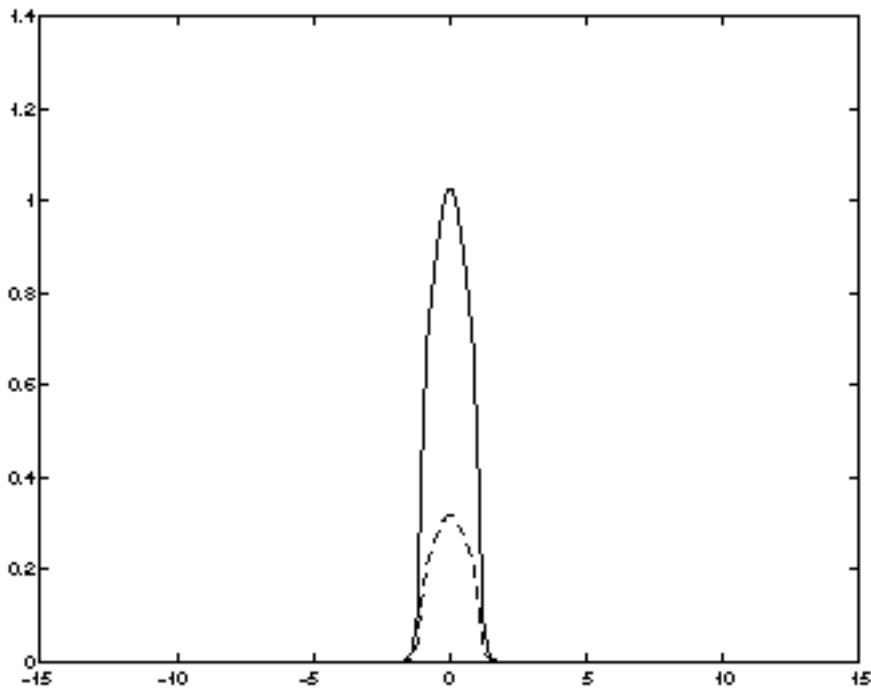
- The solid line represents the power spectrum for the  $I$  output with the function

$$S_{II}(\omega)^0$$

- The dashed line represents the cross-spectral density for  $I$  and  $Q$  with the function

$$S_{IQ}^0(\omega)$$

**Figure I-10 Power Spectra With LO Tones 72<sup>deg</sup> Out of Phase**



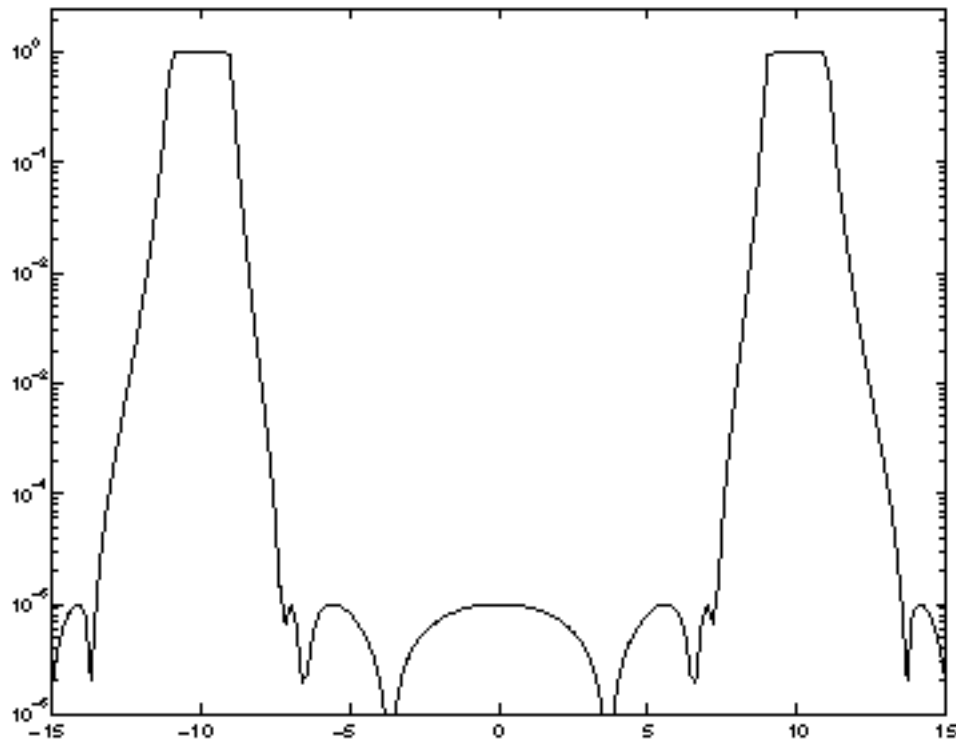
A more interesting example comes from examining the correlation between the noise at the  $I$  output and the noise at the RF input. The density function as given by

$$S_{IR}^{(1)}(\omega)$$

is significant because it represents the correlation between the noise at the  $I$  output around the baseband frequency with the noise at the RF input,  $\omega_0$  higher in frequency. The correlation

is high because the noise at the RF input is centered around  $\omega_0$  and converted to zero-centered noise by the mixer.

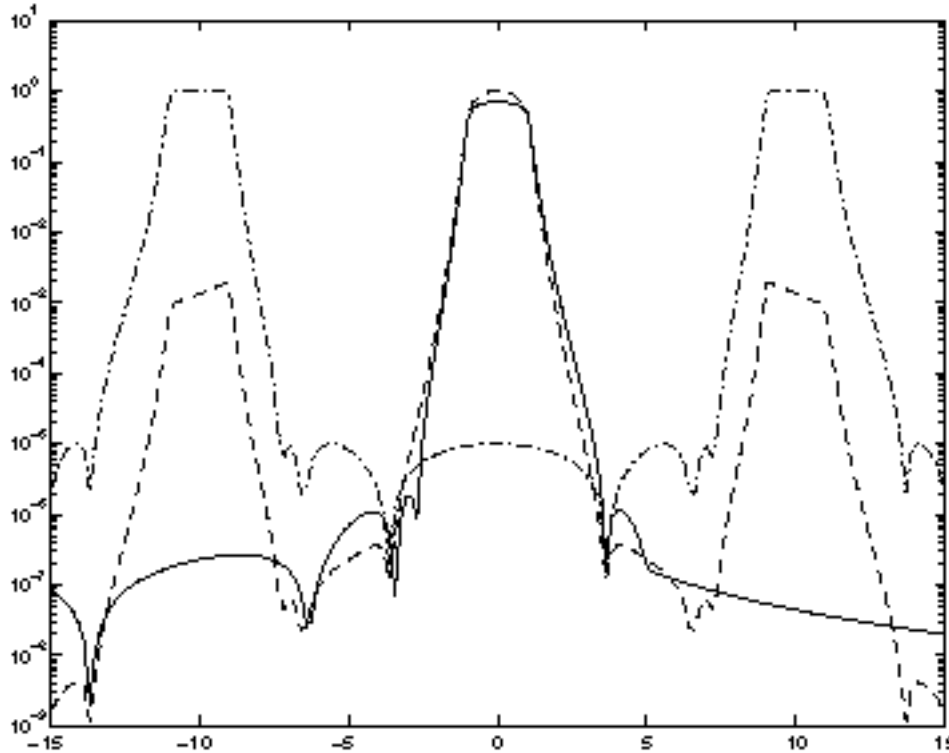
**Figure I-11 Noise Spectrum at the RF Input**



In Figure [I-11](#), the noise spectrum at the RF input is given by the following function

$$S_{RR}^0(\omega)$$

**Figure I-12 Noise Spectrum at Various Power Densities**



In Figure I-12, the solid, dashed, and dashed-dot lines represent the following

- The solid line represents the cross power spectrum which indicates correlation between output noise power at the  $I$  output versus noise at the RF input that is one harmonic higher in frequency. This is represented by the following function

$$S_{IR}^{(1)}(\omega)$$

- The dashed line represents the noise spectrum at the  $I$  output with the following function

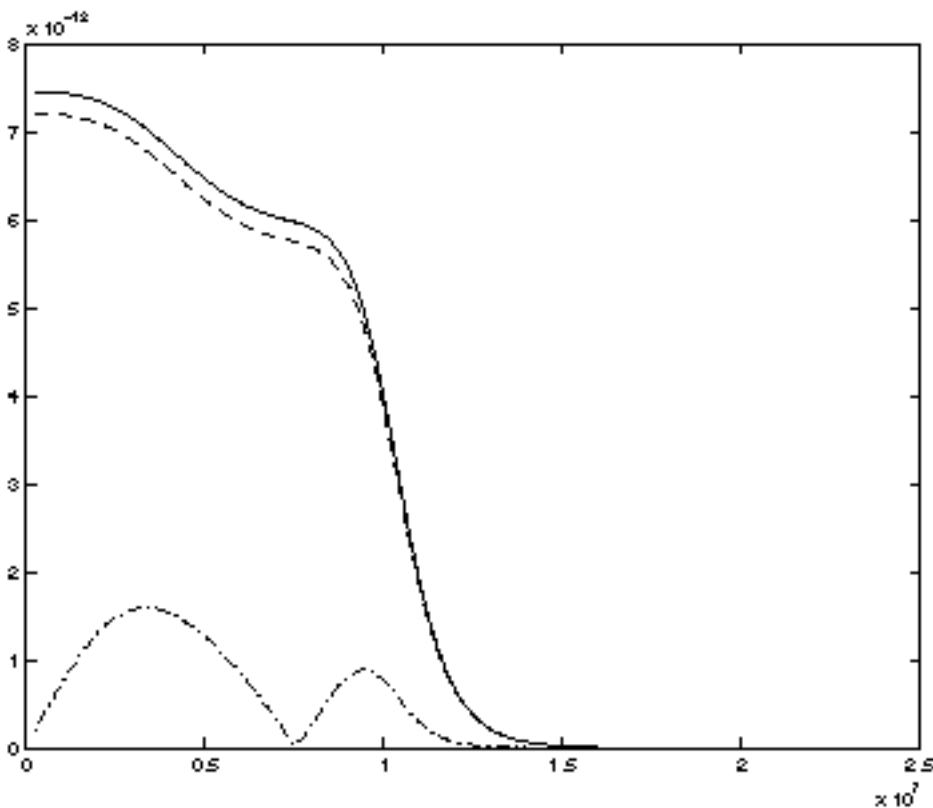
$$S_{II}^{(0)}(\omega)$$

- The dashed-dot line represents the noise spectrum at the RF input with the following function

$$S_{RR}^{(0)}$$

Finally a detailed circuit example was considered. A transistor-level image-reject receiver with  $I$  and  $Q$  outputs was analyzed. The noise spectra at the  $I$  and  $Q$  outputs were found to be very similar, as shown in Figure I-13.

**Figure I-13 Power Spectral Densities of an Image-Reject Receiver**

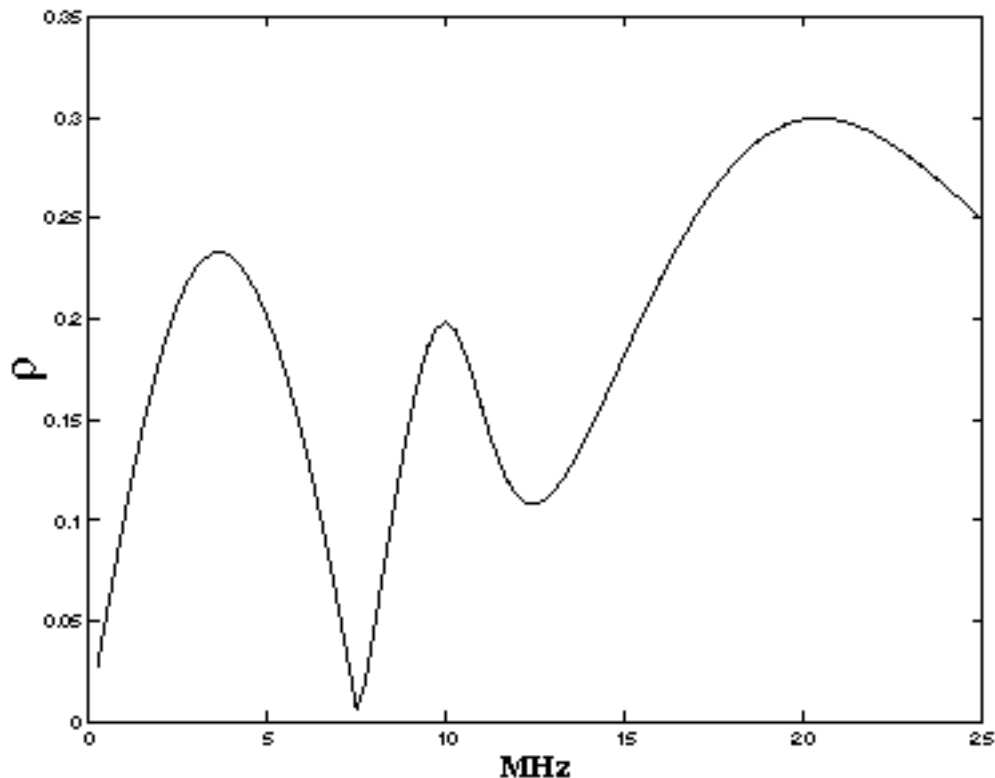


In the image-reject receiver example shown in Figure I-13, the power spectral densities are represented as follows

- $I$  output is a solid line
- $Q$  output is a dashed line
- $IQ$  cross-power density is a dash-dot line

The  $IQ$  cross-power density was smaller, but not negligible, indicating that the noise at the two outputs is partially correlated. The correlation coefficient between noise at the  $I$  and  $Q$  outputs of the image-reject receiver is shown in Figure I-14.

**Figure I-14 Correlation Coefficient for Output Noise of Image-Reject Receiver**



## Reference Information on Time-Varying Noise

The following sections provide background and reference information on the following noise-related topics

[Thermal Noise](#)

[Linear Systems and Noise](#)

[Time-Varying Systems and the Autocorrelation Function](#)

[Time-Varying Systems and Frequency Correlations](#)

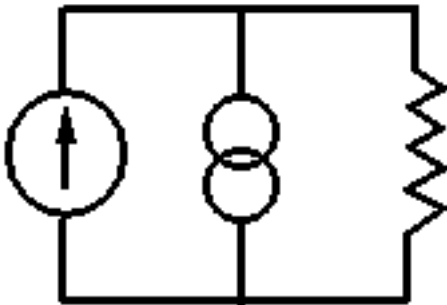
[Time-Varying Noise Power and Sampled Systems](#)



## Thermal Noise

The term *noise* is commonly used to refer to any unwanted signal. In the context of analog circuit simulation, noise is distinguished from such phenomena as distortion in the sense that it is non-deterministic, being generated from *random* events at a microscopic scale. For example, suppose a time-dependent current  $i(t)$  is driven through a linear resistor, as shown in Figure I-15.

**Figure I-15 Deterministic Current Source Driving a Noisy Linear Resistor**

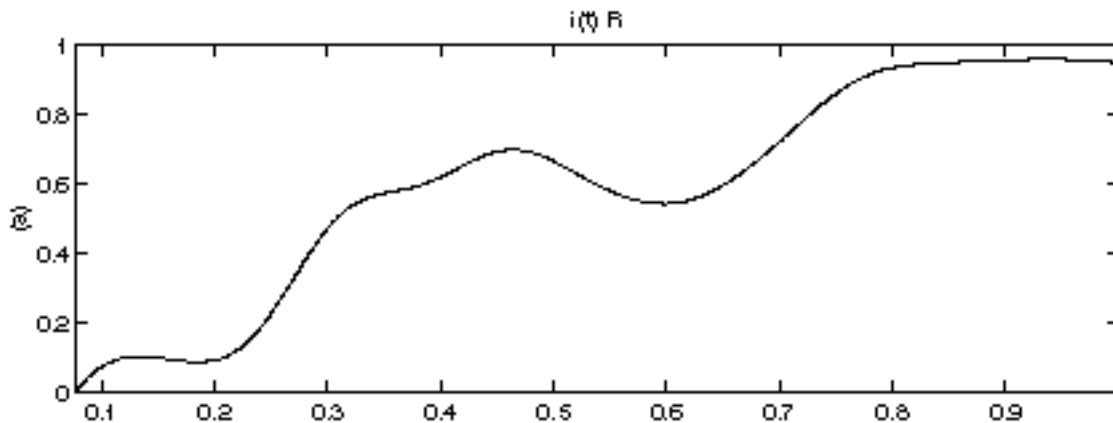


The voltage that appears across the resistor is

$$v(t) = i(t)R + n(t)$$

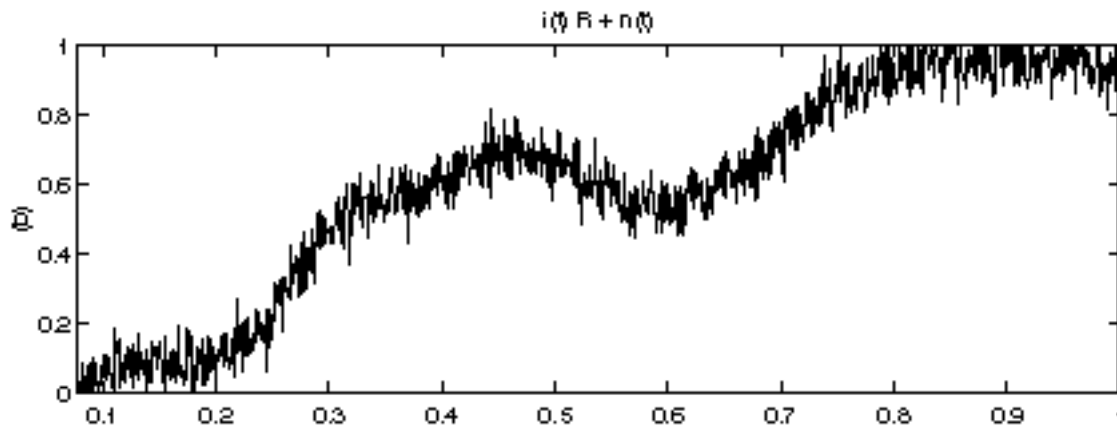
The desired signal  $i(t)R$ , shown in Figure I-16, is corrupted by an added noise voltage  $n(t)$  that is due to resistive thermal noise. The thermal noise of the resistor is modelled by a current source in parallel with the resistor.

**Figure I-16 The Desired Signal  $i(t)R$**



The total measured voltage is shown in Figure I-17.

**Figure I-17 The Total Measured Voltage**

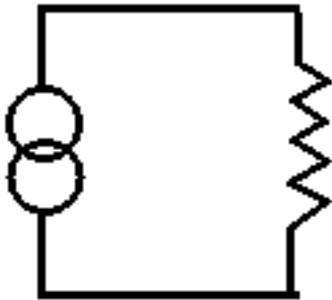


The added noise process alone,  $n(t)$ , is a random process and so it must be characterized in ways that are different than for deterministic signals. That is, at a time  $t_0$  the voltage produced by the driven current can be exactly specified—it is  $i_0 \sin t_0 R$ . Just by inspecting Figure I-16 we can predict this part of the measured signal.

On the other hand, the exact value of the noise signal cannot be predicted in advance, although it can be measured to be a particular value  $n(t_0)$ . However, if another measurement is performed, the noise signal  $n(t)$  we obtain is different and Figure I-17 changes. Due to its innate randomness, we must use a statistical means to characterize  $n(t)$ .

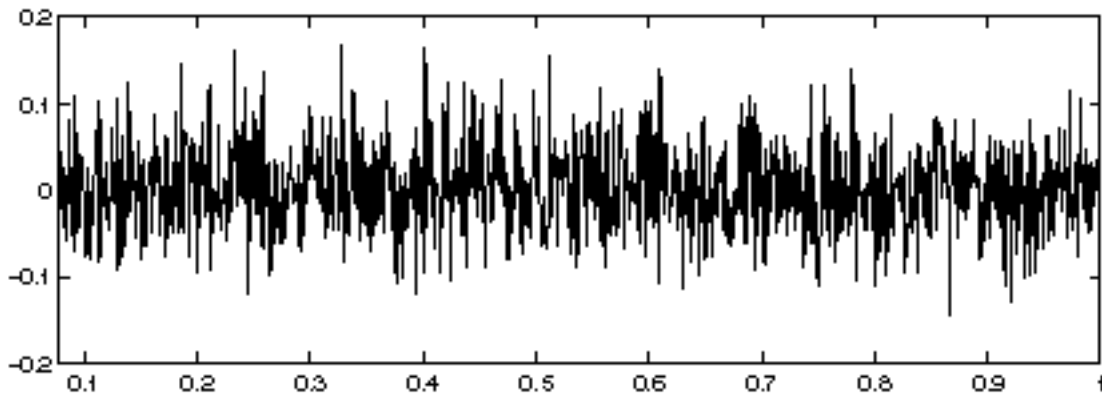
Now consider the circuit in Figure I-18, where we restrict attention to the noise source/resistor pair alone.

**Figure I-18 Resistor Modeled as a Noiseless Resistance with an Equivalent Noise Current Source**



A typical measured noise current/voltage is shown in Figure I-19.

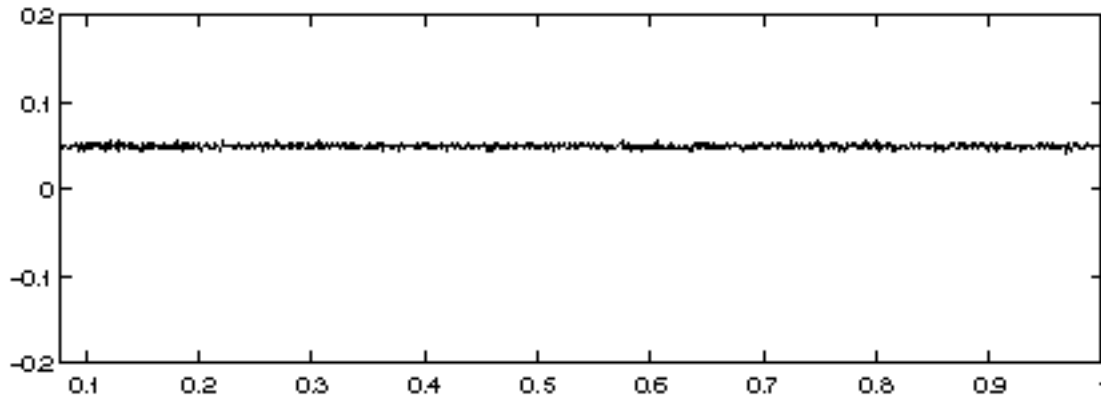
**Figure I-19 Typical Measured Noise Current/Voltage**



Since we cannot predict the specific value of  $n(t)$  at any point, we might instead try to predict what its value would be on average, or what we might *expect* the noise to be. For example, if we measure many noise voltage curves in the time domain,  $n(t)$ , and average over many

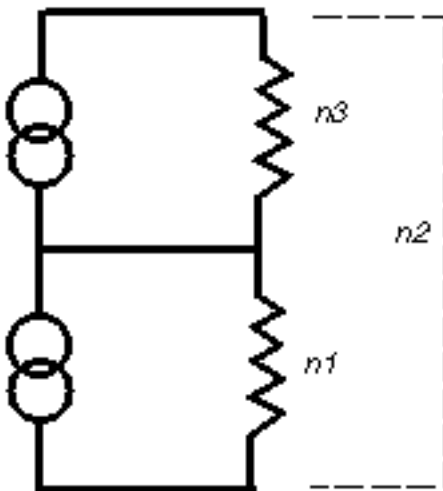
different curves, we obtain an approximation to the expected value of  $n(t)$  which we denote by  $E\{n(t)\}$ . For thermal noise, we find that  $E\{n(t)\} = 0$ . Therefore, instead of computing  $E\{n(t)\}$ , let us instead compute  $E\{n(t)^2\}$ , the expected noise power. An example of this sort of measurement is shown in Figure I-20. 250 measurements were needed to compute this curve.

**Figure I-20 Expected Noise Power**



Now suppose that we wish to tap the circuit at multiple points. Each point has its own noise characteristics, but they are not necessarily independent. Consider the circuit shown in Figure I-21.

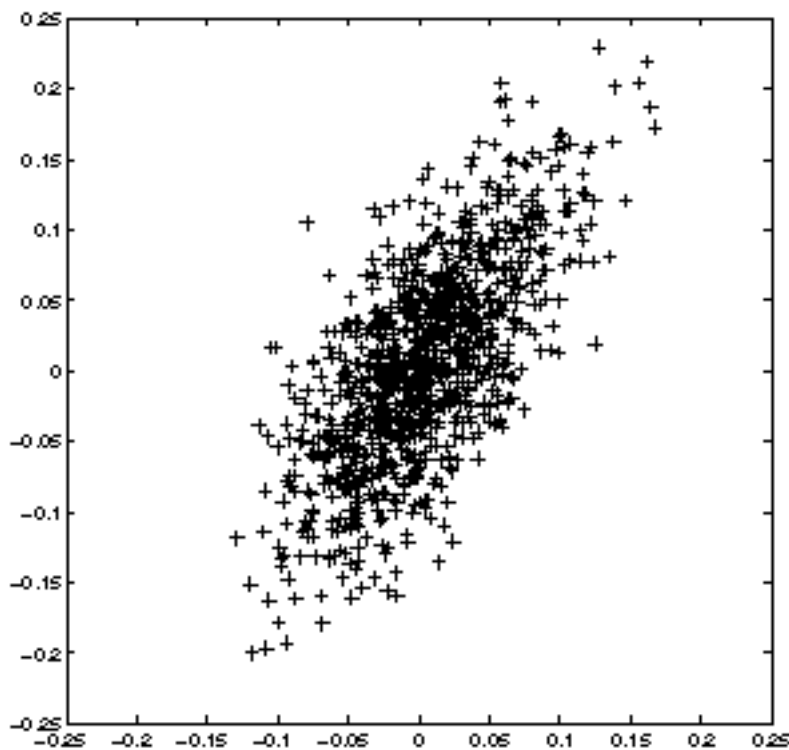
**Figure I-21 Circuit Illustrating Correlated Noise**



The signals  $n_1(t)$  and  $n_2(t)$  are obtained by measuring the voltage across a single resistor ( $n_1(t)$ ), and across both resistors ( $n_2(t)$ ), respectively. Just measuring  $E\{n_1(t)^2\}$  and  $E\{n_2(t)^2\}$  is not enough to predict the behavior of this system, because  $n_1(t)$  and  $n_2(t)$  are not independent.

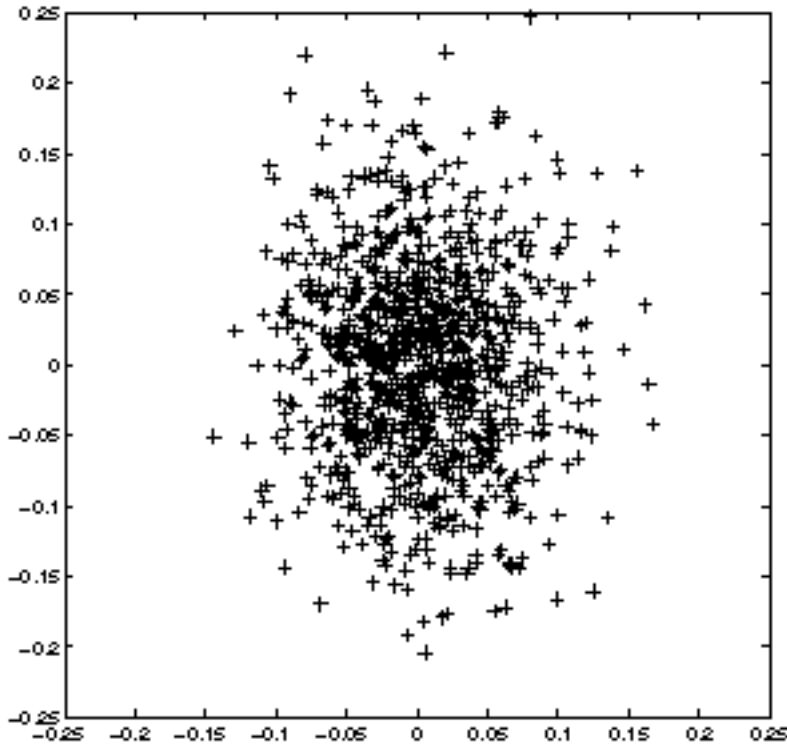
To see  $n_1(t)$  and  $n_2(t)$  are not independent, consider Figures [I-22](#) and [I-23](#). Samples of each of the processes are taken and plotted on an X-Y graph.

**Figure I-22 Samples of  $n_1(t)$  Plotted Versus  $n_2(t)$**



Because  $n_1(t)$  composes part of  $n_2(t)$ ,  $n_1(t)$  and  $n_2(t)$  are correlated so in Figure [I-22](#), the X-Y plot has a characteristic skew along the  $X=Y$  line, relative to the  $n_1(t)$ ,  $n_3(t)$  plot in Figure [I-23](#),

**Figure I-23 Samples of  $n_1(t)$  Plotted Versus  $n_3(t)$**



The signals  $n_1(t)$  and  $n_3(t)$  are uncorrelated because they represent thermal noise from different sources. The additional measurement needed to describe the random processes is the measurement of the correlation between the two processes,  $E\{n_1(t)n_2(t)\}$ . We can also define a time-varying correlation coefficient  $\rho$ , with  $\rho \in [0,1]$ , as

$$\rho(t) = \frac{E\{n_1(t)n_2(t)\}}{\sqrt{E\{n_1(t)^2\}E\{n_2(t)^2\}}}$$

A value of  $\rho=0$  indicated completely uncorrelated signals, and a value near one indicates a high degree of correlation. In this example we would find that  $\rho(t) = 1/2$ , representing the fact that each of the two noise sources contributes half of the process  $n_2(t)$ .

When there are multiple variables of interest in the system, it is convenient to use matrix notation. We write all the random processes of interest in a vector, for example

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

and then we can write the correlations as the expected value of a vector outer product,  $E\{x(t)x^H(t)\}$ , where the  $H$  superscript indicates Hermitian transpose.

For example, we might write a time-varying correlation matrix as

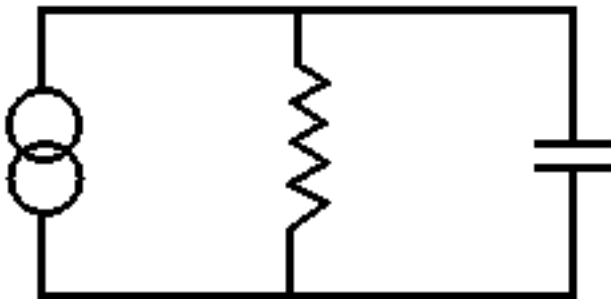
$$R_{xx}(t, t) \equiv E\{x(t)x^H(t)\} = \begin{bmatrix} E\{x_1(t)x_1(t)\} & E\{x_1(t)x_2(t)\} \\ E\{x_2(t)x_1(t)\} & E\{x_2(t)x_2(t)\} \end{bmatrix}$$

## Linear Systems and Noise

The examples in the preceding sections describe how to characterize purely static systems. Now we need to add some elements with memory, such as inductors and capacitors.

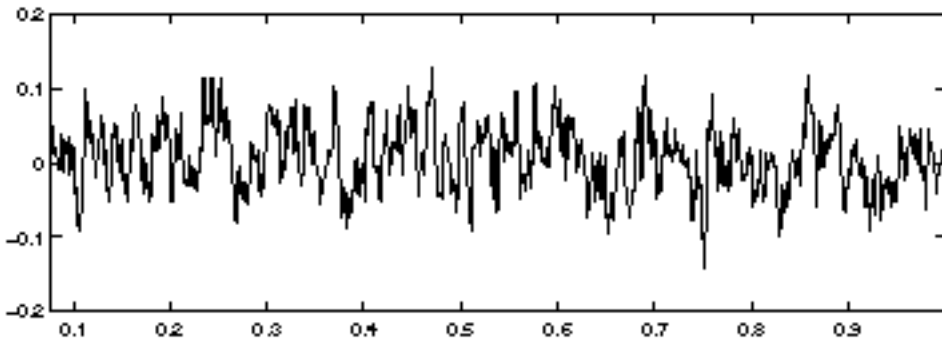
As a first example, consider adding a capacitor in parallel to the simple resistor, as shown in Figure I-24.

**Figure I-24 A Simple RC Circuit**



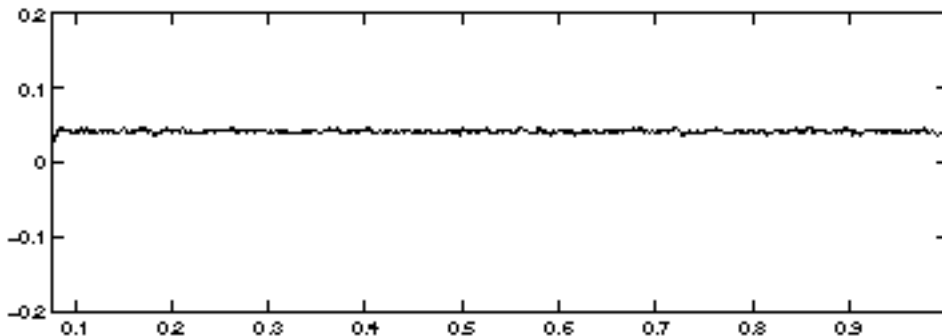
A sample of the noise process is shown in Figure I-25.

**Figure I-25 Noise Process for a Simple RC Circuit**



The noise looks different than the noise of the resistor alone, because the low-pass filter action of the RC circuit eliminates very high frequencies in the noise. However, we cannot see this effect simply by measuring  $E\{n(t)^2\}$  as shown in Figure I-26.

**Figure I-26 Expected Noise Power for an RC Circuit**



The measurement of  $E\{n(t)^2\}$  is independent of time for an RC circuit, just as it was for the resistor circuit.

### Spectral Densities in Two Simple Circuits

Instead of expected noise power, let us look at the expected power density in the frequency domain. Let  $n(\omega)$  denote the Fourier transform of one sample of  $n(t)$ . Then,  $E\{n(\omega)n(\omega)^*\}$  is the expected power spectral density, which we denote by  $S_n(\omega)$ .

In the present case, the capacitor has a pronounced effect on the spectral density. Figure I-26 shows a computed power spectral density for the resistor thermal noise previously considered. The spectrum is essentially flat (some deviations occur because a finite number



of samples was taken to perform the calculation). The flat spectrum represents the fact that in the resistor's noise, all frequencies are, in some statistical sense, equally present. We call such a process *white noise*.

### Power Spectral Density for Resistor Thermal Noise

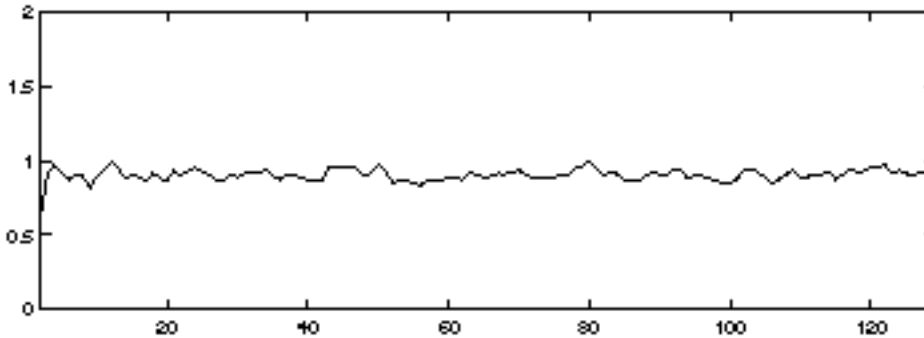
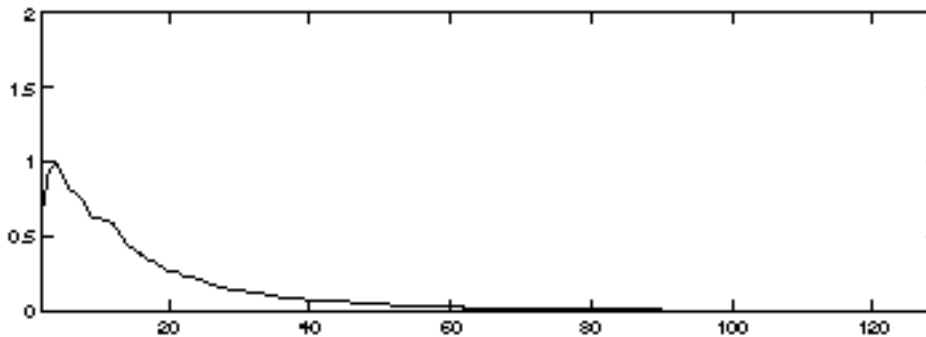


Figure I-27 shows the spectrum of the noise process after filtering by the resistor-capacitor system.

### Figure I-27 Resistor-Capacitor Filtered Spectral Noise Process



It is easy to rigorously account for the effect of the RC-filter on the power spectrum of the noise signal. Suppose a random signal  $x$  is passed through a time-invariant linear filter with frequency-domain transfer function  $h(\omega)$ . Then the output is  $y(\omega)=h(\omega)x(\omega)$ .

Because expectation is a linear operator, we can easily relate the power spectral density of  $y$ ,  $S_y(\omega)$  to  $S_x(\omega)$ , the power spectral density of  $x$ , by using the definitions of  $y$  and power density. Specifically,

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

$$S_y(\omega) = E\{y(\omega)y(\omega)^*\} = E\{h(\omega)x(\omega)x(\omega)^*h(\omega)^*\} = |h(\omega)|^2 S_x(\omega)$$

The noise from the resistor can be considered to be generated by a noise current source  $i$ , with power density

$$S_i(\omega) = \frac{4k_B T}{R}$$

placed in parallel with the resistor. With the capacitor in parallel, the transfer function from the current source to the resistor voltage is just the impedance  $Z(\omega)$ ,

$$h(\omega) = Z(\omega) = \frac{(1/C)}{j\omega + \frac{1}{RC}}$$

and so the noise voltage power density is

$$S_n(\omega) = \frac{\frac{4k_B T}{RC}}{\omega^2 + \left(\frac{1}{RC}\right)^2}$$

Clearly the spectrum is attenuated at high frequencies and reaches a maximum near zero.

For a vector process, we may define a matrix of power-spectral densities,

$$S_{xx}(\omega) \equiv E\left\{x(\omega)x^H(\omega)\right\}$$

The diagonal terms are simple real-valued power densities, and the off-diagonal terms are generally complex-valued cross-power densities between two variables. The cross-power density gives a measure of the correlation between the noise in two separate signals at a specific frequency. We may define a correlation coefficient as

$$\rho_{ij}(\omega) \equiv \frac{S_{x_i x_j}(\omega)}{[S_{x_i}(\omega)S_{x_j}(\omega)]^{1/2}}$$

It is often more useful to examine the correlation coefficient because the cross-power density may be small. As an example, consider a noiseless amplifier. The noise at the input is simply a scaled version of the noise at the output leading to a  $\rho=1$ , but the cross-power density is much smaller than the output total noise power density if the amplifier has small gain.

In a numerical simulation it is important to compute *only* the correlation coefficient when the diagonal spectral densities are sufficiently large. If one of the power densities in the denominator of the correlation-coefficient definition is very small, then a small numerical error could lead to large errors in the computed coefficient, because of division by a number close to zero.

In the vector case, the transfer function is also a matrix  $H(\omega)$ , such that  $y(\omega)=H(\omega)x(\omega)$  and so the spectral densities at the input and output are related by

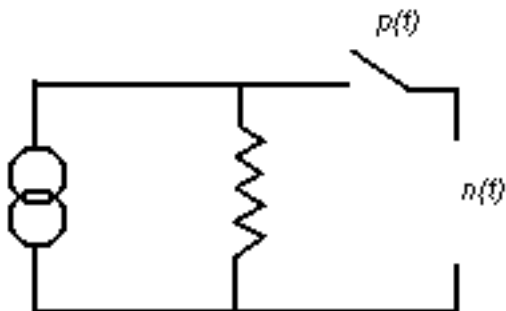
$$S_{yy}(\omega) = E\left\{H(\omega)x(\omega)x^H(\omega)H^H(\omega)\right\} = H(\omega)S_{xx}(\omega)H^H(\omega)$$

## Time-Varying Systems and the Autocorrelation Function

If all the sources of noise in a system are resistors, and the circuit consists strictly of linear time-invariant elements, then the matrix of spectral densities  $S_{xx}(\omega)$  is sufficient to describe the noise. However, most interesting RF circuits contain nonlinear elements driven by time-varying signals. This introduces time-varying noise sources as well as time-varying filtering. Because most noise sources are small, and generate small perturbations to the circuit behavior, for purposed of noise analysis, most RF circuits can be effectively modeled as linear time-varying systems. The simple matrix of power spectra is not sufficient to describe these systems.

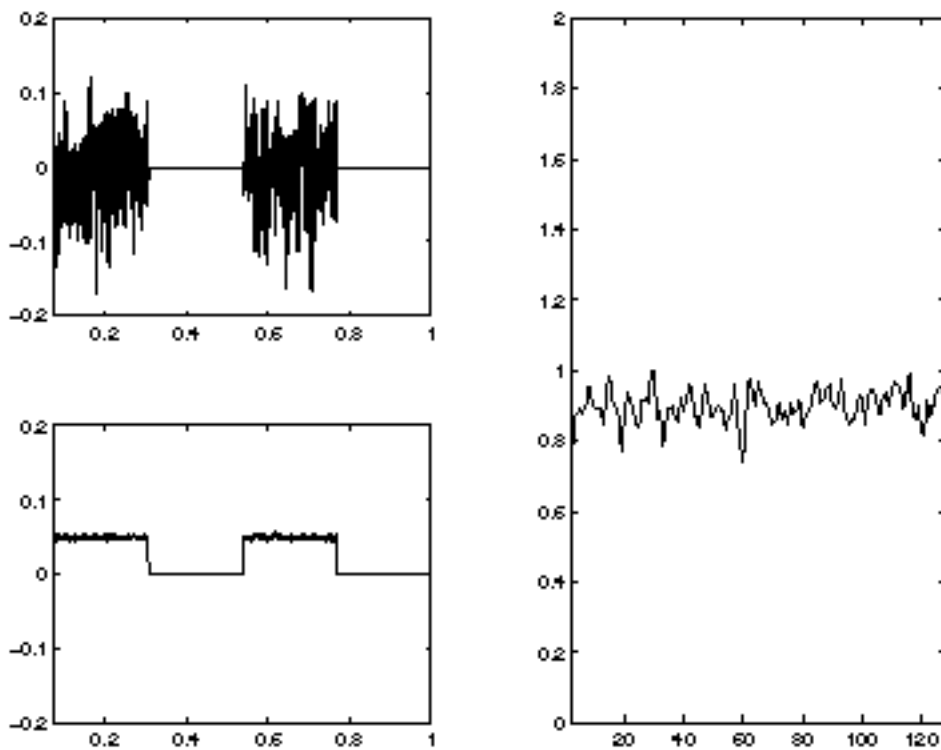
To see this, return to the simple resistor example. Suppose that a switch is connected between the resistor and the voltage measuring device, as shown in Figure [I-28](#).

Figure I-28 SimpleTime-Varying Circuit with Switch



Further suppose that the switch is periodically opened and closed. When the switch is open, there is no noise measured. When the switch is closed, the thermal noise is seen at the voltage output. A typical noise waveform is shown on the bottom left in Figure I-29.

Figure I-29 Typical Waveforms for Noise, Noise Power and White Noise



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

The time-varying noise power  $E\{n(t)^2\}$  can be computed and is shown in Figure I-29 on the top left, above the time-varying noise waveform. The expected power periodically switches between zero and the value expected from the resistor noise. This is different than the resistor-only and resistor-capacitor systems considered previously. Indeed, no linear time-invariant system could create this behavior. However, if we examine the power spectrum on the right in Figure I-29, we again find that it is flat, corresponding to *white* noise.

#### The Autocorrelation Function

At this point it is clear that  $E\{n(t)\}$  and  $E\{n(t)^2\}$  do not completely specify the random process  $n(t)$ , nor does the power spectral density. To obtain a complete characterization, consider measuring  $n(t)$  at two different timepoints,  $t_1$  and  $t_2$ .  $n(t_1)$  and  $n(t_2)$  are two separate random variables. They may be independent of each other, but in general they have some correlation. Therefore, to completely specify the statistical characteristics of  $n(t_1)$  and  $n(t_2)$  together, we must specify not only the variances  $E\{n(t_1)^2\}$  and  $E\{n(t_2)^2\}$ , but also the covariance  $E\{n(t_1)n(t_2)\}$ . In fact since  $n(t)$  has infinite dimension, an infinite number of these correlations must be specified to characterize the entire random process. The usual way of doing this is by defining the autocorrelation function  $R_n(t, t+\tau) = E\{n(t)n(t+\tau)\}$ .

If  $x(t)$  is a vector process,

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix}$$

then we define the autocorrelation matrix as

$$R_{xx}(t, t+\tau) \equiv E \left\{ x(t)x^H(t+\tau) \right\} = \begin{bmatrix} E\{x_1(t)x_1(t+\tau)\} & E\{x_1(t)x_2(t+\tau)\} \\ E\{x_2(t)x_1(t+\tau)\} & E\{x_2(t)x_2(t+\tau)\} \end{bmatrix}$$

where superscript  $H$  indicates Hermitian transpose.

The diagonal term gives the autocorrelation function for a single entry of the vector, e.g.,  $E\{x_1(t)x_1(t+\tau)\}$ . For  $\tau=0$ , this is the time-varying power in the single process, e.g.  $E\{x_1(t)^2\}$ . If the process  $x(t)$  is Gaussian, it is completely characterized by its autocorrelation function  $R_x(t, t+\tau)$  since all the variances and co-variances are now specified.

We can also precisely define what it means for a process to be *time-independent*, or *stationary*—A stationary process is one whose autocorrelation function is a function of  $\tau$  only, not of  $t$ . This means that not only is the *noise power*  $E\{n(t)^2\}$  independent of  $t$ , but the

correlation of the signal at a time point with the signal at another timepoint is only dependent on the difference between the timepoints,  $\tau$ . The white noise generated by the resistor, and the RC-filtered noise, are both stationary processes.

### Connecting Autocorrelation and Spectral Densities

At different points in the discussion above it was claimed that the expected time-varying power  $E\{n(t)^2\}$  of the resistor voltage is constant in time, and also the power density  $S_n(\omega)$  is constant in frequency. At first this seems odd because a quantity that is *broad* in time should be *concentrated* in frequency, and vice versa.

The answer comes in the precise relation of the spectral density to the autocorrelation function. Indeed, it turns out that the spectral density is the Fourier transform of the autocorrelation function, but with respect to the variable  $\tau$ , not with respect to  $t$ . In other words, the measured spectral density is related to the correlation of a random process with time-shifted versions of itself. Formally, for a stationary process  $R_n(t, t+\tau) = R_n(\tau)$  we write

$$S_n(f) = \int_{-\infty}^{\infty} e^{i\omega\tau} R_n(\tau) d\tau$$

For example, in the resistor-capacitor system considered above, we can calculate the autocorrelation function  $R_n(\tau)$  by an inverse Fourier transform of the power spectral density, with the result

$$R_n(\tau) = \left(\frac{4k_B T}{C}\right) e^{-|\tau|/(RC)}$$

From inspecting this expression we can see that what is happening is that adding a capacitor to the system creates memory. The random current process generated by the thermal noise of the resistor has no memory of itself so the currents at separate time-instants are not correlated. However, if the current source adds a small amount of charge to the capacitor, the charge takes a finite amount of time to discharge through the resistor creating voltage. Thus voltage at a time-instant is correlated with the voltage at some time later, because part of the voltage at the two separated time instants is due to the same bit of added charge. From inspecting the autocorrelation function it is clear that the correlation effects last only as long as the time it takes any particular bit of charge to decay, in other words, a few times the  $RC$  time constant of the resistor-capacitor system.

Note that the process is still stationary because this memory effect depends only on how long has elapsed since the bit of charge has been added, or rather how much time the bit of charge has had to dissipate, not the absolute time at which the charge is added. Charge added at

separate times is not correlated since arbitrary independent amounts can be added at a given instant. In particular, the time-varying noise power,

$$E\left\{n(t)^2\right\} = \int_{-\infty}^{\infty} S_n(\omega)d\omega$$

## Time-Varying Systems and Frequency Correlations

Now we have seen that the variation of the spectrum in frequency is related to the correlations of the process, in time. We might logically expect that, conversely, variation of the process in time (that is, non-stationarity) might have something to do with correlations of the process in frequency. To see why this might be the case, suppose we could write a random process  $x$  as a sum of complex exponentials with random coefficients,

$$x = \sum_{k=-K}^K c_k e^{i\omega t}$$

Noting that  $c_{-k}=c_k^*$ , the time-varying power in the process is

$$E\left\{x^2(t)\right\} = \sum_{k=-K}^K \sum_{l=-K}^K E\{c_k c_l^*\} e^{i(\omega_k - \omega_l)t}$$

and it is clear that  $E\{x(t)^2\}$  is constant in time if and only if

$$E\{c_k c_l^*\} = |c_k|^2 \delta_{kl}$$

In other words, the coefficients of expansion of sinusoids of different frequencies must be uncorrelated. In general, a stationary process is one whose frequency-domain representation contains no correlations across different frequencies.

To see how frequency correlations might come about, let us return to the resistor-switch example. Let  $n(t)$  denote the voltage noise on the resistor, and  $h(t)$  the action of the switch, so that the measure voltage is given by  $v(t) = h(t)n(t)$ , where  $h(t)$  is periodic with period  $T$  and frequency

$$\omega_0 = \frac{2\pi}{T}$$

The time-domain multiplication of the switch becomes a convolution in the frequency domain,

$$v(\omega) = h(\omega) \otimes n(\omega)$$

where  $\otimes$  denotes convolution.

Since  $h(t)$  is periodic, its frequency-domain representation is a series of Dirac deltas,

$$h(\omega) = \sum_k h_k \zeta(\omega - k\omega_0)$$

and so

$$v(\omega) = \sum_k h_k n(\omega - k\omega_0)$$

and the spectral power density is simply

$$S_v(\omega) = E\{\langle v(\omega)v(\omega)^* \rangle\} = \sum_k \sum_l h_k h_l^* E\{n(\omega - k\omega_0)n(\omega - l\omega_0)^*\}$$

Since the process  $n$  is stationary, this reduces to

$$S_v(\omega) = \sum_k |h_k|^2 S_n(\omega - k\omega_0)$$

Since  $S_n(\omega)$  is constant in frequency,  $S_v(\omega)$  is also.

However, the process  $v$  is no longer stationary because frequencies separated by multiples of  $\omega_0$  have been correlated by the action of the time-varying switch. We may see this effect in the time-variation of the noise power, as in [Figure I-29](#) on page 1044, or we may examine the correlations directly in the frequency domain.

To do this, we introduce the *cycle spectra*



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

$$S_{xx}^{\alpha}(\omega)$$

that are defined by

$$S_{xx}^{\alpha}(\omega) = E \left\{ x(\omega) x^H(\omega + \alpha) \right\}$$

and are a sort of cross-spectral density, taken between two separate frequencies.  $S_0(\omega)$  is just the power spectral density we have previously discussed. In fact we can define a frequency-correlation coefficient as

$$\rho_n^{\alpha}(\omega) \equiv \frac{S_n(\omega)^{\alpha}}{\sqrt{S_n(\omega) S_n(\omega + \alpha)}}$$

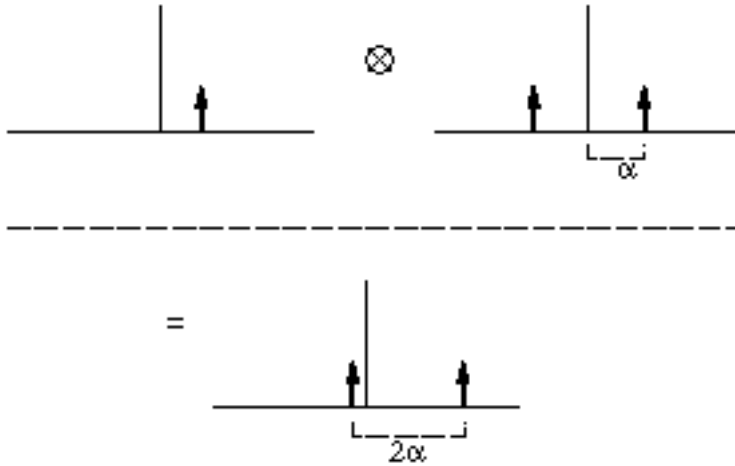
and if

$$\rho_n^{\alpha}(\omega) = 1$$

then the process  $n$  has frequency content at  $\omega$  and  $\omega + \alpha$  that is perfectly correlated.

Consider separating out a single frequency component of a random process and multiplying by a sinusoidal waveform of frequency  $\alpha$ , as shown in Figure I-30. The component at  $\omega$  is shifted to re-appear at  $\omega + \alpha$  and  $\omega - \alpha$ . The new process' frequency components at  $\omega - \alpha$  and  $\omega + \alpha$  are deterministically related to the components of the old process located at  $\omega$ . Therefore, they are correlated, and  $S^{2\alpha}(\omega)$  is non-zero.

**Figure I-30 Time-Variation Introduces Frequency Correlation**



Physically, what happens is that to form a waveform with a defined *shape* in time, the different frequency components of the signal must add in a coherent, or correlated fashion. In a process like thermal noise, the Fourier coefficients at different frequencies have phase that is randomly distributed with respect to each other, and the Fourier components can only add incoherently. Their powers add, rather than their amplitudes. Frequency correlation and time-variation of statistics are thus seen to be equivalent concepts.

Another way of viewing the cycle spectra is that they represent, in a sense, the two-dimensional Fourier transform of the autocorrelation function, and are therefore just another way of expressing the statistics of the process.

#### Time-Varying Noise Power and Sampled Systems

Again supposing the signal  $n$  to be cyclostationary with period  $T$ , for each sample phase  $\xi \in [0, T)$ , we may define the discrete-time autocorrelation function

$$R_n^\xi(p, q)$$

to be

$$R_n^\xi(p, p + q) = R_n(\xi + pT, \xi + (p + q)T)$$

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

Because the cyclostationary process  $R_n$  is periodic, by inspection

$$R_n^\xi(p, p+q)$$

is independent of  $p$  and thus stationary, that is

$$R_n^\xi(p, p+q) = R_n^\xi(q)$$

Note that

$$R_n^\xi(p, p) = R_n^\xi(0)$$

gives the expected noise power,  $R_n(\xi, \xi)$ , for the signal at phase  $\xi$ . Plotting  $R_n^\xi(0)$  versus  $\xi$  shows how the noise power varies periodically with time.

The discrete-time process

$$R_n^\xi(p, p+q) = R_n^\xi(q)$$

can be described in the frequency-domain by its discrete Fourier transform,

$$R_n^\xi(\phi) = \sum_{q=-\infty}^{\infty} R_n^\xi(q) e^{iq2\pi\phi T}$$

Note that the spectrum of the discrete (sampled) process

$$R_n^\xi(\phi)$$

is periodic in frequency with period  $1/T$ .

All noise power is aliased into the Nyquist interval  $[-1/2T, 1/2T]$  (or, equivalently, the interval  $[0, 1/T]$ ). Generally it is the noise spectrum which is available from the circuit simulator. To obtain the autocorrelation function or time-varying noise power, an inverse Fourier integral must be calculated by

$$R_n^{\xi}(q) = \int_0^{1/T} R_n^{\xi}(\phi) e^{iq2\pi\phi} d\phi$$

## Summary

- All useful noise metrics can be interpreted in terms of correlations. Physically these can be interpreted as the expected value of two-term products. In the case of random vectors these are expected values of vector outer products.
- The power spectral density of a variable indexed  $i$  is

$$S_{x_i x_i}(\omega) = E\{x_i(\omega)x_i(\omega)^*\}$$

This is what the current Spectre RF noise analysis computes.

$S_{xx}(\omega)$  is constant if and only if  $x$  is a white noise process. In that case  $R_{xx}(\tau) = R\delta(\tau)$  if there are no correlations in time for the process.

The cross-power densities of two variables  $x_i$  and  $x_j$  are

$$S_{x_i x_j}(\omega) = E\left\{x_i(\omega)x_j(\omega)^H\right\}$$

If and only if the two variables have zero correlation at that frequency, then

$$S_{x_i x_j} = 0$$

A correlation coefficient may be defined as

$$\rho_{ij}(\omega) \equiv \frac{S_{x_i x_j}(\omega)}{\sqrt{S_{x_i}(\omega)S_{x_j}(\omega)}}$$

and  $\rho_{ij}(f) \in [0,1]$ .

The cycle-spectra

$$S_{xx}^{\alpha}(f)$$

represent correlations between frequencies separated by the cycle-frequency  $\alpha$

$$S_{xx}^{\alpha}(f) = E \left\{ x(\omega) x^H(\omega + \alpha) \right\}$$

For a single process  $x_i$ , a correlation coefficient may be defined as

$$\rho_{x_i}^{\alpha}(\omega) \equiv \frac{S_{x_i x_i}^{\alpha}(\omega)}{\sqrt{S_{x_i}(\omega) S_{x_i}(\omega + \alpha)}}$$

and

$$\rho_{x_i}^{\alpha}(f) \in [0, 1]$$

- A process is stationary if and only if

$$S_{xx}^{\alpha}(\omega) = 0$$

for all  $\omega$  and all  $\alpha \neq 0$ , that is, if there are no correlations in frequency for the process.

In other words,

$$S_{xx}^{\alpha}(\omega) = S_{xx}(\omega) \delta(\alpha)$$

- A process is cyclostationary if

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Analyzing Time-Varying Noise

---

$$S_{xx}^{\alpha} = 0$$

for all  $\alpha \neq m\omega_0$  for some  $\omega_0$  and integer  $m$ . Frequencies separated by  $m\omega_0$  are correlated. A stationary process passed through a periodically linear-time varying filter in general is cyclostationary with  $\omega_0$  the fundamental harmonic of the filter.

We might also compute correlations between different nodes at different frequencies, with the obvious interpretation and generalization of the correlation coefficients.

---

## Using Tabulated S-parameters

---

Many passive component models commonly used in RF designs are available only as tables of S-parameter data. You can completely characterize any linear, time-invariant circuit network or component by specifying its S-parameter network at each frequency of interest. See [“Using the nport Components”](#) on page 1056 for information on how to use the S-parameter data tables as input for RF analyses.

Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) uses time-domain shooting methods to achieve excellent performance on large, highly nonlinear circuits. Consequently, using S-parameter data in Spectre RF is not as straightforward as it is with frequency-domain simulators such as those based on harmonic balance. Before you perform an RF simulation, such as a PSS analysis, you must first convert the frequency-domain S-parameter data to an equivalent time-domain model. For large, complicated S-parameter data sets this conversion can be time-consuming. However, you can avoid repeating the time consuming conversion process. You can convert the S-parameter data set to a time-domain model only once and then use one of the *nport* components to read in the converted time-domain data set multiple times. Thus, you avoid converting the S-parameter data set more than once. The procedure is described in [“Model Reuse”](#) on page 1062.

You might encounter three potential difficulties while converting a frequency-domain S-parameter data set into an equivalent time-domain model.

Some frequency-domain S-parameter data sets do not have valid time-domain descriptions. Time-domain models must be stable and causal and you cannot generate a time-domain model for a frequency-domain data that lacks these properties.

When you convert the frequency-domain data to an equivalent time-domain model, the frequencies between the tabulated data points must be interpolated. This is because the time-domain description of the frequency-domain model depends on every frequency not just the frequencies given at the tabulated data points. A special, robust, high-order, rational interpolation algorithm performs the interpolation. The rational interpolation process introduces some error into the final model description. See [“Controlling Model Accuracy”](#) on page 1058 to understand and control this error. This section also describes how to deal with data that contains noise that might corrupt the rational interpolation process.

Any algorithm that converts S-parameter data to use it in a time-domain simulator must extrapolate the data outside the range of tabulated frequencies. By definition, some frequencies are not included in a tabular data file. In addition, extrapolation might introduce nonphysical effects into the model, particularly when the S-parameter data is given over a very narrow frequency range. See “[Troubleshooting](#)” on page 1061 for information on how to diagnose and solve any problems. This section also describes how to interpret the warning messages that Spectre RF produces when a non-physical extrapolation might be occurring.

## Using the *nport* Components

Use the *nport* components to read in S-parameter data. Follow these steps to prepare an *nport* component for use in Spectre RF simulations.

1. Select an appropriate *nport* component from the *analogLib*.

For example, you might select an *n2port* for a two-port S-parameter description. The number of ports ranges from one (*nport* and *n1port*) to four (*n4port*).

2. Select the *n2port* component for a two-port S-parameter description. Place the *n2port* component in the Schematic window.
3. In the Schematic window highlight the new component and then choose *Edit—Properties—Objects* to display the Edit Object Properties form for the *n2port* component.
4. In the *S-parameters data file* field, type the S-parameter data file name.

For example, *sparam1.dat*.

5. In the *S-parameter data format* cyclic field, select an industry standard data format if it describes the format of your S-parameter data file.

If your S-parameter data is in one of the following industry standard formats—*touchstone* or *citi*, you can select that format. If you do not select a data format, Spectre RF attempts to determine the data format.

You can use the *sptr* tool to convert your S-parameter data to a format that Spectre RF can read. For more information, see the “[The S-Parameter File Format Translator \(SPTR\)](#)” on page 1063.

6. In the *Multiplier* field, enter the multiplicity factor. The default value is 1.
7. In the *Scale factor* field, enter the frequency scale factor. The default value is 1.
8. In the *No. of Harmonics for PSS* field, enter the number of harmonics to consider in the PSS solution. The default value is 20.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using Tabulated S-parameters

---

9. In the *Thermal Noise* cyclic field, select *yes* or *no*.
10. In the *Thermal noise model* cyclic field, select *internal* or *external*.
11. Select *internal* to specify use of the internal thermal noise model.
12. Select *external* to use the noise data in the S-parameter data file.  
By default, Spectre RF uses external data whenever it is available.
13. In the *Use smooth data windowing* cyclic field, select *yes* or *no*.  
This determines whether or not to use a smooth data windowing function.
14. Set the *interpolation method* cyclic field to *rational* if you are planning to use Spectre RF analyses.  
(Select *linear* or *spline* when you are planning to use only Spectre analyses.)

#### *Important*

You must use *rational* interpolation with Spectre RF analyses.

When you select *rational*, four new fields are displayed.

15. In the *ROM data file* field, enter the path for the time-domain reduced order model data file (ROM) you are creating for the converted S-parameter data.  
  
You must enter an absolute pathname, for example, */usr/mydir/sparamtorom*. Once Spectre RF converts the S-parameter data to a time-domain model, the file (*sparamtorom* here) stores the time-domain model. Be sure the ROM data file name is distinct from the S-parameter data file name.  
  
Storing the time-domain model ensures that you do not have to repeat interpolation and conversion later. See [“Model Reuse”](#) on page 1062 for further details on the ROM data file.
16. In the *Relative error* field, enter the maximum relative tolerance to allow for rational interpolation errors.  
  
The default value is 0.01. When the *nport* model deviates from the supplied S-parameter data by a relative magnitude less than *relerr*, the deviation is generally ignored. Fill in the *Relative error* field as described in [“Controlling Model Accuracy”](#) on page 1058.
17. In the *Absolute error* field, enter the maximum absolute tolerance to allow for rational interpolation errors.

The default value is  $1e^{-4}$ . When the *nport* model deviates from the supplied S-parameter data by an absolute magnitude less than *abserr*, the deviation is generally ignored. Fill in the *Absolute error* field as described in “Controlling Model Accuracy” on page 1058.

18. In the *Rational order* field, enter the order of rational function to use in fitting the S-parameter data.

If you enter this argument, *relerr* and *abserr* are ignored in selecting the rational function interpolation order. If you do not enter this argument, the simulator attempts to select an order of rational interpolation that satisfies the *abserr* and *relerr* criteria. Fill in the *Rational order* field as described in “Controlling Model Accuracy” on page 1058.

19. Click *Apply* in the Edit Object Properties form for the *nport* component and run the simulation.
20. Choose *Design—Check and Run* in the Schematic window.
21. Run the simulation.

## Controlling Model Accuracy

The *nport* component has three parameters to control the accuracy of the rational interpolation process,

- *relerr* (The *Relative error* field)
- *abserr* (The *Absolute error* field)
- *ratorder* (The *Rational order* field)

You can use these parameters to trade off accuracy against model size and simulation time. In general, the more stringent the accuracy requirement, the higher the model order. Higher-order models require longer simulation time. Spectre RF can automatically generate a model that meets a specified accuracy requirement. You can also specify the model order directly to Spectre RF.

### Using *relerr* and *abserr*

Let

$$S_{ij}(\omega)$$

denote the *i,j* entry of the scattering parameter matrix at frequency  $\omega$  and let

$$\hat{S}_{ij}(\omega)$$

denote the corresponding rational interpolant.

The rational interpolation algorithm attempts to find an interpolant such that

$$\max_{\omega, i, j} |S_{ij}(\omega) - \hat{S}_{ij}(\omega)| < \max(\text{relerr} \times |S_{ij}(\omega)|, \text{abserr})$$

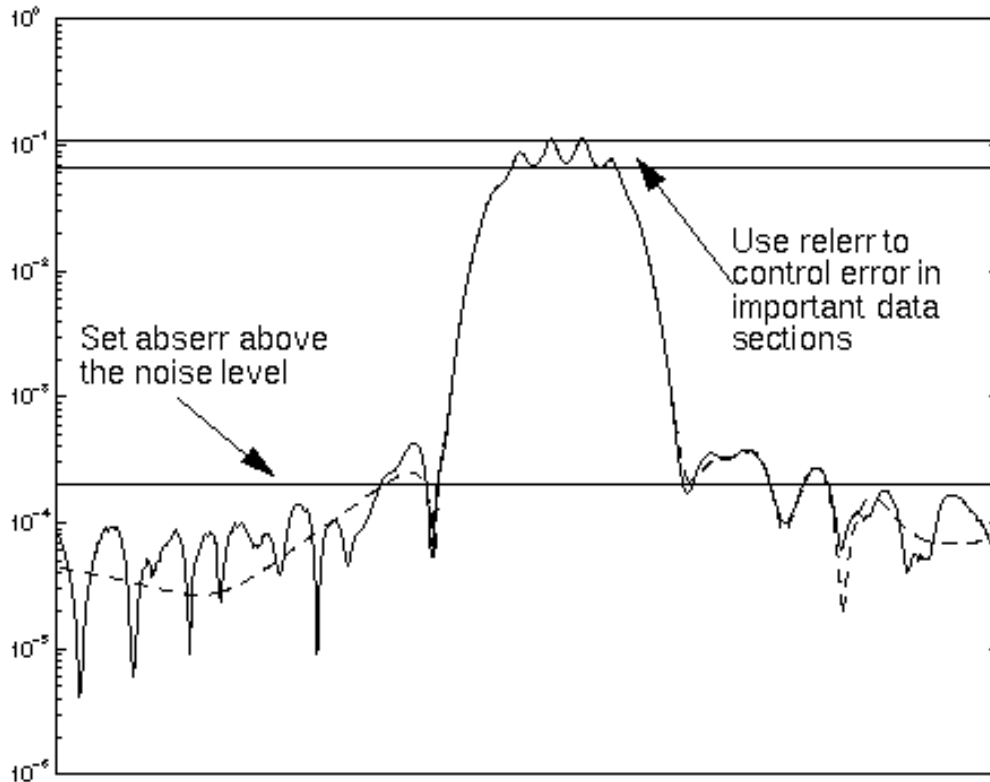
Generally, you use *relerr* for relative error control, and *abserr* for absolute error control.

Consider the data shown by the solid line in Figure [J-1](#). In this example

- *relerr* can control the error in the passband
- *abserr* can control the error in the stopband

If you are not interested in the details of the filter behavior in the stopband, you can set *abserr* to the level shown and these details are ignored in the interpolation.

Figure J-1 Using *relerr* and *abserr*



This example has approximately a 10% ripple in the passband. To ensure adequate error control there, set *relerr* to much less than 0.1, perhaps to 0.01.

The dashed line in Figure J-1 shows an interpolation function with the error control levels specified in Figure J-1: *abserr*= $2e^{-4}$ , and *relerr*=0.01. Details of the data below the *abserr* threshold are not resolved by the rational interpolation.

To recover the details in the stopband, set *abserr* to about  $1e^{-5}$  for this example.

If there is noise in the data, set *relerr* and *abserr* above the respective noise levels. Otherwise, Spectre RF attempts to interpolate the noise, resulting in very high order models and very long simulation times.

Remember that if you set *relerr* to zero, then, from the formula above, pure absolute error is used. Conversely, if you set *abserr* to zero, the error control is based solely on the errors relative to the magnitude of the input data.

## Using the *ratorder* Parameter

In general, it is usually best to specify only the accuracy parameters, *relerr* and *abserr*, and to let Spectre RF automatically select *ratorder*, the order of the rational approximation. However, if you have special information about your data set, you can direct Spectre RF to use a specific order of approximation in the rational interpolation. For example, if you know that your tabulated S-parameter data represents a sixth-order filter, you might instruct Spectre RF to use a seventh or eighth order fit. The slightly higher order gives Spectre RF flexibility to adjust for any noise or non-ideal behavior in the data.

When you specify the *ratorder* parameter, *relerr* and *abserr* are used to generate warnings when the order you selected is insufficient to meet accuracy requirements. Otherwise, the *relerr* and *abserr* parameters are not used.

## Troubleshooting

Certain S-parameter data sets might cause difficulty for the rational interpolation process. Types of data to avoid are

- Data specified only over a very narrow frequency range.  
Time-domain simulation requires time-domain models with a wider range of values; values that lie outside of this narrow frequency range. Data extrapolation might be difficult and is always risky.
- Very noisy data.  
Noisy data might lead to large, unreliable time-domain models.
- Data on a very sparse frequency grid.  
Accurate interpolation of such data might be impossible.
- Data with long ideal delays.
- Data representing idealized lossless elements, such as lossless transmission lines.

## Assessing the Quality of the Rational Interpolation

If you suspect a problem with the rational interpolation process, you can investigate it using the *sp* analysis with the following steps:

1. Construct a test schematic consisting of an *nport* component with interpolation set to rational, as discussed in [“Using the \*nport\* Components”](#) on page 1056.

2. Next add the appropriate number of *port* components to the schematic.
3. Perform an *sp* analysis on the *nport* component and look for anomalies.

Refer to [“Using S-Parameter Input Files”](#) on page 469 for an example of how to set up and run this type of simulation.

Large swings in interpolated values and S-parameter magnitudes greater than one both suggest a problem.

Large changes in the interpolant result from an inaccurate fit

S-parameter values greater than one result from a non-passive (energy-generating) model that might create unstable time-domain solutions. Be particularly critical of anomalies near the zero frequency (DC).

When the anomalous behavior occurs *within* the frequency range of data in the S-parameter data file, it usually indicates an inaccurate rational interpolation.

4. Verify that all the conditions listed at the beginning of the [Troubleshooting](#) section are met.
5. Try decreasing *relerr* or *abserr* or both.
6. Try specifying a higher-order interpolation with the *ratorder* parameter.

Remember that measured data can contain fine details that might require a higher order than you might expect from a casual inspection of the data.

If the anomalous behavior occurs *outside* the tabulated frequency range, try changing the *abserr*, *relerr*, or *ratorder* parameters. Sometimes anomalies can be removed by using a more accurate fit. However, there are limits to the ability to extrapolate outside the frequency interval you specify. You might need to specify additional data points to fix the problem.

## Model Reuse

The ROM data file (time-domain model) feature of the *nport* components lets you perform the conversion from a S-parameter data set to a time-domain model once and reuse the ROM data file in many designs.

For a given S-parameter data set, after you have specified a location for its ROM data file and Spectre RF has performed the conversion to a time-domain model and written the time-domain model to the ROM data file, you can reuse the model file in future simulations. To reuse the model for another *nport*, enter the time-domain model file name in the *ROM data*

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using Tabulated S-parameters

---

*file* field on the *nport* component's Edit Object Properties form in the new design. At this point, you no longer need to specify and convert a raw S-parameter data file.

When you specify both an S-parameter data file and a ROM data file,

- If the two files are consistent, Spectre RF reuses the time-domain model in the ROM data file
- If the two files are *not* consistent, Spectre RF generates a new time-domain model from the raw tabulated S-parameter data and overwrites the ROM data file with the new time-domain model data

This feature lets you specify both the ROM data file and the S-parameter data file in a design at the time when you place the *nport* component. Spectre RF then automatically generates the time-domain model during the first simulation and then reuses the time-domain model for all subsequent simulations without needing to change the *nport* component parameters.

If you require a more accurate rational interpolation, then you must regenerate the model in the ROM data file by changing the *relerr*, *abserr* or *ratorder* fields as described in [“Controlling Model Accuracy”](#) on page 1058.

## The S-Parameter File Format Translator (SPTR)

The S-parameter data file format translator (*sptr*) is a separate program from the Spectre RF simulator. You can find documentation for *sptr* in the *Virtuoso<sup>®</sup> Spectre User Guide*.

## References

To learn technical details about how Spectre RF converts S-parameter data to a time-domain description, see the article “Robust rational function approximation algorithm for model generation,” by C. P. Coelho, J. R. Phillips, and L. M. Silveira. This article appeared in the proceedings of the 36th Design Automation Conference, New Orleans, LA, June 1999.

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using Tabulated S-parameters

---



---

## Measuring AM, PM, and FM Conversion

---

### Derivation

Consider a sinusoid that is simultaneously both amplitude and phase modulated as in Equation [K-1](#).

$$(K-1) \quad v_m(t) = A_c(1 + \alpha(t))\cos(\omega_c t + \phi_c + \phi(t))$$

In Equation [K-1](#),  $A_c$ ,  $\phi_c$ ,  $\omega_c$ , are the amplitude, phase and angular frequency of the carrier, while  $\alpha(t)$ , and  $\phi(t)$  are the amplitude and phase modulation.

When you assume that  $\phi(t)$  is small for all  $t$ , this allows the narrowband angle modulation approximation as in Equation [K-2](#). See [ziemer76].

$$(K-2) \quad v_m(t) = A_c(1 + \alpha(t))[\cos(\omega_c t + \phi_c) - \phi(t)\sin(\omega_c t + \phi_c)]$$

Converting to complex exponentials gives Equation [K-3](#).

$$(K-3) \quad v_m(t) = \frac{A_c}{2}(1 + \alpha(t)) \left[ e^{j(\omega_c t + \phi_c)} + e^{-j(\omega_c t + \phi_c)} + j\phi(t) \left( e^{j(\omega_c t + \phi_c)} - e^{-j(\omega_c t + \phi_c)} \right) \right]$$

Letting both the amplitude and phase modulation be complex exponentials with the same frequency,  $\omega_m$ , gives Equations [K-4](#), [K-5](#), [K-6](#), [K-7](#) and [K-8](#).

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Measuring AM, PM, and FM Conversion

---

$$(K-4) \quad \alpha(t) = A e^{j\omega_m t}$$

$$(K-5) \quad \phi(t) = \Phi \frac{e^{j\omega_m t} + e^{-j\omega_m t}}{2}$$

Where

$$(K-6) \quad A = A e^{j\phi_A t}$$

$$(K-7) \quad \Phi = A e^{j\phi_\Phi t}$$

$$(K-8) \quad v_m(t) = \frac{A_c}{2} \left( 1 + A e^{j\omega_m t} \right) \left[ e^{j(\omega_c t + \phi_c)} + e^{-j(\omega_c t + \phi_c)} + j \frac{1}{2} \Phi \left( e^{j\omega_m t} + e^{-j\omega_m t} \right) \left( e^{j(\omega_c t + \phi_c)} - e^{-j(\omega_c t + \phi_c)} \right) \right]$$

Assuming that both A and  $\Phi$  are small and neglecting cross modulation terms gives Equation K-9.

$$(K-9) \quad v_m(t) = \frac{A_c}{2} \left[ e^{j(\omega_c t + \phi_c)} + e^{-j(\omega_c t + \phi_c)} + A e^{j\omega_m t} e^{j(\omega_c t + \phi_c)} + A e^{j\omega_m t} e^{-j(\omega_c t + \phi_c)} + j \frac{\Phi}{2} \left( e^{j\omega_m t} + e^{-j\omega_m t} \right) e^{j(\omega_c t + \phi_c)} - j \frac{\Phi}{2} \left( e^{j\omega_m t} + e^{-j\omega_m t} \right) e^{-j(\omega_c t + \phi_c)} \right]$$

Simplifying gives Equation K-10.

$$\begin{aligned}
 v_m(t) = & \frac{A_c}{2} \left[ e^{j(\omega_c t + \phi_c)} + e^{-j(\omega_c t + \phi_c)} \right. \\
 & + A e^{j((\omega_m + \omega_c) t + \phi_c)} + A e^{j((\omega_m - \omega_c) t - \phi_c)} \\
 & + \frac{1}{2} j \Phi \left[ e^{j((\omega_m + \omega_c) t + \phi_c)} - e^{j((\omega_m - \omega_c) t - \phi_c)} \right. \\
 & \left. \left. + e^{j((-\omega_m + \omega_c) t + \phi_c)} - e^{j((-\omega_m - \omega_c) t - \phi_c)} \right] \right]
 \end{aligned}
 \tag{K-10}$$

In Equation [K-10](#),

- The AM terms are

$$A e^{j((\omega_m + \omega_c) t + \phi_c)} + A e^{j((\omega_m - \omega_c) t - \phi_c)}$$

- The PM terms are

$$\begin{aligned}
 & \frac{1}{2} \left[ j \Phi e^{j((\omega_m + \omega_c) t + \phi_c)} - j \Phi e^{j((\omega_m - \omega_c) t - \phi_c)} \right. \\
 & \left. + j \Phi e^{j((-\omega_m + \omega_c) t + \phi_c)} - j \Phi e^{j((-\omega_m - \omega_c) t - \phi_c)} \right]
 \end{aligned}$$

Because the left-side term  $v_m(t)$  represents a real-time signal, only the real parts of the complex terms are of interest. Dropping the imaginary parts and rearranging Equation [K-10](#) produces Equation [K-11](#),

$$\begin{aligned}
 v_m(t) = & \frac{A_c}{2} \left[ e^{j(\omega_c t + \phi_c)} + e^{-j(\omega_c t + \phi_c)} \right. \\
 & + A e^{j(\omega_m - \omega_c) t} e^{-j\phi_c} - j \Phi e^{j(\omega_m - \omega_c) t} e^{-j\phi_c} \\
 & \left. + A e^{j(\omega_m + \omega_c) t} e^{j\phi_c} + j \Phi e^{j(\omega_m + \omega_c) t} e^{j\phi_c} \right]
 \end{aligned}
 \tag{K-11}$$

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Measuring AM, PM, and FM Conversion

---

When you ignore the negative  $\omega_m$  term in Equation [K-11](#), you get

- The LSB terms are

$$Ae^{j(\omega_m - \omega_c)t - j\phi_c} - j\Phi e^{j(\omega_m - \omega_c)t - j\phi_c}$$

- The USB terms are

$$Ae^{j(\omega_m + \omega_c)t + j\phi_c} + j\Phi e^{j(\omega_m + \omega_c)t + j\phi_c}$$

Assume that you perform a PAC analysis, which applies a single complex exponential signal that generates responses at the upper and lower sidebands of the  $\omega_c$  signal. Assume the transfer functions are  $L$  and  $U$ , so the lower and upper sideband signals are given by Equations [K-12](#) and [K-13](#).

$$(K-12) \quad l(t) = L e^{j(\omega_m - \omega_c)t}$$

$$(K-13) \quad u(t) = U e^{j(\omega_m + \omega_c)t}$$

Where

$$L = A_L e^{j\phi_L}$$

$$U = A_U e^{j\phi_U}$$

Matching common frequency terms between Equations [K-11](#), [K-12](#), and [K-13](#) gives Equations [K-14](#), [K-15](#), [K-16](#) and [K-17](#).

$$(K-14) \quad L = \frac{A_c}{2} \left( A e^{-j\phi_c} - j\Phi e^{-j\phi_c} \right)$$

$$(K-15) \quad U = \frac{A_c}{2} \left( A e^{j\phi_c} + j\Phi e^{j\phi_c} \right)$$

$$(K-16) \quad \frac{2}{A_c} L e^{j\phi_c} = A - j\Phi$$

$$(K-17) \quad \frac{2}{A_c} U e^{-j\phi_c} = A + j\Phi$$

Solving for the modulation coefficients gives Equations [K-18](#) and [K-19](#).

$$(K-18) \quad A = \frac{1}{A_c} \left( L e^{j\phi_c} + U e^{-j\phi_c} \right)$$

$$(K-19) \quad \Phi = \frac{j}{A_c} \left( L e^{j\phi_c} - U e^{-j\phi_c} \right)$$

Thus, Equation [K-18](#) gives the transfer function for amplitude modulation and Equation [K-19](#) gives the transfer function for phase modulation.

## Positive Frequencies

Notice that  $\tilde{L}$  is defined in [Equation K-12](#) on page 1068 to be the transfer function from the input to the sideband at  $\omega_m - \omega_c$ , which is a negative frequency. This is usually a natural definition for use with the Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) small signal analyses (depending on the setting of the `freqaxis` parameter). It can be cumbersome though when the only data available is at positive frequencies. Thus, the transfer function to  $\omega_c - \omega_m$  is defined as

$$\tilde{L}$$

Then, as in Equation [K-20](#),

$$(K-20) \quad \tilde{l}(t) = \tilde{L} e^{j(\omega_c - \omega_m)t}$$

Because the signals are real, L is a complex conjugate of

$$\tilde{L}$$

And the reverse is also true, as in Equation [K-21](#),

$$(K-21) \quad L = \tilde{L}^*$$

Equations [K-22](#) and [K-23](#) are produced by rewriting [Equation K-18](#) on page 1069 and [Equation K-19](#) on page 1069 in terms of

$$\tilde{L}$$

$$(K-22) \quad A = \frac{1}{A_c} \left( \tilde{L}^* e^{j\phi_c} + U e^{-j\phi_c} \right)$$

$$(K-23) \quad \Phi = \frac{j}{A_c} \left( \tilde{L}^* e^{j\phi_c} - U e^{-j\phi_c} \right)$$

## FM Modulation

For FM modulation, the phase modulation  $\phi(t)$  becomes the integral of the FM modulation signal,  $\omega(t)$  as shown in Equation [K-24](#).

$$(K-24) \quad v_m(t) = A_c \cos(\omega_c t + \phi(t))$$

Where

$$(K-25) \quad \phi(t) = \int \omega(t) dt$$

Recall from [Equation K-5](#) on page 1066 and [Equation K-19](#) on page 1069 that

$$(K-26) \quad \phi(t) = \frac{j}{A_c} \left( L e^{j\phi_c} - U e^{-j\phi_c} \right) e^{j\omega_m t}$$

Combining [Equation K-25](#) and [Equation K-26](#) and the differentiating both sides results in [Equations K-27](#), [K-28](#), and [K-29](#).

$$(K-27) \quad \omega(t) = \frac{\omega_m}{A_c} \left( U e^{-j\phi_c} - L e^{j\phi_c} \right) e^{j\omega_m t}$$

$$(K-28) \quad \Omega = A_c \omega e^{j\phi_\Omega} = \frac{\omega_m}{A_c} \left( U e^{-j\phi_c} - L e^{j\phi_c} \right)$$

Or

$$(K-29) \quad \Omega = j\omega_m \Phi$$

## Simulation

The test circuit, represented by the two netlists shown in [Example](#) on page 1072 and [Example](#) on page 1072, was run with Spectre RF. The test circuit consists of three, linear, periodically-varying modulators that are driven with the same input. The input is constant valued in the large signal PSS analysis, and generates a single complex exponential analysis during the PAC analysis. The idea is to compute the transfer functions from this input to the upper and lower sidebands at the output of the modulators and then use the derivation just

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Measuring AM, PM, and FM Conversion

---

described to convert these transfer functions into transfer functions to the AM, PM, and FM modulations and then check the simulation results against the expected results.

Notice that `freqaxis=out`. This is necessary to match the derivation. If you would rather use `freqaxis=absout`, you would have to use the complex conjugate of  $L$  as in [Equation K-22](#) on page 1070 and [Equation K-23](#) on page 1070.

#### Netlist for the AM, PM, and FM Conversion Test Circuit

```
// AM, PM, and FM modulation test circuit

simulator lang=spectre
ahdl_include "modulators.va"

parameters MOD_FREQ=10MHz
parameters CARRIER_FREQ=1GHz

Vin (in 0) vsource pacmag=1 pacphase=0
Mod0 (unmod in) AMmodulator freq=CARRIER_FREQ mod_index=0
Mod1 (am in) AMmodulator freq=CARRIER_FREQ mod_index=1
Mod2 (pm in) PMmodulator freq=CARRIER_FREQ kp=1
Mod3 (fm in) FMmodulator freq=CARRIER_FREQ fd=MOD_FREQ

waves pss fund=CARRIER_FREQ outputtype=all tstab=2ns harms=1
xfer pac start=MOD_FREQ maxsideband=4 freqaxis=out
```

The netlist for the modulator models shown in [Example K-29](#), has the filename `modulators.va`.

#### Netlist for the Modulator Models Written in Verilog-A

```
`include "discipline.h"
`include "constants.h"

module AMmodulator (out, in);
    input in;
    output out;
    electrical out, in;
    parameter real freq = 1 from (0:inf);
    parameter real mod_index = 1;

    analog begin
        V(out) <+ (1+mod_index*V(in)) * cos(2*`M_PI*freq*$abstime);
        $bound_step( 0.05 / freq );
    end
endmodule

module PMmodulator (out, in);
    input in;
    output out;
    electrical out, in;
    parameter real freq = 1 from (0:inf);
```



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Measuring AM, PM, and FM Conversion

---

```
parameter real kp = 1 from (0:inf);

analog begin
  V(out) <+ cos(2*_M_PI*freq*$abstime + kp*V(in));
  $bound_step( 0.05 / freq );
end
endmodule

module FMmodulator (out, in);
  input in;
  output out;
  electrical out, in;
  parameter real freq = 1 from (0:inf);
  parameter real fd = 1 from (0:inf);
  real phi;

  analog begin
    V(out) <+ cos(2*_M_PI*(freq*$abstime + idtmod(fd*V(in),0,1, -0.5)));
    $bound_step( 0.05 / freq );
  end
endmodule
```

## Results

The simulations were run with various values for `pacphase` on `Vin`.

Table [K-1](#) shows results for the output of the AM modulator with  $v_{LO} = \cos(\omega_c t)$ .

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Measuring AM, PM, and FM Conversion

---

Table [K-2](#) shows results for the output of the PM modulator with  $v_{LO} = \cos(\omega_c t)$ .

**Table K-1 Results for the AM Modulator Output**

pacphase	L	U	A	$\Phi$
0	1/2	1/2	1	0
45	$\frac{1+j}{2\sqrt{2}}$	$\frac{1+j}{2\sqrt{2}}$	$\frac{1+j}{2}$	0
90	$j/2$	$j/2$	$j$	0
180	-1/2	-1/2	-1	0

**Table K-2 Results for the PM Modulator Output**

pacphase	L	U	A	$\Phi$
0	-1/2	1/2	0	1
45	$\frac{1-j}{2\sqrt{2}}$	$\frac{j-1}{2\sqrt{2}}$	0	$\frac{1+j}{\sqrt{2}}$
90	1/2	-1/2	0	$j$
180	$j/2$	$-j/2$	0	-1

If you repeat the simulations but replace the *cos* function in the modulators with the *sin* function, which is equivalent to changing the LO to  $v_{LO} = \sin(\omega_c t)$  or setting  $\phi_c = -90$ , you achieve the following results.

- Table [K-3](#) shows results for the output of the AM modulator with  $v_{LO} = \sin(\omega_c t)$ .

**Virtuoso Spectre Circuit Simulator RF Analysis User Guide**  
Measuring AM, PM, and FM Conversion

---

- Table [K-4](#) shows results for the output of the PM modulator with  $v_{LO} = \sin(\omega_c t)$ .

**Table K-3 Results for the AM Modulator Output**

pacphase	L	U	A	$\Phi$
0	$j/2$	$-1/2$	1	0
45	$\frac{j-1}{2\sqrt{2}}$	$\frac{1-j}{2\sqrt{2}}$	$\frac{1+j}{\sqrt{2}}$	0
90	$-1/2$	$1/2$	$j$	0
180	$-j/2$	$j/2$	$-1$	0

**Table K-4 Results for the PM Modulator Output**

pacphase	L	U	A	$\Phi$
0	$1/2$	$1/2$	0	1
45	$\frac{1+j}{2\sqrt{2}}$	$\frac{1+j}{2\sqrt{2}}$	0	$\frac{1+j}{\sqrt{2}}$
90	$j/2$	$j/2$	0	$j$
180	$-1/2$	$-1/2$	0	$-1$

Finally, Table [K-5](#) shows the results for the FM modulator with  $v_{LO} = \cos(\omega_c t)$ . The FM modulator has a modulation coefficient of  $\omega_m$  built-in, which renormalizes the results.

**Table K-5 Results for the FM Modulator Output**

pacphase	L	U	$\Omega$
0	$-1/2$	$1/2$	1
45	$\frac{1+j}{2\sqrt{2}}$	$\frac{1+j}{2\sqrt{2}}$	$\frac{1+j}{\sqrt{2}}$
90	$-j/2$	$j/2$	$j$
180	$1/2$	$-1/2$	$-1$

## Conclusion

This appendix shows that the PAC analysis can be used to determine the level of AM or PM modulation that appears on a carrier. This is done by applying a small signal and using the phase of the carrier along with the transfer function to the upper and lower sidebands of the carrier to compute an AM or PM transfer function.

## References

- [Ziemer 76] R. Ziemer and W. Tranter. *Principles of Communications: Systems, Modulation, and Noise*. Houghton Mifflin, 1976.
- [Robins 96] W. Robins. *Phase Noise in Signal Sources (Theory and Application)*. IEE Telecommunications Series, 1996.



---

# Using PSP and Pnoise Analyses

---

## Overview of PSP and Pnoise Analyses

This appendix describes how to calculate small-signal quantities such as noise, noise figure, periodic scattering parameters, and gain in periodically-driven circuits. The appendix explains

- The concepts of periodic S-parameters,
- The concepts of noise correlation parameters
- The various definitions of noise figure and gain

Virtuoso<sup>®</sup> Spectre<sup>®</sup> circuit simulator RF analysis (Spectre RF) provides four small-signal analyses for circuits with a DC operating point: AC, XF, Noise and SP. The Spectre RF simulator also provides four small-signal analyses for circuits with a periodically time-varying operating point: PAC, PXF, Pnoise and PSP. Because the periodic small-signal analyses linearize the circuit about the time-varying operating point that is obtained using the PSS analysis, they can analyze frequency conversion effects.

- PAC analysis computes the small-signal response at all outputs to the small stimulus of a single group of sources.
- PXF analysis computes the transfer function from every source in the circuit to a single output.
- Pnoise analysis computes noise parameters such as noise figure as well as detailing noise contributions by devices.
- PSP analysis contains some of the capabilities of PAC, PXF, and Pnoise analyses. PSP analysis can compute periodic scattering parameters that describe the small-signal relations between several different ports in a circuit. It can also compute noise parameters, such as noise correlation matrices, equivalent noise sources, and noise figure.

## Periodic S-parameters

### Linear Time-Invariant S-Parameters

Designers of microwave and RF circuits typically characterize the frequency-dependent behavior of linear networks through sets of *scattering* or S-parameters. The notion of scattering parameters is rooted in transmission line concepts where the scattering parameter matrix relates the magnitude and phase of incident and reflected waves.

Consider an arbitrary  $N$ -port linear time-invariant (LTI) network, such as the two-port shown in [Figure L-1](#) on page 1079. Each port is driven by a source of reference impedance  $Z_i$ , where the index  $i$  runs from 1 to  $N$ . For the remainder of this document we assume a real valued reference impedance,  $R_i$ , for each port. In terms of the port currents  $I_i$  and voltages  $V_i$ , the *incident* and *reflected* quantities,  $a_i$  and  $b_i$  respectively, are defined for each port.

The *incident* quantity,  $a_i$  as

$$a_i = \frac{v_i}{2\sqrt{R_i}} + \frac{\sqrt{R_i}}{2}I_i$$

The *reflected* quantity,  $b_i$  as

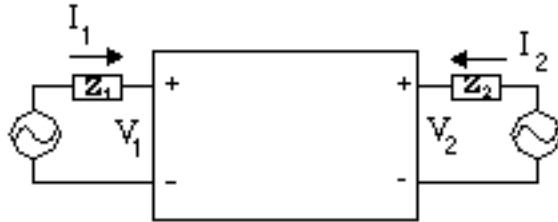
$$b_i = \frac{v_i}{2\sqrt{R_i}} - \frac{\sqrt{R_i}}{2}I_i$$

The frequency-dependent S-parameter matrix relates  $a$  and  $b$  by

$$b(\omega) = S(\omega)a(\omega)$$

These definitions are possible because for LTI systems, sources at a frequency  $\omega$  generate steady-state responses, and therefore outputs, at the same frequency.

**Figure L-1 Two-Port Linear Network**



## Frequency Translating S-Parameters

The Spectre RF small-signal analyses treat the circuit as linear time-varying (LTV). The primary difference between linear time-varying networks, those that come from circuits with a time-varying operating point, and LTI networks, those that come from circuits with a DC operating point, is that LTV networks shift signals in frequency.

For periodically linear time-varying (PLTV) systems, inputs at a frequency  $\omega$  may generate circuit responses, and therefore outputs, at the frequencies  $\omega + n\omega_0$ , where  $\omega_0$  is the fundamental frequency and  $n$  is a (signed) integer. We can adopt the S-parameter concept to PLTV systems by considering the inputs and outputs generated at the sidebands of each harmonic to be *virtual ports* of a generalized linear system.

That is, we can define  $a_{i,n}$  by

$$a_{i,n}(\omega) = \frac{v_i(\omega + n\omega_0)}{2\sqrt{R_i}} + \frac{\sqrt{R_i}}{2} I_i(\omega + n\omega_0)$$

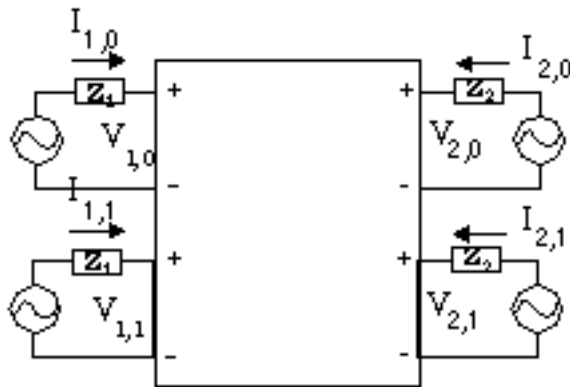
And we can define  $b_{i,n}$  by

$$b_{i,n}(\omega) = \frac{v_i(\omega + n\omega_0)}{2\sqrt{R_i}} - \frac{\sqrt{R_i}}{2} I_i(\omega + n\omega_0)$$

where the integer  $n$  represents an harmonic index.

For example, as shown in [Figure L-2](#) on page 1080, an ideal mixer may be represented as a four-port.

**Figure L-2 Ideal Mixer Represented as a Four-Port**



Note that each *virtual port* of a given *physical port* has the same reference impedance. The periodic S-parameter matrix is the  $4 \times 4$  matrix  $\tilde{S}$  that relates the extended vectors

$$\tilde{b} = [b_{1,0} \ b_{1,1} \ b_{2,0} \ b_{2,1}]^T, \tilde{a} = [a_{1,0} \ a_{1,1} \ a_{2,0} \ a_{2,1}]^T$$

by

$$\tilde{b}(\omega) = \tilde{S}(\omega)\tilde{a}(\omega)$$

For example, consider an upconverting mixer. You might write  $S(2,1|1,0)$  to represent the signal generated at port #2 (typically the output) on the upper sideband of harmonic +1 by an incident signal at port #1 (typically the input) at harmonic zero (baseband).  $[S(2,1|1,0)(\omega)]^2$  would represent the power gain from baseband to RF at the baseband-referenced frequency  $\omega$ . In the PSP results generated by spectre,  $S(2,1|1,0)$  is accessible as S21~1:0.

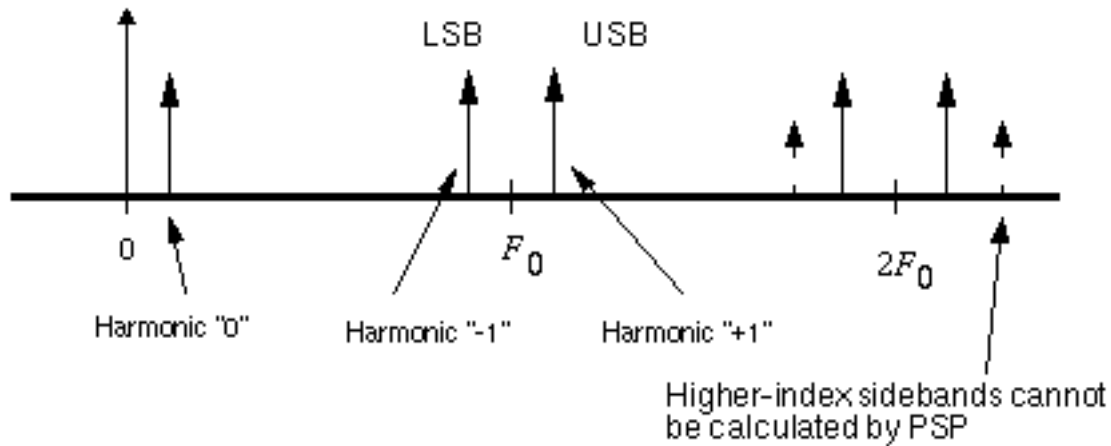
Note that because *multiple virtual ports* are used as both inputs and outputs in PSP analysis, PSP analysis must follow an absolute indexing scheme for the small-signal responses. This is different from the relative indexing scheme used in PXF, PAC, and PSP analyses in releases 4.4.5 and earlier. See [“Harmonics and Sidebands in PSP, PAC, PXF, and Pnoise Analyses”](#) on page 1113 for a discussion of the differences.



## Upper and Lower Sidebands

Each harmonic may have an input, or response, at both the upper and lower sidebands of each harmonic. In PSP analysis, the upper sideband is denoted by a positive integer, and the lower sideband as a negative integer. See [Figure L-3](#) on page 1081.

**Figure L-3 Harmonics, Sidebands, and Virtual Ports**



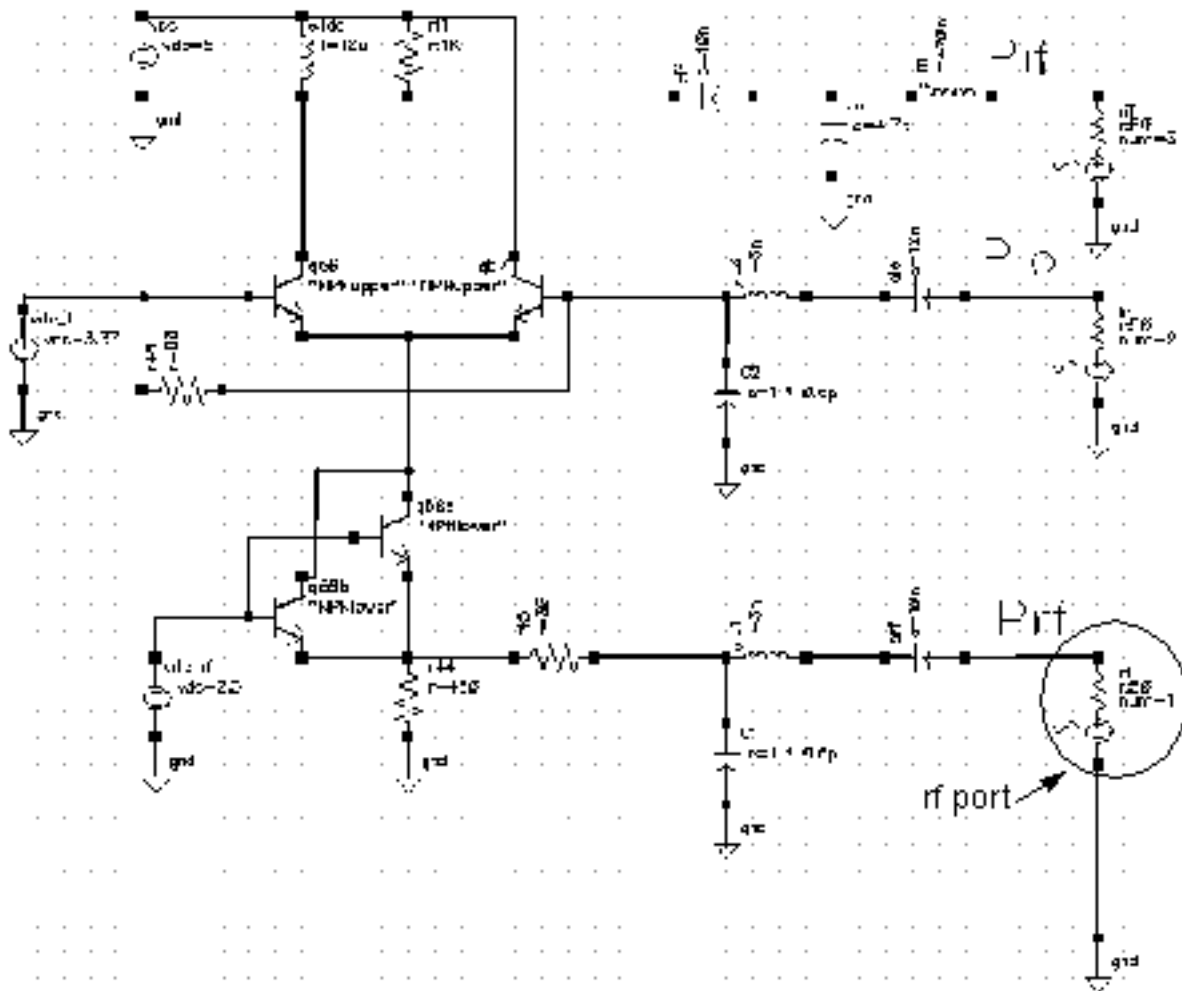
For a small-signal frequency  $\Delta F$ , the upper sideband of the  $k^{\text{th}}$  harmonic is at the frequency  $|k| F_0 + \Delta F$  and the lower sideband is at  $|k| F_0 - \Delta F$ .

In general, an input (perhaps at baseband at the frequency  $\Delta F$  in an upconversion mixer) can generate responses at all the frequencies  $|k| F_0 + l\Delta F$ , for  $k$  and  $l$  integers. However, since small-signal analyses are linear, they can only calculate the signals at the first sideband. Thus we only need notation for the  $l = 1$  terms in PSP analysis.

### PSP Analysis Example

Consider performing a PSP analysis on the *NE600* mixer schematic from the *rfExamples* library. The schematic is shown in [Figure L-4](#) on page 1082.

Figure L-4 NE600 Mixer



Suppose the RF input signal is at 900 MHz, the LO at 1 GHz, and the IF at 100 MHz. Before the PSP analysis is performed, a PSS analysis must be run. For small-signal analysis, in many cases it would be sufficient to treat the RF input as small-signal (for example, by setting the *source type* to *DC*). However, sometimes it is important to analyze additional noise folding terms induced by the RF input, so in this example we assume the RF source is a large signal (e.g., *source type* = *sine*). The PSS fundamental need to be set to 100 MHz.

Now for the sake of demonstration suppose we wish to perform the small-signal analysis from 20 MHz below the RF center frequency to 30 MHz above. To set up the analysis, we first select a *frequency sweep*. We select *sweeptype=relative*, with a range of -20 MHz to 30 MHz. This accounts for inputs on the RF port in the range of 880 MHz to 930 MHz. Noise

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

parameters such as *noise figure* are computed in a 50 MHz band around the frequency specified by the output harmonic.

Next we select the ports and harmonics. The *input* and *output ports* are selected from the schematic as always. Selecting the harmonics is somewhat trickier.

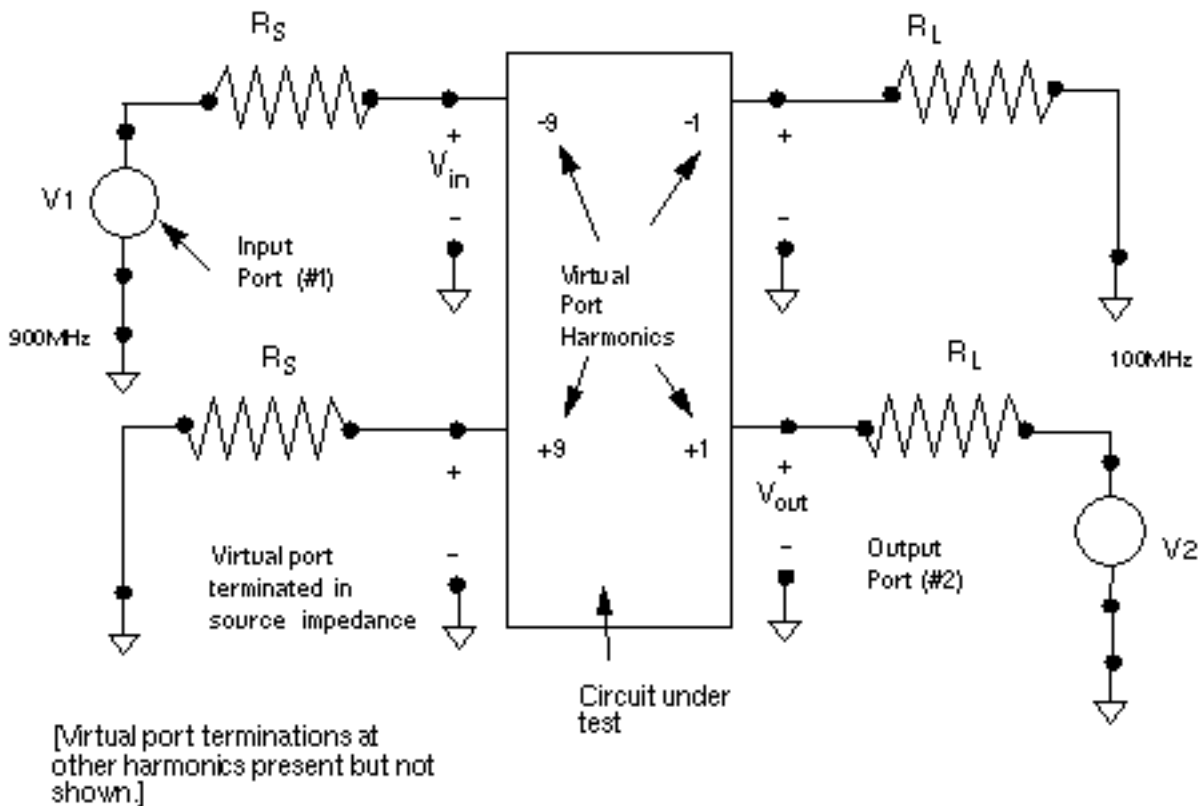
With the 1 GHz LO, small-signal inputs at around 900 MHz, or harmonics  $\pm 9$  of the PSS fundamental, appear at around 100 MHz, or harmonics  $\mp 1$  of the fundamental. We may select harmonic 1 as the output harmonic and harmonic -9 as the input harmonic because a single complex-exponential input

$$e^{i\omega_s t}$$

input on the lower side of 900 MHz (harmonic -9) appears as the upper sideband of harmonic 1 (around 100 MHz). -9 and 1 are separated by 10 fundamental periods, which corresponds to the LO frequency of 1 GHz. [Figure L-5](#) on page 1084 conceptually illustrates this setup.

Once the simulation is complete, a limited set of data is accessible through direct plot, and the full data set is accessible with the results browser. The default is to output all quantities versus the input frequency, which in this case would be a sweep from -920 MHz to -870 MHz, because the input harmonic is -9 and the sweep ran from -20 MHz to 30 MHz. The *freqaxis* parameter on the options form may be used to change the axes that are output by Spectre RF. In 4.4.5 the *freqaxis* parameter and *frequency sweep* specifications are solely responsible for the data's axis generation. Setting *freqaxis=out* in this example would produce an axis running from 80 MHz to 130 MHz.

**Figure L-5 Measuring Periodic S-Parameters**



An additional port/harmonic pair can be included in the PSP analysis by using the *auxiliary port* fields.

If it is desirable to include more than three harmonics in the PSP analysis, they can be added to the list in the form below the *input./output/auxiliary*. For example, to examine additional images in the PSP analysis, +9 and -1 could be additional harmonics. S21~9:-1 would represent the transducer gain from RF-USB to IF-LSB. Note that S21:9:1 is likely to be small, because this term represents a frequency shift of 800 MHz. There are no elements in the circuit that vary at 800 MHz, so significant 800 MHz frequency translations are not present.

### Noise and Noise Parameters

## Calculating Noise in Linear Time-Invariant (DC Bias) Circuits

The standard Noise analysis has two parts.

First, the circuit is analyzed without the noise sources present in order to find a DC operating point.

Next, the circuit is linearized around that operating point and the noise sources are turned on.

The linearized circuit is used to compute a set of transfer functions that represent the gain from each noise source to the node pair or probe that is identified as the output for the Noise analysis. All noise generators present in the circuit are automatically included in the Noise analysis, as are the noise sources of the source and load.

## Calculating Noise in Time-Varying (Periodic Bias) Circuits

Noise analysis in RF circuits, where the circuit operating point is time-varying, is computationally involved, but conceptually similar to Noise analysis for circuits with a DC bias.

Find the circuit operating point using the PSS analysis.

Linearize the circuit around that operating point.

Use the PXF analysis, based on the linearized circuit, to compute a set of transfer functions that relate the noise sources to the noise at the circuit output.

The treatment of the sources and the transfer functions is more complicated for RF circuits because of the time-varying operating point.

For noise sources that are bias dependent, such as shot noise sources, the time-varying operating point acts to modulate the noise sources. Active elements with a time-varying bias point can convert noise from one frequency to another, a process known as *noise folding*, regardless of the origin of the noise.

Because of these effects, noise generated in RF circuits usually has *cyclostationary* properties. Cyclostationary random processes are processes whose statistical properties are periodically time-varying. In the frequency domain, a simple way to think of cyclostationarity is as frequency-correlation. For example, noise at the input of a mixer appears on the output, but shifted in frequency. Thus the noise at the mixer output at a given frequency is correlated with noise at the mixer input at a frequency separated by the frequency of the local oscillator. In contrast, the noise generated by a circuit with a DC bias point is usually modeled as being uncorrelated with noise at any other frequency. Spectre RF correctly accounts for cyclostationary statistics when calculating noise, noise figure, noise correlation parameters, and equivalent noise sources.

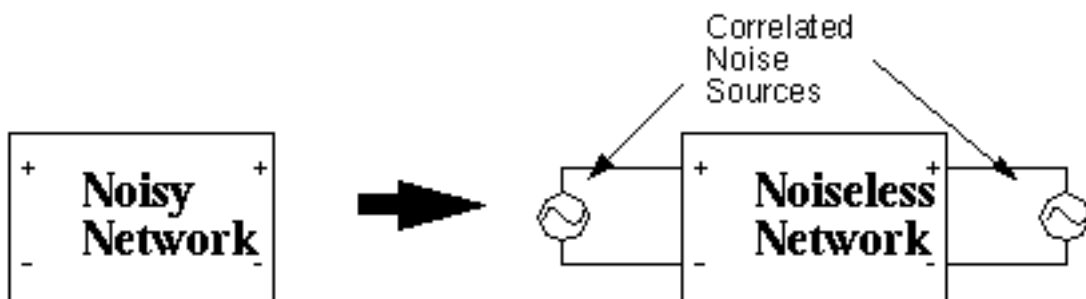
### The maxsideband Parameter

All the noise computations in PSP involve noise folding effects. The *maxsideband* parameter specifies the maximum sideband included for summing noise contributions either up-converted or down-converted to the output at the frequency of interest. The contribution of the noise source to the output is modulated by the periodic transfer function. Modulation with a periodic transfer function is convolution with the discrete spectrum of the transfer function. *Maxsideband* specifies the number of sidebands to be involved in this calculation.

## Noise Correlation Matrices and Equivalent Noise Sources

Noise correlation matrices represent a decomposition of a linear circuit into a noiseless linear network and correlated noise sources. For example in [Figure L-6](#) on page 1086, a noisy linear network is decomposed into a noiseless network and equivalent noise current sources, one for each circuit port, that represent the effect of all the noise generators internal to the original network. Note that in general the equivalent noise sources are correlated. The equivalent sources can be completely described by a (frequency-dependent) noise correlation matrix. Note that once a noise correlation matrix along with an admittance, impedance, or S-parameter matrix are known, then the properties of the noisy linear network as seen from the I/O ports is completely specified. All circuit input/output properties—gain, noise figure, etc.—can be calculated from the S-parameter and noise correlation matrices.

**Figure L-6 Decomposing a Network Into Noiseless and Noisy Elements.**



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

The SP analysis computes noise correlation parameters in the admittance representation, i.e., the sources in [Figure L-6](#) on page 1086 are current sources. If we let  $I_N$  denote the vector of equivalent noise currents as

$$I_N = \begin{bmatrix} I_{N1} \\ I_{N2} \end{bmatrix}$$

Then the noise correlation matrixes  $C_{Y11}$  and  $C_{Y12}$  are defined as

$$C_{Y1x} = \left[ \frac{1}{4k \times T0\_1 \times df} \right] \times E \left\{ I_N I_N^H \right\}$$

And the noise correlation matrixes  $C_{Y21}$  and  $C_{Y22}$  are defined as

$$C_{Y2x} = \left[ \frac{1}{4k \times T0\_2 \times df} \right] \times E \left\{ I_N I_N^H \right\}$$

where

$k$  is Boltzmann's constant

$T0\_1$  is the noise temperature of the input port

$T0\_2$  is the noise temperature of the output port

$df$  is noise bandwidth

superscript  $H$  ( $I^H$ ) denotes the Hermitian transpose

$E\{\}$  denotes statistical expectation.

For example, for a two-port,

$$C_Y = \left[ \frac{1}{4k \times T0\_1 \times df} \right] \times E \left\{ \begin{bmatrix} I_{N1} \overline{I_{N1}} & I_{N1} \overline{I_{N2}} \\ I_{N2} \overline{I_{N1}} & I_{N2} \overline{I_{N2}} \end{bmatrix} \right\}$$

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

where the overbar signifies complex-conjugate. Note that this matrix must have real diagonal elements, because the diagonals represent a total noise power, but the off-diagonals, which represent the correlations, may be complex.

In the periodic case, we define the periodic admittance noise correlation matrix in precisely the same way. The situation is slightly more complicated because the *virtual ports* may lie at different frequencies in RF systems.

Letting

$$\tilde{I}_N$$

denote the extended vectors of noise currents at each of the virtual ports (each virtual port consisting of a physical port. harmonic pair), for example

$$\tilde{I}_N = [I_{1,0} \ I_{1,1} \ I_{2,0} \ I_{2,1}]^T$$

where the first index indexes the physical port, and the second index specifies the harmonics.

The periodic admittance noise correlation matrix

$$\tilde{C}_Y$$

is defined as

$$\tilde{C}_Y = \left[ \frac{1}{4k \times T0\_1 \times df} \right] \times E \left\{ \tilde{I}_N \tilde{I}_N^H \right\}$$

When you specify the PSP analysis option *donoise=yes*, then the complex noise correlation matrix of order (#active ports X #active sidebands) is computed.

#### Two-Port Noise Parameters

As an alternative to the noise correlation matrices that define the equivalent sources, Spectre RF calculates the values of the equivalent noise parameters  $F_{min}$ ,  $R_n$ ,  $G_{opt}$ ,  $B_{opt}$ , and  $NF_{min}$ . These are calculated from the two-port admittance ( $Y_{11}$ ,  $Y_{12}$ ,  $Y_{21}$ ,  $Y_{22}$ ) and noise correlation admittance ( $CY_{11}$ ,  $CY_{12}$ ,  $CY_{21}$ ,  $CY_{22}$ ) parameters. These calculations are done as part of the



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

SP and PSP analysis. In terms of the admittance and noise correlation matrices, the parameters are

$$F_{min} = 1 + 2 \frac{CY22}{|Y21|^2} (G_{opt} + \text{Re}\{Y11 - Y21(CY12/CY22)\})$$

$$G_{opt} = \sqrt{(|Y21|^2 (CY11/CY22) - |Y21|^2 (|CY12|^2/CY222) + (\text{Re}\{Y11 - Y21(C12/C22)\})^2)}$$

$$B_{opt} = -\text{Im}\{Y11 - Y21(CY12/CY22)\}$$

$$R_n = CY22/|Y21|^2$$

$$Y_{opt} = G_{opt} + jB_{opt}$$

$$NF_{min} = 10\log(F_{min})$$

$Y_{opt}$  is the source admittance that gives the minimal noise factor  $F_{min}$  (corresponding to the source reflection coefficient  $\Gamma_{opt}$ , or  $\Gamma_{opt}$ ) and  $R_n$  is the equivalent noise resistance. Finally,  $NF_{min}$ , the minimum noise figure, is  $F_{min}$ , the minimum noise factor, in dB.

For more information, see Janusz A. Dobrowolski, *Introduction to Computer Methods for Microwave Circuit Analysis and Design*, Artech House, Boston, 1991, page 193.

### Noise Circles

Noise factor is a function of the source admittance  $Y_s = G_s + jB_s$ . For a given  $Y_s$  the noise factor is

$$F = F_{min} + \frac{R_n}{G_s} (Y_s - Y_{opt})^2$$

Varying  $Y_s$  traces out circles of constant noise factor  $F$ . In the 4.4.5 release, noise circles are only available in direct plot for the SP analysis.

## Noise Figure

### Performing Noise Figure Computations

The Noise, SP, Pnoise and PSP analyses all provide the ability to calculate various types of noise figure. Noise and SP analyses are used for circuits with a DC bias. Pnoise and PSP analyses are used for circuits with a periodic bias. The *generic* way Spectre RF calculates the noise figure is by first computing the noise factor  $F$ ,

$$F = \frac{\text{totalOutputNoise} - \text{outputNoiseFromLoad}}{\text{outputNoiseFromSource}}$$

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

where the noise is specified in units of power (e.g.,  $V^2/Hz$ ). Noise figure is then  $NF=10\log_{10}F$ .

The various definitions of noise figure differ in the following ways

- How the contributions to the total output noise are calculated (e.g. Pnoise analysis has noise folding effects, Noise analysis does not)
- What noise is considered to be due to the load
- What noise is considered to be due to the source

All the analyses share some common rules that must be followed to obtain correct answers

- You must specify a *port* (not a *vsource*, *isource*, or *ahdl* source) as an circuit *input* or *iprobe*.
- You must specify the *load* as an *output* or *oprobe*. The *load* may be a *resistor* or a *port*. Note that all noise from the source is included in the denominator of the noise factor fraction, including excess noise, so do not specify excess noise on the input port. (Excess noise is specified with the *noisefile* or *noisevec* option.)

In rare cases there may be no load, in which case you can specify the output using a pair of nodes. Be warned, however, that if there is a load in the circuit, and a pair of nodes is specified as an output, then you obtain different results than if a load was specified as output. This is because the load contributes some noise to the total output noise that must be subtracted out before using the equation above to compute the noise figure. If only a pair of nodes is used, Spectre RF has no way of determining which of the elements in the circuit is the load (there could be multiple resistors connected to the output nodes, for example) and so cannot determine the amount of output noise due to the load.

Note that these requirements are automatically enforced in SP and PSP analyses, since the input and output sources are always ports that must be identified to the analyses.

## Noise Figure From Noise and SP Analyses

The Noise and SP analyses perform noise figure computations on circuits with a DC operating point. The above prescription for noise figure computation is straightforward: the output noise, contribution from source, and contribution from load must be computed. Mathematically, if we let  $X_L$  denote the transfer function from output load to output, (at the same frequency) and  $X_S$  denote the transfer function from input source to output then

$$F(f) = \frac{N_o(f) - |X_L|^2 n_L(f)}{|X_S|^2 n_s(f)}$$

In this equation,  $X_S$  plays a role similar to transducer gain in traditional treatments of noise figure.

## Pnoise (SSB) Noise Figure

Because noise in an RF circuit can originate at many different frequencies, the denominator in the noise factor computation is in a sense ill-defined. See the book *Microwave Mixers* by S. Maas for a discussion of various possible noise figure definitions.

There are three common noise factor definitions in use. The Spectre RF Pnoise and PSP analyses compute as  $F$  or  $NF$  what is referred to as conventional single-sideband (SSB) noise figure. The conventional SSB noise figure is typically useful for heterodyne receivers. To compute the conventional SSB noise figure, a reference sideband must be specified that identifies the input noise used in the denominator of the noise factor computation. Only the contribution of the noise from the input source, generated at the frequency specified by the reference sideband, is included in the noise factor denominator.

In the Pnoise context, the numerator contains the total output noise, except the noise from the output load that was generated at the output frequency. Note in particular that noise from the input source folded from all sidebands, and noise from the output load folded from the non-zero sidebands, is included in the noise factor numerator. Mathematically, Pnoise computes conventional single-sideband noise factor as

$$F_{ssb}(f_{out}) = \frac{N_o(f_{out}) - |X_L^{(0)}|^2 n_L(f_{out})}{|X_S^{(K_{ref})}|^2 n_S(f_{out} + K_{ref}f_0)}$$

where

- $f_{out}$  is the output frequency swept by Pnoise
- $X_L^{(0)}$  is the transfer function associated with the zero sideband, from load to output
- $X_S^{(K_{ref})}$  is the transfer function associated with the reference sideband, from source to the output
- $n_L(f_{out})$  is noise generated by load at the output frequency
- $n_S(f_{out} + K_{ref}f_0)$  is the noise generated by the source at the input frequency.

In the PSP analysis context, the noise factor denominator includes only noise from the input harmonic. The numerator contains all output noise, except noise from the output load at the output harmonic. (Refer to [“Harmonics and Sidebands in PSP, PAC, PXF, and Pnoise](#)

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

Analyses" on page 1113 for a discussion of differences in frequency indexing in PXF and PSP analyses.)

To be mathematically precise in what follows, let

- $x_L^{(K)}$  denote the transfer function from output load,  $k^{th}$  harmonic to output
- $x_S^{(K)}$  denote the transfer function from input source,  $k^{th}$  harmonic to output
- $K_o$  denote the output harmonic
- $K_i$  denote the input harmonic

The conventional single-sideband noise factor is computed by PSP as

$$F_{ssb}(f_{out}) = \frac{N_o(K_o f_0 + f) - \left| \frac{x_L^{(K_o)}}{x_S^{(K_i)}} \right|^2 n_L(K_o f_0 + f)}{\left| \frac{x_L^{(K_o)}}{x_S^{(K_i)}} \right|^2 n_S(K_i f_0 + f)}$$

where

- $f$  is the PSP relative sweep frequency
- $K_o f_0 + f$  is the output frequency
- $K_i f_0 + f$  is the input frequency
- $n_L(K_o f_0 + f)$  is the noise generated by the load at the output frequency
- $n_S(K_i f_0 + f)$  is the noise generated by the source at the input frequency

The conventional SSB noise figures computed by PSP and Pnoise analyses are the same and are computed in the same way internally, it is only the notation above that is different.

## DSB Noise Figure

In some applications, such as direct conversion receivers, it is more appropriate to compute what is called double-sideband (DSB) noise figure. Double-sideband noise factor is obtained by ratioing the same numerator as for SSB to the noise from the input at the input harmonic as well as its primary image.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

Double-sideband noise figure is usually 3 dB below single-sideband noise figure, except when the input signal band is converted to baseband output with DSB noise figure equal to SSB noise figure.

In double-sideband computation, the input signal band is assumed to be either down-converted or up-converted to the output signal band. Hence you should associate the appropriate harmonic number to the input and the output port in the *portharmsvec* parameter. For a mixer, their difference is the LO band. The image sideband is the sideband on the other side of the LO band. The distance from the image band to the LO band is the same as that from the LO band to the input band. For example, if the fundamental frequency is 100 MHz, the LO frequency is 1 GHz, and the RF input frequency is 900 MHz, then the LO band is 10, the RF input band is 9 and the image band is 11.

Double-sideband noise figure is computed by as

$$F_{dsb}(f_{out}) = \frac{N_o(K_o f_0 + f) - \left| X_L^{(K_o)} \right|^2 n_L(K_o f_0 + f)}{\left| X_S^{(K_i)} \right|^2 n_S(K_i f_0 + f) + \left| X_S^{(K_{image})} \right|^2 n_S(K_{image} f_0 + f)}$$

where

$n_S(K_{image} f_0 + f)$  is the noise generated by source at the image input frequency obtained according to the above description.

All other quantities are the same as those in  $F_{ssb}$ . Note that both the PSP and Pnoise analyses compute double-sideband noise figure.

## IEEE Noise Figure

Sometimes it is desirable to define noise figure quantities where we assume that the noise from input images that are potentially filtered is not present. The IEEE definition of noise figure in mixers differs from the conventional definition in that it does not include the contribution to the output noise from the image sideband in the numerator of the noise factor. Spectre RF eliminates all image harmonics/sidebands from the output noise in the noise factor numerator when computing  $F_{ieee}$ . Using the above notation,  $F_{ieee}$  is

$$F_{ieee} = \frac{N_o(K_o f_0 + f) - \left| X_L^{(K_o)} \right|^2 n_L(K_o f_0 + f) - \sum_{K \neq K_i} \left| X_S^{(K)} \right|^2 n_S(K f_0 + f)}{\left| X_S^{(K_i)} \right|^2 n_S(K_i f_0 + f)}$$

# Virtuoso Spectre Circuit Simulator RF Analysis User Guide

## Using PSP and Pnoise Analyses

Note that both the PSP and Pnoise analyses compute *F<sub>ieee</sub>*.

[Figure L-7](#) on page 1094, [Figure L-8](#) on page 1095, and [Figure L-9](#) on page 1096 summarize the treatment of the input source and output load for the various noise figure definitions.

**Figure L-7 Input Source Treatment for Denominator in Noise Factor Computations**

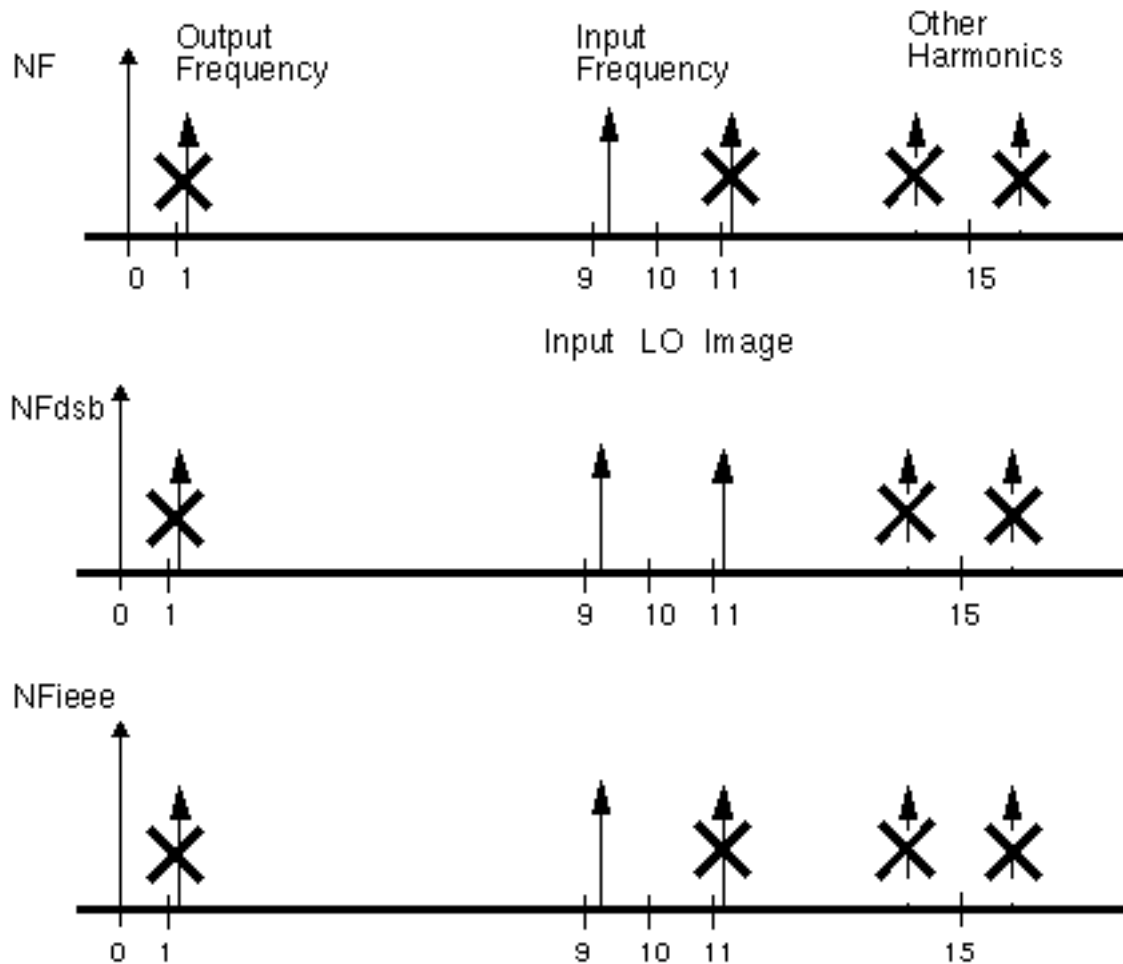


Figure L-8 Output Load Treatment for Numerator in Noise Factor and Noise Figure Computations

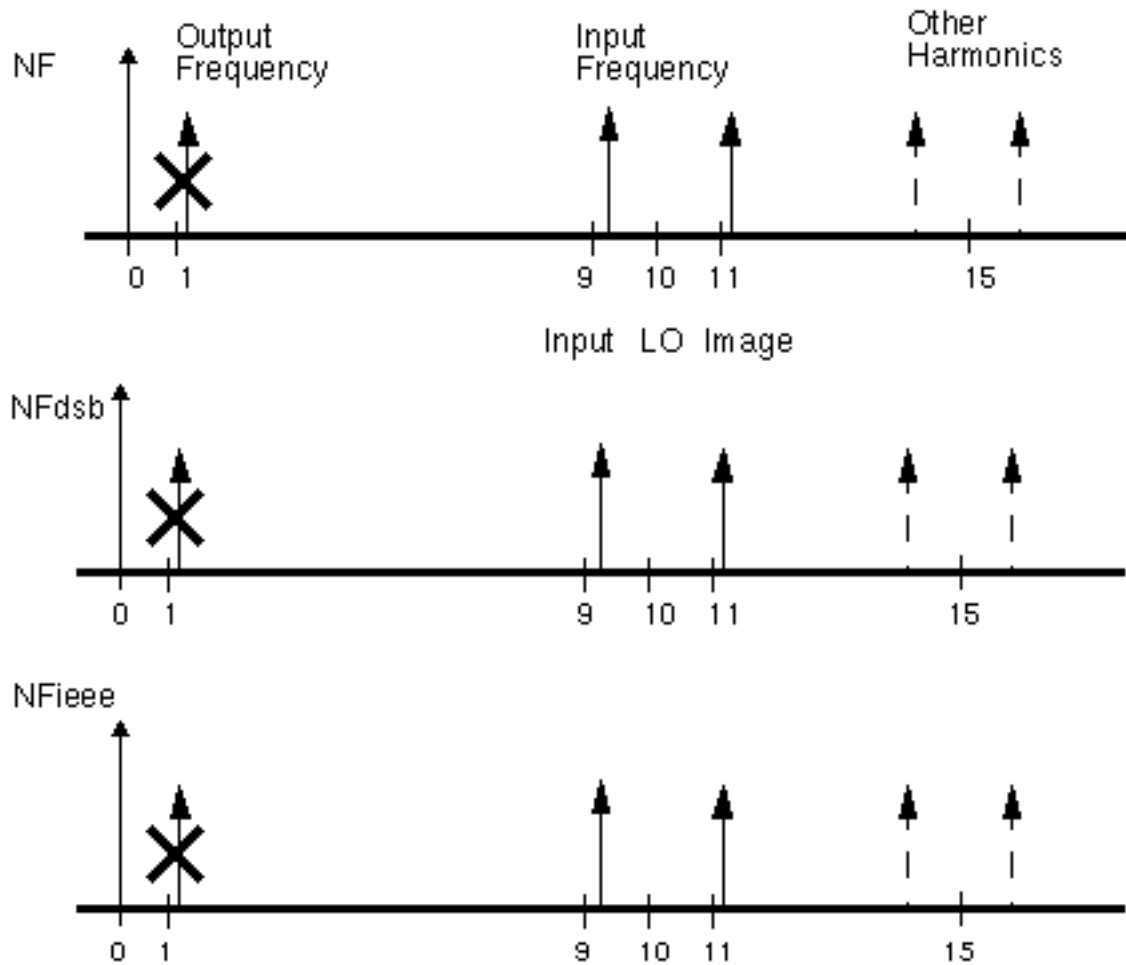
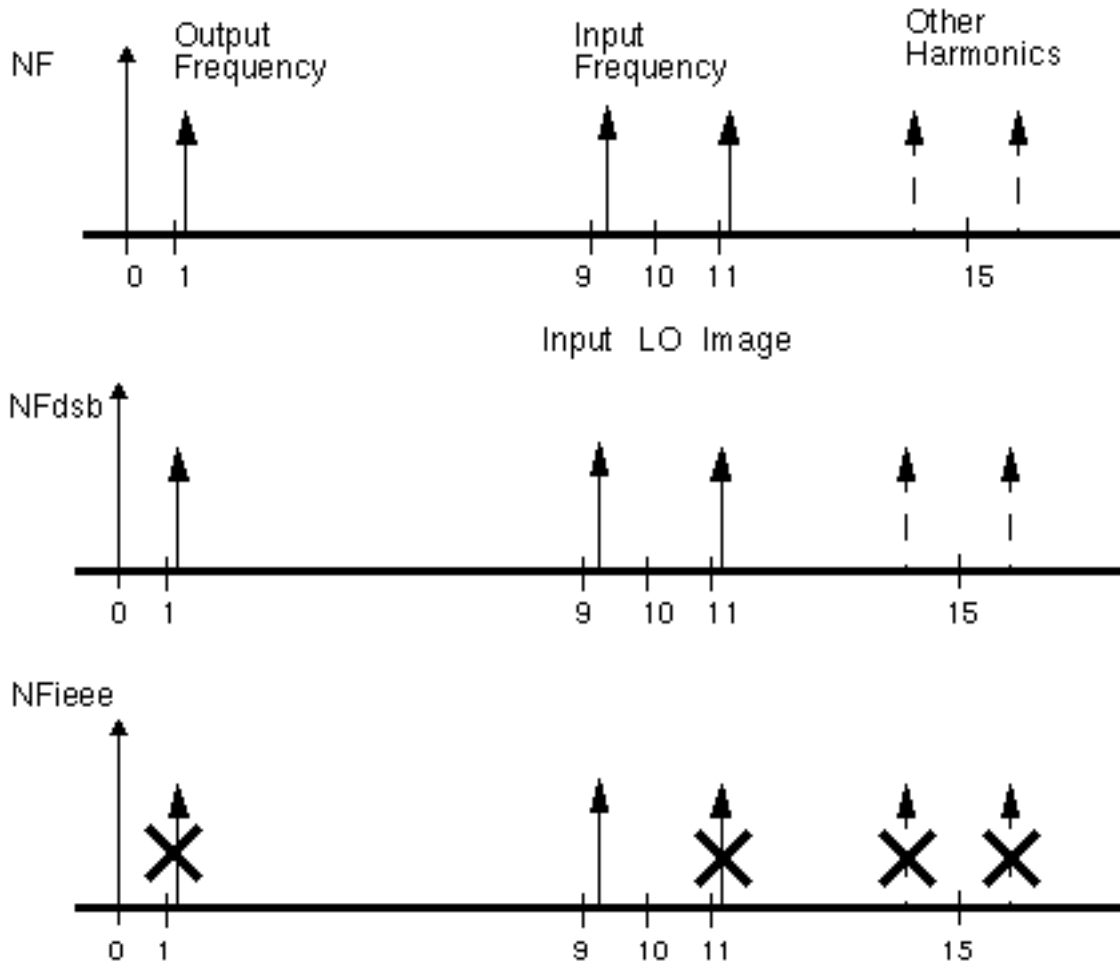


Figure L-9 Input Source Treatment for Numerator in Noise Factor Computations



## Noise Computation Example

Now consider performing noise computations as part of the PSP analysis of the *ne600* mixer example presented earlier (See “PSP Analysis Example” on page 1081).

To compute noise parameters, set *donoise=yes* on the PSP analysis form and select a reasonable number for *maxsideband*. In this case, *maxsideband* should certainly be greater than 10 and probably greater than 50 or so. Setting *maxsideband=50* would account for noise folding from up to 5 GHz in frequency. The simulation can now be run as before. In



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

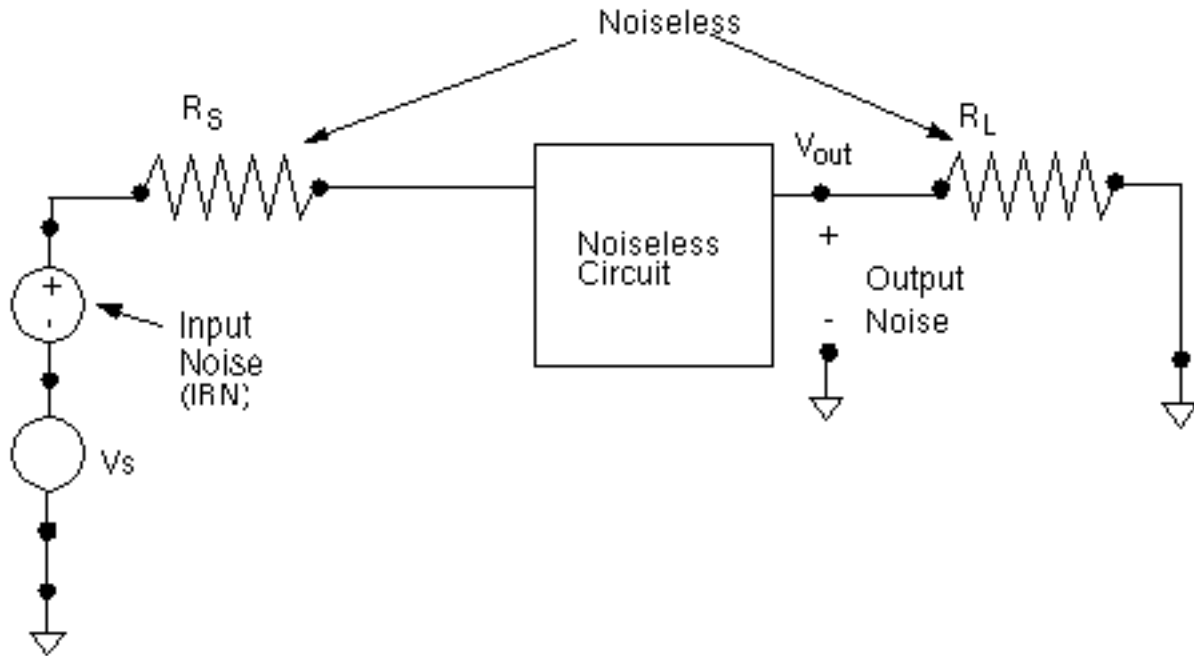
In addition to computing the periodic S-parameters, the periodic noise correlation matrices are also computed, as are noise figure and the equivalent noise parameters.

Recall that the output axes for PSP analysis are set by the *freqaxis* option and are shared by all quantities calculated by PSP analysis. This can seem odd in the case of noise figure, where the output noise is actually analyzed for the frequency range of 80 MHz to 130 MHz. This is because the output harmonic is 1, representing a center frequency of 100 MHz for the relative sweep, and the sweep ran from -20 MHz to 30 MHz. Setting *freqaxis=out* in this example would produce an axis running from 80 MHz to 130 MHz.

### Input Referred Noise

For a given output noise spectrum, the equivalent input noise, *input referred noise* or IRN, is the noise that, if it were generated by the circuit element specified as input, would produce the same output noise. This assumes that the circuit loading conditions, etc., are unchanged. Note that the equivalent input noise includes noise from the source and load if they are noisy in the original circuit. For example, consider a circuit with a voltage source as input, as in [Figure L-10](#) on page 1098. If a voltage source inserted in series with the input source generates the input referred noise, and the rest of the circuit is noiseless, the noise observed at the output is the same as in the original noisy circuit.

Figure L-10 Equivalent Input Noise



## Using Input Referred Noise

Input referred noise, or IRN, gives a direct estimate of how much the noise in a circuit corrupts signals passing through, since the amplitude of the noise can be directly compared to the amplitude of signals on the input. In principle, it can also be used to build macromodels of the circuit. If a source of the same type (`vs`, `is`, or `port`) has the `noise` argument set to a file to which the input referred noise has been written, and all the noise elements in the circuit turned off (perhaps as may happen when replacing the circuit with an S-parameter macromodel) then the noise that appears as the circuit output is the output noise of the original circuit. Note that Spectre RF `noise` arguments are usually given in terms of power, such as  $V^2/\text{Hz}$ , whereas the simulator usually outputs the noise in units of signal amplitude (such as  $V/\sqrt{\text{Hz}}$ ).

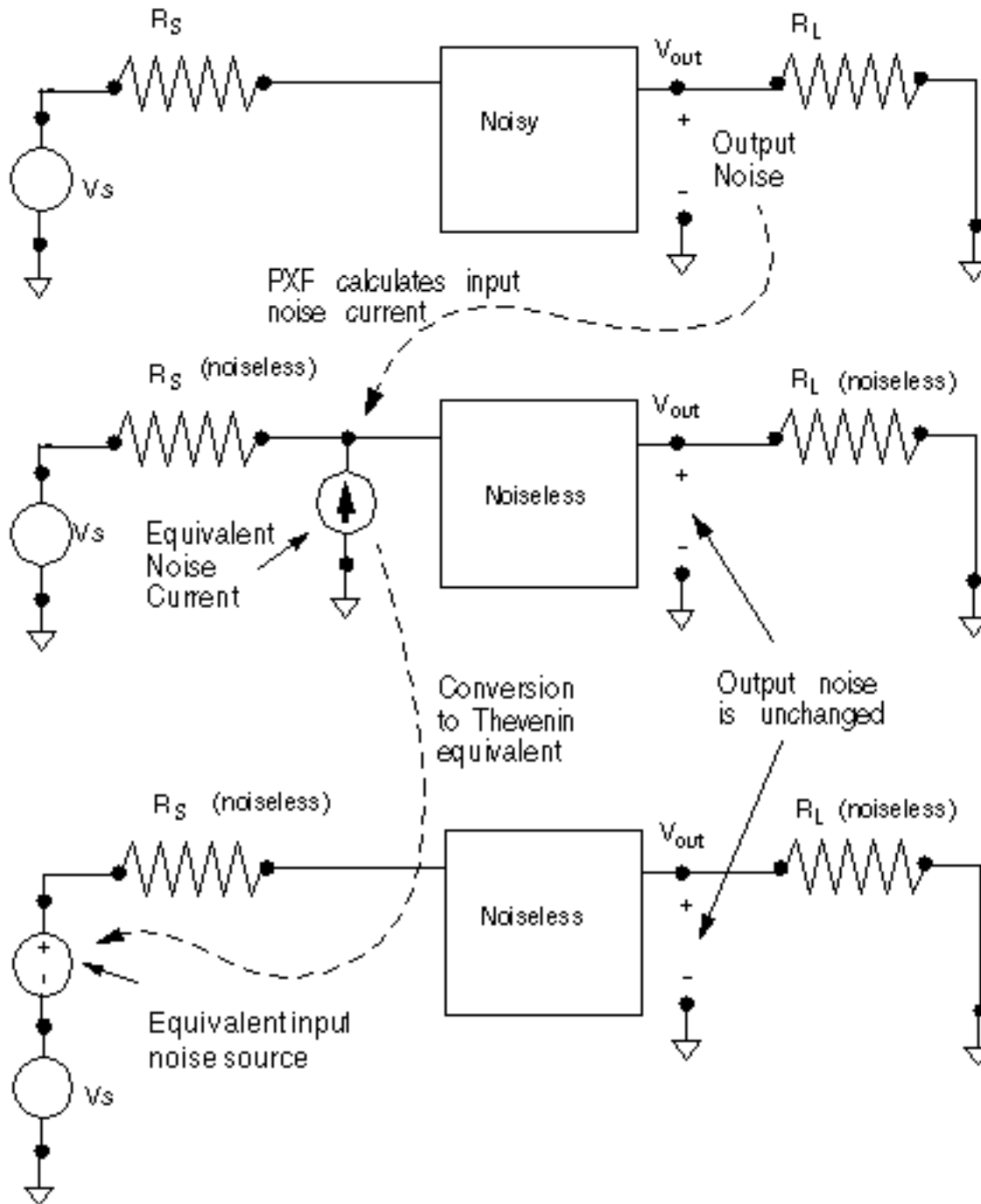
It is also possible to calculate noise figure from input referred noise, however, it is essential to be sure that any noise contribution from the load, etc., has first been properly treated. For this reason, it is recommended to use the Spectre RF built-in noise figure calculation, since the contribution of load noise, noise from other images, etc., is automatically accounted for. (See "The noise figure computed by <Procedure X> seems inaccurate, inconsistent, or just plain wrong." on page 1109 for more information.)

## How IRN is Calculated

Spectre RF calculates input referred noise from the results of the Noise or Pnoise analysis and an XF or PXF analysis. The XF or PXF analysis needed is fortunately the same analysis needed to compute the noise.

As an example of an input noise calculation, consider [Figure L-11](#) on page 1100, which shows the calculation of the equivalent input noise when a port component drives the circuit. First the total output noise is calculated. Note that this noise generally includes contributions from the circuit load. Next the results of the PXF analysis are used to express this noise as an equivalent current source attached to the circuit input. The amplitude of the current source is the amplitude of the output noise divided by the transfer function from an imaginary current source connected to the circuit input to the output node. Finally, the equivalent current source is converted into the equivalent input voltage noise by converting to Thevenin form.

Figure L-11 IRN Calculation



## Relation to Gain

It is common to think of the input noise as the output noise divided by the gain,

$$IRN = \frac{OutputNoise}{G}$$

where the output noise and input noise are measured in volts/sqrt(Hz). This is a true statement if the proper gain is used. The gain reported by Pnoise analysis is in fact the ratio between the output noise and the equivalent input noise as computed in the previous section. However, the gain reported by Pnoise analysis is not necessarily the gain useful for any other purposes (see [“Gain Calculations in Pnoise”](#) on page 1106).

## Referring Noise to Ports

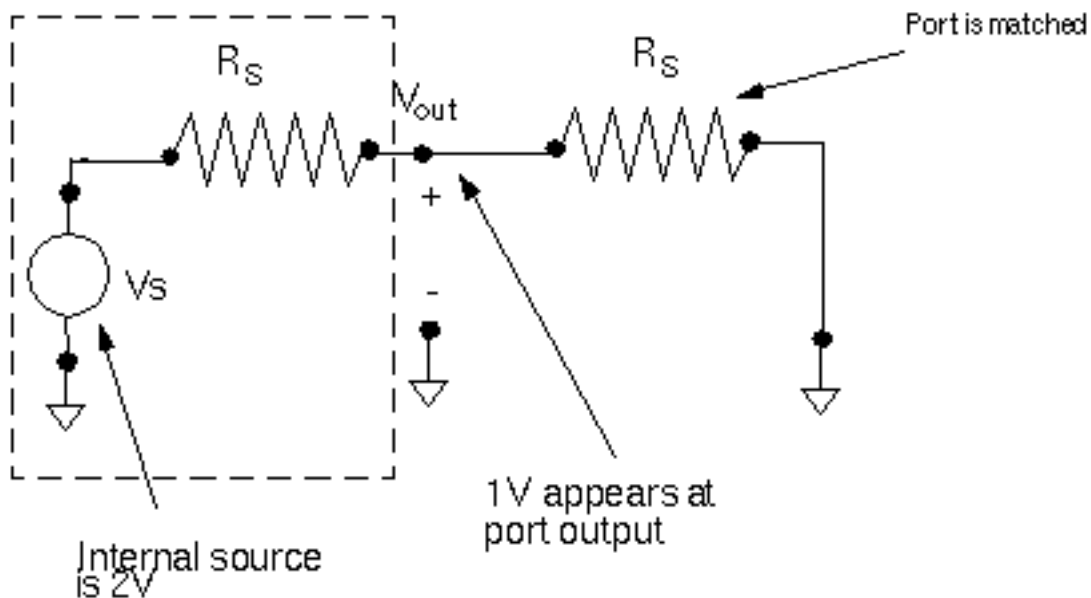
A port component in Spectre RF is a voltage source combined with a resistive impedance. Ports exist to enable easy generation of test fixtures, accurate noise figure calculations, and S-parameter calculations. Noise and S-parameter calculations in Spectre RF can (usually) only be properly performed when ports are used to drive the input and output. While the port is electrically equivalent to a source + resistor, the excitations are specified in a somewhat different way.

The port component is designed so that when a 1 V source is specified, 1 V appears at the port output when the circuit impedance is matched to the port reference impedance. The port output, however, is not the internal node that connects the voltage source and resistor, but rather the exterior resistor terminal that actually connects to the circuit under test. Thus, as shown in [Figure L-12](#) on page 1102, the voltage on the internal voltage source must be twice the voltage specified to appear at the port output.

One implication of this convention is that when a port is specified as an input in Pnoise analysis it is the noise referred to the port that is computed, not the noise referred to the internal voltage source. In other words, the input equivalent noise reported is the noise that would be fed back to spectre as a *noisefile* argument on the port to produce the same output noise. The noise referred to the port has amplitude half of the noise referred to the input voltage source.

Likewise, the gain reported by Spectre RF is the gain that would produce the equivalent input noise, referred to the port. It is half the gain from the input voltage source.

Figure L-12 Port Driving Matched Circuitry



## Gain Calculations

### Definitions of Gain

To understand gain calculations in Spectre RF it is necessary to specify which of several possible gains are being calculated.

Figure L-13 Equivalent Input Noise

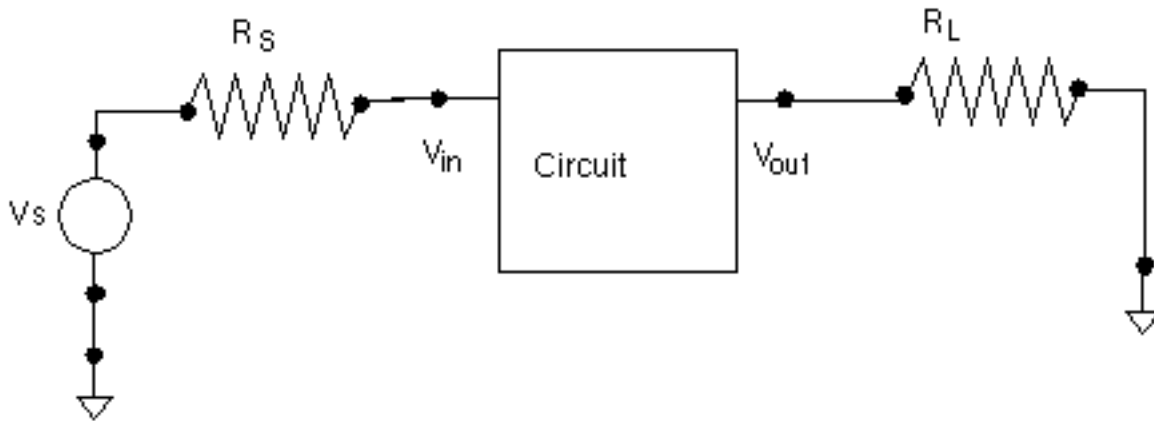


Figure L-13 on page 1103 shows a circuit configured for a gain calculation. Source and load impedances are present. These loads might represent other circuit components or *testbench* circuitry. The circuit is driven with the source  $V_s$  and the output voltage  $V_{out}$  measured. One possible gain is the gain  $G_s$ .

$$G_s = \frac{V_{out}}{V_s}$$

from the voltage source to the circuit output.

However,  $G_s$  is not the actual gain of the circuit, it is the gain of the circuit and the testbench put together. For high frequency circuits, this gain in fact cannot be measured on the bench, only in the artificial world of a circuit simulator. For these reasons, the proper way to calculate gain is the use a port component, which allows the user to specify the impedance of the driving circuitry. The Spectre RF PSP analysis computes the gain,

$$G = \frac{V_{out}}{V_{in}}$$

which is the voltage gain from the circuit input to the circuit output.

$G$  is calculated for historical reasons, and is not necessarily a useful number, but once an SP or PSP analysis is performed, several other gains can be defined and computed. Varying source and load impedance can also be considered. In the 4.4.5 release, the SP direct plot form displays various gains. The PSP direct plot form does not automatically compute the

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

various gains but they can be computed from the basic definitions using the waveform calculator.

Some gains of interest in two-port circuits are:

- $G_A$  (Available Gain)
- $G_P$  (Power Gain)
- $G_T$  (Transducer Gain)
- $G_{umx}$  (Maximum Unilateral Transducer Power Gain)
- $G_{max}$  (Maximum Available Gain)

**GA (Available Gain)**, the power gain obtained by optimally (conjugately) matching the output of the network.

$$G_A = \frac{|S_{21}|^2(1 - |\Gamma_S|^2)}{|1 - S_{11}\Gamma_S|^2(1 - |\Gamma_2|^2)}$$

where

$$\Gamma_2 = S_{22} + \frac{S_{12}S_{21}\Gamma_S}{1 - S_{11}\Gamma_S}$$

and  $\Gamma_S$  is the source reflection coefficient,

$$\Gamma_S = \frac{Z_S - Z_{S,ref}}{Z_S + Z_{S,ref}}$$

with  $Z_S$  is the source impedance and  $Z_{S,ref}$  is the reference impedance for the input port.



## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

**GP (Power Gain)**, the power gain obtained by optimally (conjugately) matching the input of the network.

$$G_P = \frac{|S_{21}|^2(1-|\Gamma_L|^2)}{|1-S_{22}\Gamma_L|^2(1-|\Gamma_1|^2)}$$

where

$$\Gamma_1 = S_{11} + \frac{S_{12}S_{21}\Gamma_L}{1-S_{22}\Gamma_L}$$

and  $\Gamma_L$  is the load reflection coefficient.

**GT (Transducer Gain)**, the ratio of the power dissipated in the load to the power available from the source,

$$G_T = \frac{(1-|\Gamma_S|^2)|S_{21}|^2(1-|\Gamma_L|^2)}{|(1-S_{11}\Gamma_S)(1-S_{22}\Gamma_L)-S_{12}S_{21}\Gamma_S\Gamma_L|^2}$$

**Gumx (Maximum Unilateral Transducer Power Gain)**

$$G_{umx} = \frac{|S_{21}|^2}{(1-|S_{11}|^2)(1-|S_{22}|^2)}$$

**Gmax (Maximum Available Gain)**, the transducer power gain when there exists a simultaneous conjugate match at both ports.

For  $Kf > 1$

$$G_{max} = \frac{|S_{21}|}{|S_{12}|} [Kf - SQRT((Kf)^2 - 1)]$$

For  $K_f < 1$

$$G_{max} = \frac{|S_{21}|}{|S_{12}|}$$

Where  $K_f$  is the stability factor,  $K_f$

$$K_f = \frac{1 - |S_{11}|^2 - |S_{22}|^2 + |D|^2}{2|S_{22}||S_{12}|}$$

and

$$D = S_{11}S_{22} - S_{21}S_{12}$$

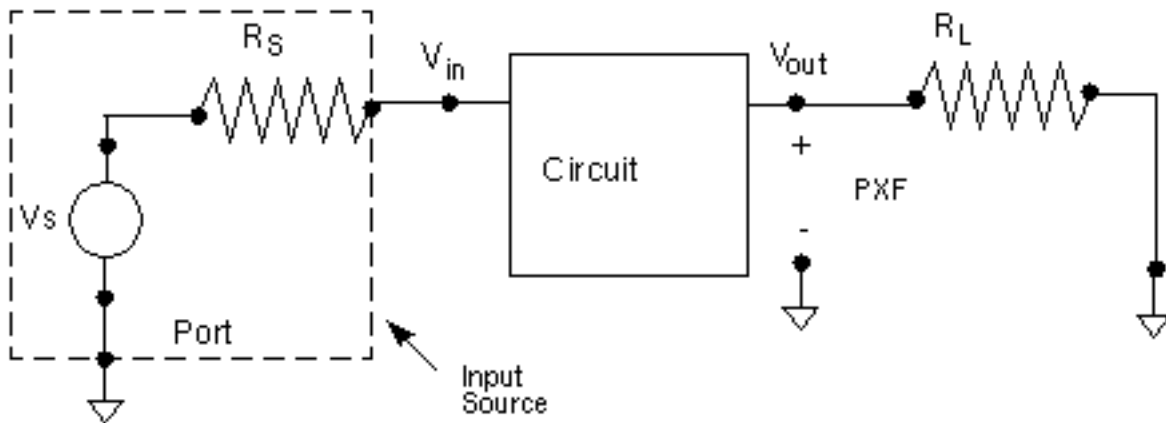
## Gain Calculations in Pnoise

When ports are used to drive the circuit input and a Pnoise analysis is performed, Pnoise analysis reports a quantity called `gain` that can be misleading. Consider the circuit in [Figure L-14](#) on page 1107 where a port component is used as the input to the circuit, and specified as such by using the `iprobe` option to Pnoise.

The goal of Pnoise analysis is usually to calculate the noise at the circuit output, as shown in [Figure L-14](#) on page 1107. Pnoise analysis also calculates an approximation to the gain  $G$  as a by-product of the input-referred noise calculation. It turns out that if it is assumed that the circuit input impedance is matched, i.e., the impedance seen by the port component shown as the dashed box in [Figure L-14](#) on page 1107, is the same as the source impedance, then the gain needed for the input-referred noise calculation is also the gain  $G$ .

However, the Pnoise analysis is based on a series of PXF analyses and when the input is not matched, then a single PXF analysis is not sufficient to calculate  $G$ . PSP analysis is needed to accurately calculate  $G$  independent of match conditions.

**Figure L-14 Pnoise with Port Component**



Coincidentally, because of the matched-input assumption made both in Pnoise analysis and in the definition of the port, the gain  $G$  from circuit input to circuit output that is reported by Pnoise analysis happens to be exactly twice the gain from the source to the output,  $G_s$ . This can be seen by noting that if the circuit input is matched, then the circuit acts as a voltage divider, and the voltage at  $V_{in}$  is

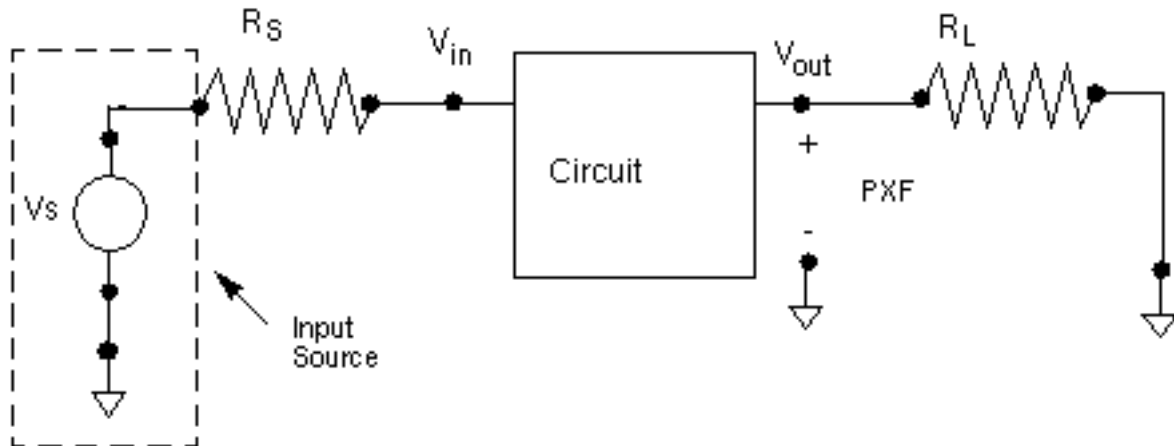
$$V_{in} = \frac{Z_{in}}{Z_{in} + R_s} V_s = \frac{R_s}{R_s + R_s} V_s = \frac{V_s}{2}$$

Thus in this special case,  $V_{out}/V_{in}$  is twice  $V_{out}/V_s$ . The *gain* reported by Pnoise is twice  $G_s$  regardless of the match conditions.

To alleviate this confusion, in future releases the *gain* number will not be reported by Pnoise analysis when the input is driven by a port.

Now consider when a separate `vsources` and resistor are used to drive the circuit, as shown in [Figure L-15](#) on page 1108. In this case, the voltage source would be specified as the input probe. Spectre RF computes the gain from the input probe to the output, which in this case is the gain from  $V_s$  to the output,  $G_s$ , since  $V_s$  was specified as the input source.

Figure L-15 Pnoise With vsrc Component



## Phase Noise

The Direct Plot form in the analog design environment plots *output noise* and *phase noise*. The phase noise form is designed for use in oscillator noise computations.

The *output noise* plots are the Total Output Noise ( $out$ ) data, taken directly from the psf files with no modification.

The *phase noise* plots show the noise power relative to the carrier power. Technically the plot is not phase noise, simply normalized output noise. However, for oscillators, close to the fundamental frequency, the noise is mostly phase noise.

The normalization is done relative to the power in the fundamental component of the noise-free oscillation as calculated by the PSS analysis. If you look at the PSS analysis results in the frequency domain, the fundamental has a particular amplitude  $V_1$ , which means that the fundamental component of the oscillation is  $V_1 \cos(2\pi f_c + j)$ , for some  $\phi$ , where  $f_c$  is the fundamental frequency.

### Defining

$$\text{noise power} = out^2,$$

where  $out$  is the *Total Output Noise*

$$\text{carrier fundamental power} = (V_1)^2/2$$

The normalized power is  $\text{normalized power} = (\text{out}^2)/[(V_1)^2/2]$

Thus the formula used by Direct Plot in Artist 4.4.3 and later to plot the normalized power (*phase noise*) in dB10 is:

$\text{normalized power in dB} = 10\log_{10}\{(\text{out}^2)/[(V_1)^2/2]\}$

For more information on phase noise, see [Appendix A, "Oscillator Noise Analysis."](#) which summarizes how to get good phase noise calculations.

## Frequently Asked Questions

**The noise figure computed by <Procedure X> seems inaccurate, inconsistent, or just plain wrong.**

The key to noise figure computation is to remember that three pieces of information are needed:

- The total noise at the output
- How much of that noise is due to the output load
- How much of that noise is due to the input source

Noise figure is the log of noise factor, and noise factor is fundamentally the total output noise less the noise due to the load ratioed to the noise due to the input. The easiest way to be sure of getting the correct noise figure is to use ports as input and output sources and loads and to have the Spectre RF simulator perform the noise figure computation.

Spectre RF correctly accounts for the noise due to the load, as well as the different possible treatments of noise from the input source that result in different types of noise figure that are used in RF circuits (customary SSB, IEEE SSB, DSB, as discussed in ["Noise Figure"](#) on page 1089). See ["Performing Noise Figure Computations"](#) on page 1089 for information on setting up noise figure computation.

Most mistakes in noise figure calculations arise when improperly accounting for items two and three in the list above. Hand computations of noise figure require that you correctly account for noise from the load and source, which can be difficult. For example, some users have tried to use the information from the noise summary table to calculate noise figure. The noise summary table reports total noise contributions, from all sidebands present in the Pnoise analysis, sorted by contributor. If the total noise reported in the noise summary is used as the numerator for the noise factor computation the wrong answer is obtained because the noise due to the load has not been subtracted off. Likewise, the noise summary table does not sort information by sideband. To compute SSB noise figure, the denominator of the noise

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

factor fraction must be only the noise from the reference sideband, not the total from all sidebands as reported in the noise summary. Noise figure calculations based on the reported input-equivalent noise, or on output noise and gain, suffer similar problems.

You can compute the noise figure by hand, but you must take care. For example, use the following steps to compute the customary single-sideband noise figure from raw data.

Perform a Pnoise analysis with the *saveallsidebands* parameter set to *yes*.

Using the results browser or direct plot, obtain the total noise at the output. Call this *OutputNoise*.

Now use the results browser to obtain the amount of total noise due to the load contributed from the zero sideband. Call this *LoadNoise*.

Finally, again using the results browser, obtain the amount of noise due to the source contributed from the reference sideband. Call this *sourcenoise*.

The noise factor is then

$$F = \frac{\text{OutputNoise} - \text{LoadNoise}}{\text{SourceNoise}}$$

Be sure to use the same units on all quantities. Note that, because in this procedure the noise due to the source and load are obtained by hand inspection, the procedure can be used to compute noise figure even if ports are not used to drive the circuit. Hand calculations are not recommended because they are tedious and prone to error.

In versions of the Analog Circuit Design Environment prior to 4.4.5, it was possible to perform noise figure computations in ways that incorrectly or incompletely specified the sources/input or load/output. Since, at that time, there was no way for the simulation environment to track down which component was contributing, for example, the *LoadNoise*, this part of the noise factor might have been neglected.

The rule of thumb is: *if you have not specified your sources and loads to the simulator, the simulator does not know what the loads and sources are*, so the simulator may not be able to compute noise figure properly. Generally, the input to the circuit must be a *port*, and *not* some other electrically equivalent combination of components. The output may be a resistor probe, or it may be a port. In up-to-date versions of software, 4.4.2 and later, using ports (or possibly a resistor as the output probe) will insure accurate computations. Inspect the netlist if you are unsure about the identity of the components presented to the simulator.

Finally, if you ever have any doubt about the propriety of the procedure you are using, use the results browser to access the noise figure directly from the simulation data. This data is correct in all software versions.

### **Why is the gain reported by Pnoise twice the gain that I expect?**

- First, unlike some circuit simulators, Pnoise and PSP compute the gain of the circuit from its input to its output, instead of from an external source to the output. See [“Definitions of Gain”](#) on page 1102 to see if the gain you expect is actually the gain that you want.
- Second, Pnoise makes certain assumptions about input matching in order to calculate this gain. See [“Gain Calculations in Pnoise”](#) on page 1106 for further explanations of the way Pnoise calculates gain.
- In general, you should not use the gain calculated by Pnoise unless you know your circuit is input-matched. Use PSP instead.
- Does Spectre RF compute single-sideband (SSB) or double-sideband (DSB) noise figure?

The Spectre RF Pnoise analysis computes SSB noise figure, (except in certain special cases). PSP can compute both SSB and DSB noise figure. See [“Noise Figure”](#) on page 1089.

### **Why does the axis for noise or noise figure have negative frequencies?**

The axis labeling in Spectre RF is fairly independent of the actual computation. In all cases, Spectre RF computes single-sided noise (that is, there is no need to add noise from *negative* frequencies to get the total noise power or noise figure).

Recall that PSP analysis performs many computations at once—periodic S-parameters, noise correlation matrices, noise figure, etc. In 4.4.5, for historical reasons, all these quantities are stored as a sweep with a single axis. An axis appropriate for visualizing  $s_{11}$  may not be good to display NF.

To obtain strictly positive axis labels for noise quantities in PSP, choose a positive integer for the output harmonic, use *sweep<sub>type</sub>=relative*, and set *freqaxis=out* on the PSP analysis options form.

### **How does match affect the noise and gain calculations in Spectre RF?**

If the input impedance is not matched to the port impedance, the gain and input referred noise, or IRN, reported by Pnoise analysis is probably not the gain you want. *out*, *F*, *NF*, and *in* (input-referred noise) are still correct.

PSP analysis computes the correct *G*, *in* (input-referred noise), *F* and *NF* regardless of match.

## **Does Spectre RF include noise parameter data from S-parameter files?**

Spectre RF supports writing and reading S-parameter files, but as of the 4.4.5 release it does not write noise parameter data into files, nor can noise parameter data be read from file and included in a simulation.

## **Known Problems and Limitations**

### **Dubious AC-Noise Analysis Features**

When computing noise figure in circuits with a DC bias point in releases prior to 4.4.5, the noise figure available from the direct plot form was obtained from knowledge of the total output noise computed with a Noise analysis, the source impedance as entered on a form, and circuit gain data obtained in an auxiliary AC analysis. Simulators other than Spectre RF still require this procedure.

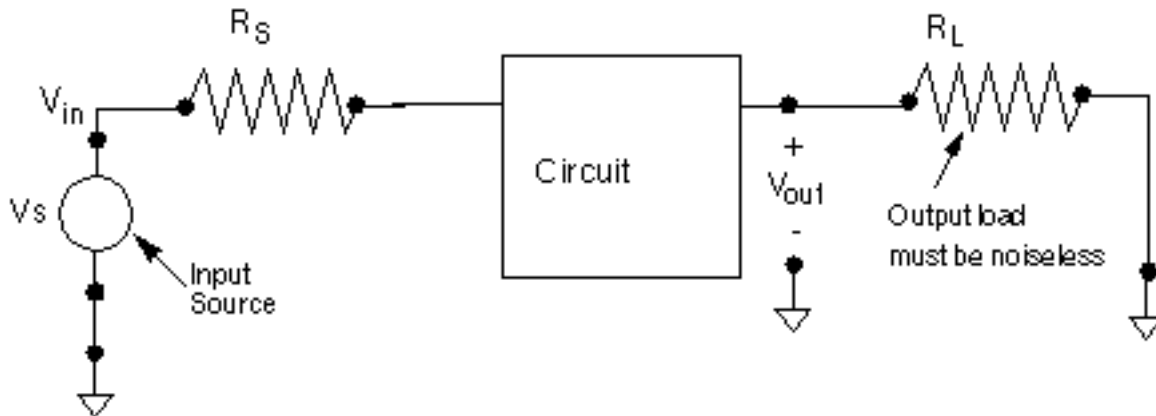
In 4.4.5, if you use ports as the output load and input source, the noise figure is obtained directly from the Spectre RF simulator data. An artifact of the previous use model is that the noise figure direct plot still requires an AC analysis to run, but this analysis has no effect on the computed noise figure if ports are used as sources and loads.

If ports are not used as the input source and load (for example, using a *vsin* component as the input and a pair of nodes as the output) it is still possible to compute noise figure in the Analog Circuit Design Environment, but the procedure is not recommended as it can be tricky to obtain the desired result (see [“Performing Noise Figure Computations”](#) on page 1089 and [“The noise figure computed by <Procedure X> seems inaccurate, inconsistent, or just plain wrong.”](#) on page 1109).

To obtain reasonable noise figure calculations using this (unsupported) procedure, you must configure the circuit as shown in [Figure L-16](#) on page 1113. Be sure to select the proper nodes as input and output (note that a *port* component cannot be used in this topology because the *Vin* node cannot be accessed). The contribution of the output load to the total noise must be eliminated by making the node noiseless, for example by setting `Generate noise?` in a resistor component to `no`.



Figure L-16 Alternative (Undesirable) Noise Figure Computation Setup



Note that you cannot use this topology to compute noise figure in Pnoise, SP or PSP analyses.

### Gain in Pnoise and PSP Analyses Inconsistent

The *gain* computed by the Pnoise analysis is really a number that relates equivalent input and output noise. PSP analysis computes a somewhat different quantity that can be interpreted as a circuit gain. If the input impedance of the circuit is matched, the PSP and Pnoise gains match. See the ["Gain Calculations"](#) on page 1102 for details on computing circuit gains.

### Harmonics and Sidebands in PSP, PAC, PXF, and Pnoise Analyses

Frequent users of Spectre RF may already be familiar with the concepts of *harmonics* and *sidebands*.

Typically, the term *harmonic* is used to refer to a signal at a multiple of the carrier or local oscillator frequency.

If an additional input is applied, such as for RF data, additional responses appear as *sidebands*.

In small-signal analysis, only the first-order sidebands are computed.

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

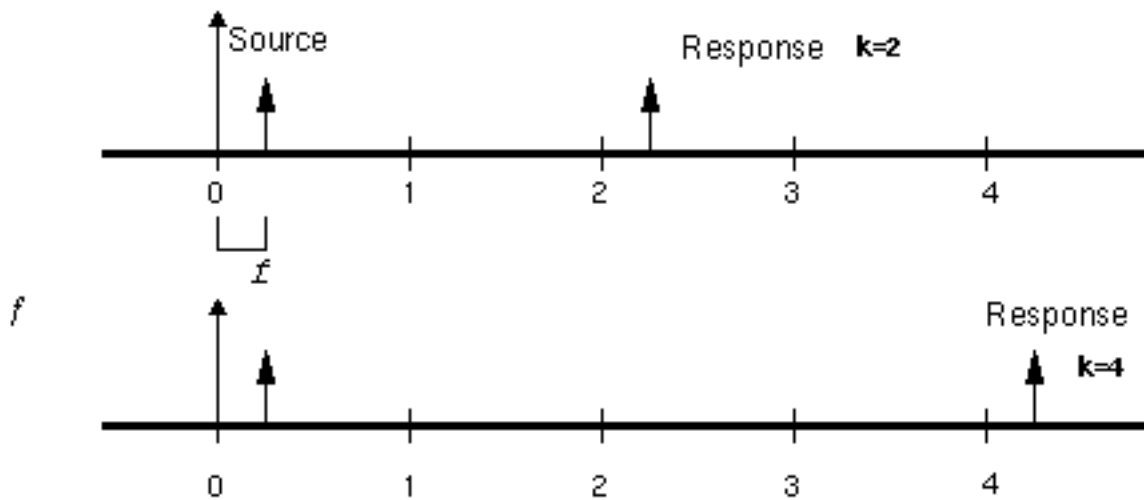
### Using PSP and Pnoise Analyses

---

In the context of small-signal analysis, a frequency is specified by an index  $k$  and a frequency offset  $f$ . These numbers specify a frequency  $kf_0 + f$ . In an absolute indexing scheme, such as used in PSP analysis, if the frequency  $f$  is restricted to lie in an interval of size  $f_0$ , then for any given frequency, the numbers  $k$  and  $f$  are unique.

For example, consider an upconversion mixer, as shown in Figure L-17. Two possible computations are shown. In both cases, the source is at  $k=0$ , baseband, and relative offset  $f$ . In the top half of the figure, the response is at  $k=+2$ , frequency  $f$ , and in the bottom half,  $k=4$ , offset  $f$ .

**Figure L-17 Absolute Indexing Schemes in Spectre RF Small-Signal Analyses**



In contrast, PAC, PXF, and Pnoise analyses use a relative indexing scheme. In a relative indexing scheme, a single frequency may be referred to in different ways depending on context.

**Figure L-18 Relative Indexing in Spectre RF**

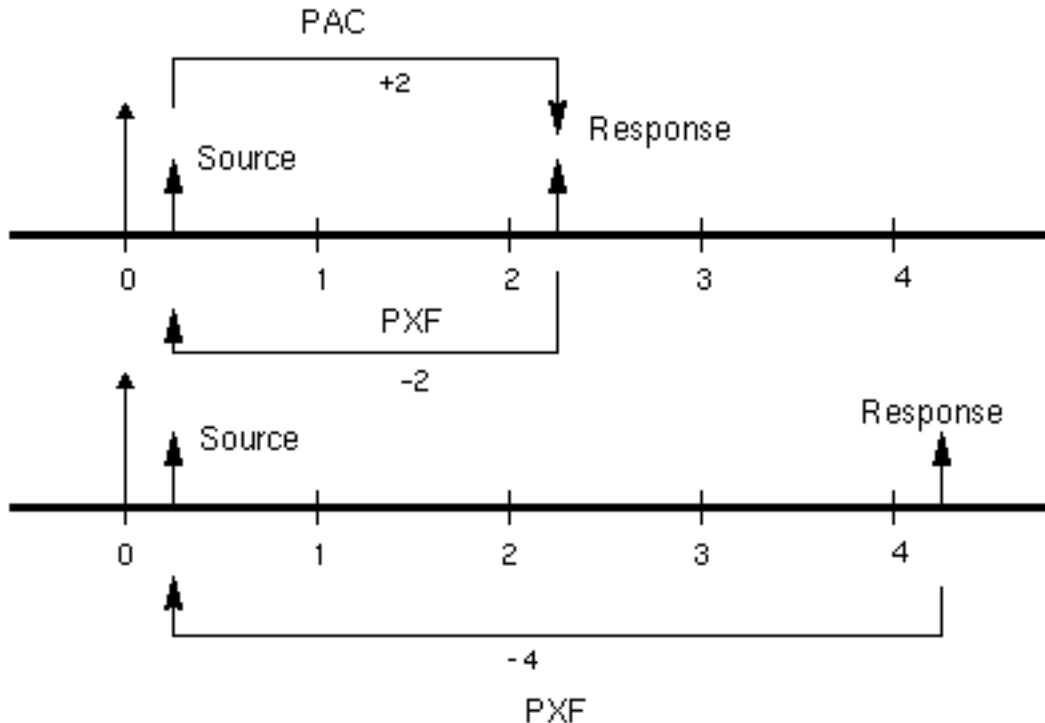


Figure L-18 on page 1115 shows how relative indexing may change depending on context. In the top half of the figure, a source may be specified in PAC analysis using either *sweeptype=absolute* and frequency  $-f_0+f$ , or *sweeptype=relative*, *relharmnum=-1*, and frequency  $f$ . The response is shown at frequency  $f_0+f$ , which is  $k=2$ , because  $f_0+f = -f_0+f + 2f_0$ .

If instead a PXF analysis was used to analyze the circuit, then the location of the *response* is specified with either *sweeptype=absolute* and frequency  $f_0+f$ , or *sweeptype=relative* and *relharmnum=+1*. The source is at  $k=-2$ . Now consider the bottom half of the figure. The response is shown at the frequency  $3f_0+f$ . If PXF analysis is used to analyze the circuit, the source lies at  $k=-4$ .

Note that if all the frequencies involved in the above examples were shifted upwards by  $f_0$ , then the indexing in PSP analysis would change by  $+1$ , but the indexing of the transfer functions in PXF and PAC analyses would remain unchanged, even though different frequencies, and thus physical different transfer functions, would be computed.

The names used for the responses and transfer functions in the Spectre RF environment are not entirely consistent in release 4.4.5 and earlier. The Spectre RF interface asks for *sidebands* in order to specify which of the transfer functions associated with the relative

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

### Using PSP and Pnoise Analyses

---

index  $k$  are wanted. This is reasonable because the small-signal responses appear as sidebands to the harmonics of the fundamental frequency of the PSS steady-state solution. However, the data is output and displayed with the label *harmonic*. Thus the index naming is not consistent. However, since the transfer function associated with the relative index  $k$  results from the small-signal being convolved with the  $k^{\text{th}}$  harmonic of the steady-state solution, the *harmonic* label is not entirely inappropriate either.

# Index

## Symbols

./inilmg file, in LMG [204](#)

## Numerics

1st Order Harmonic [128](#)  
3rd Order Harmonic [128](#)

## A

Accuracy Defaults specification [33](#)  
accuracy parameters  
    allglobal [108](#)  
    alllocal [108](#)  
    lteratio [106](#), [108](#)  
    maxacfreq [108](#)  
    maxperiods [108](#)  
    pointlocal [109](#)  
    relref [108](#)  
    sigglobal [109](#)  
    steadyratio [109](#)  
accuracy, of LMG models [284](#)  
Add to Outputs specification, on Results  
    forms [128](#)  
Additional Time for Stabilization, entry on  
    Choosing Analyses form [34](#)  
allglobal parameter [108](#)  
alllocal parameter [108](#)  
Analysis type specification, on Results  
    forms [129](#)  
annotate parameter [110](#)  
annotation parameters  
    annotate [110](#)  
    stats [110](#)  
Auto Calculate Button [35](#)

## B

before Simulation value of Start MATLAB  
    field [680](#)

## C

Cadence libraries, setting up [179](#), [181](#)  
Calculate Parameters function button, in  
    LMG [201](#), [242](#), [297](#)  
Center - Span specification, Frequency  
    Sweep Range (Hz) fields [39](#), [160](#)  
Choosing Analyses form [30](#)  
    Accuracy Defaults specification [33](#)  
    Analysis specification [34](#)  
    Auto Calculate Button [35](#)  
    Do Noise [37](#)  
    Enabled button [37](#)  
    Frequency (Hz) field [39](#)  
    Frequency Sweep Range (Hz) fields [38](#)  
        Center - Span specification [39](#), [160](#)  
        Single - Point specification [39](#)  
        Start - Stop specification [38](#), [159](#)  
    Input Source specification [48](#), [49](#), [50](#),  
        [52](#), [53](#)  
    Input Voltage Source specification [49](#),  
        [52](#)  
    Options specification [62](#)  
    Oscillator button [63](#)  
    Output  
        harmonics selection [67](#)  
        Number of harmonics [67](#)  
    Output harmonics selection  
        Array of harmonics [70](#), [82](#)  
        Select from range [68](#)  
    Output specification for Pnoise and  
        PXF [39](#), [64](#)  
    Output specification for Pnoise and  
        QPnoise [65](#)  
    Output specification for PXF [64](#)  
    Save Initial Transient Results  
        specification [73](#)  
    Select Ports [73](#)  
    Sidebands specification [78](#), [82](#)  
        Maximum clock order [83](#)  
        Maximum sideband [37](#), [79](#)  
        Select from range [80](#), [83](#)  
    Small Signal Periodic Analysis list  
        box [92](#)  
    Sweep Range (Hz) specification  
        Center - Span [95](#)

- Start - Stop [95](#)
- Sweep specification [92](#)
  - variable [92](#)
- Sweep Type specification [95](#)
  - Add Specific Points [41](#), [98](#)
  - Logarithmic [41](#), [97](#), [161](#)
- Choosing Analyses form fields [31](#)
- Clear/Add button, Fundamental Tones list box [45](#)
- cmin parameter [110](#)
- compression parameter [113](#)
- Conductivity type-in field
  - in LMG [296](#)
- Conductivity type-in field, in LMG [200](#)
- Conductor Distances type-in field, in LMG [295](#)
- Conductor Height (h) type-in field, in LMG [295](#)
- Conductor Height type-in field, in LMG [200](#)
- Conductor Length type-in field, in LMG [200](#), [222](#), [296](#)
- Conductor Thickness type-in field, in LMG [200](#), [222](#), [295](#)
- Conductor Width type-in field, in LMG [200](#), [221](#), [294](#)
- convergence parameters
  - cmin [110](#)
  - gear\_order [110](#)
  - oscsolver [110](#)
    - ira [110](#)
    - std [110](#)
    - turbo [111](#)
  - readns [111](#)
  - solver [111](#)
    - std [111](#)
    - turbo [111](#)
  - tolerance [111](#)
- cosimulation
  - generating netlist for lower-level block [677](#)
  - inputs, selecting [678](#)
  - options, enabling [679](#)
  - outputs, selecting [678](#)
  - running [685](#)
  - setting up and running [672](#)
  - simulation response timeout, setting [675](#)
  - socket mode for, setting [675](#)
  - socket port [679](#)
  - socket port for, setting [675](#)
  - software required for [672](#)

- Spectre command used in [675](#)
  - start method for [677](#), [679](#)
  - starting ADE manually and MATLAB automatically [681](#)
  - starting MATLAB and Spectre RF separately [680](#), [686](#)
  - starting MATLAB manually and Spectre RF automatically [682](#), [687](#)
  - starting Spectre RF manually and MATLAB automatically [686](#)
  - stop time [681](#)
    - with MATLAB, introduction to [672](#)
- coupler block, connecting to system-level Simulink schematic [673](#)
- Create Macromodel function button, in LMG [297](#)

## D

- data entry
  - Fundamental Tones list box
    - buttons [42](#)
    - fields [42](#)
  - data entry section on form in LMG [292](#)
  - dB10, as Results form modifier [139](#)
  - dB20, as Results form modifier [139](#)
  - dBm, as Results form modifier [139](#)
  - default values, in LMG [194](#)
  - Delete button [45](#)
  - Dielectric Constant type-in field
    - in LMG [200](#), [221](#)
  - Dielectric Thickness (d) type-in field, in LMG [294](#)
  - Dielectric Thickness type-in field, in LMG [200](#), [221](#)
  - display section on form, in LMG [292](#)
  - Do Noise, Choosing Analyses form [37](#)

## E

- Edit Object Properties form
  - use external model file button [210](#)
- Enabled button [37](#)
- errpreset parameter [33](#)
- euler parameter [112](#), [113](#)
- examples
  - modeling transmission lines using the LMG form (use model one)
    - contents of example model [195](#),

[218](#)  
 modeling transmission lines using the  
     Transmission Line Modeler form  
     (use model one)  
     creating the model [195](#)  
 modeling transmission lines using the  
     Transmission Line Modeler form  
     (use model one)  
     use of lumped models within Analog  
         Artist [195](#), [205](#), [213](#), [247](#),  
         [273](#)  
 modeling transmission lines without  
     using the Transmission Line  
     Modeler form (use model  
     two) [213](#)  
 Expr field, Fundamental Tones list box [42](#)  
 Extrapolation Point specification, on Results  
     forms [130](#)

## F

File menu  
     in LMG [288](#), [289](#)  
 FIR filters  
     Editing with Modelwriter [782](#)  
 first-order harmonic specification, on Results  
     forms [130](#)  
 Fmax type-in field, in LMG [200](#), [222](#), [296](#)  
 Format — Port/Signal Displays — Signal  
     Dimensions [676](#)  
 forms  
     Choosing Analyses form fields [31](#)  
     Options [104](#)  
     Results [128](#)  
 frame size, specifying [674](#)  
 Freq. Multiplier field [132](#)  
 Freq. Unit menu choices, in LMG [291](#)  
 freqaxis parameter  
     absin [113](#)  
     in [113](#)  
     out [113](#)  
 Frequency (Beat) specification,  
     Fundamental Tones [35](#)  
 Frequency (Hz) field [39](#)  
 frequency multiplier [132](#)  
 Frequency Sweep Range (Hz) fields [38](#)  
 function  
     buttons, in LMG [287](#)  
     specification, on Results forms [132](#)  
 function buttons

    in LMG [296](#)  
 Fundamental Tones [42](#)  
     Frequency (Beat) specification [35](#)  
     list box [42](#), [45](#)  
         Clear/Add button [45](#)  
         data entry buttons [42](#)  
         data entry fields [42](#)  
         Delete button [45](#)  
         Expr field [42](#)  
         Harms field [42](#)  
         Name field [42](#)  
         Signal field [42](#)  
         Srcld field [42](#)  
         Update from Schematic button [45](#)  
         Value field [42](#)  
     Period (Beat) specification [35](#)  
 Fundamental Tones list box [45](#)

## G

gear\_order parameter [110](#), [112](#), [113](#)

## H

Harms field, Fundamental Tones list  
     box [42](#)  
 harms parameter [108](#)

## I

ic parameter [111](#)  
 Imaginary, as Results form modifier [139](#)  
 initial condition parameters  
     ic [111](#)  
     readic [111](#)  
     skipdc [111](#)  
     no [111](#)  
     yes [111](#)  
 Input Current Source specification  
     Input Current Source specification [49](#),  
     [53](#)  
 input pins, number of in SpectreRF Engine  
     block [674](#)  
 Input Port Source specification [50](#), [53](#)  
 Input Port Source specification, Input Port  
     Source specification [49](#), [53](#)  
 Input Source specification [48](#), [52](#)  
 integration method parameters

method [112](#)  
 euler [112, 113](#)  
 gear2 [112, 113](#)  
 gear2only [112, 113](#)  
 trap [112, 113](#)  
 traponly [112, 113](#)  
 ira parameter value [110](#)

### L

Length Unit menu choices, in LMG [291](#)  
 LMG [191](#)  
 ./inilmg file [204](#)  
 accuracy of [284](#)  
 data entry section [292](#)  
     Conductivity [200, 296](#)  
     Conductivity type-in field, in LMG [222](#)  
     Conductor Distances [295](#)  
     Conductor Height [200, 295](#)  
     Conductor Length [200, 222, 296](#)  
     Conductor Thickness [200, 222, 295](#)  
     Conductor Width [200, 221, 294](#)  
     Dielectric Constant [200, 221](#)  
     Dielectric Constant (er) [293](#)  
     Dielectric Thickness [200, 221](#)  
     Dielectric Thickness (d) [294](#)  
     Fmax [200, 222, 296](#)  
     Number of Conductors [200, 221, 293](#)  
 default values [194](#)  
 Dielectric Constant (er) type-in field, in LMG [293](#)  
 display section on form [292](#)  
 File menu  
     Quit [288, 290](#)  
     Save Current Settings [289](#)  
     Subcircuit Name [288, 289](#)  
 function buttons [287, 296](#)  
     Calculate Parameters [201, 242, 297](#)  
     Create Macromodel [297](#)  
 Macromodel Creation pop-up window [203, 243](#)  
 Options menu  
     Freq. Unit [291](#)  
     Length Unit [291](#)  
     Lossless Single Line menu choice [291](#)  
     Lossless Single Line menu choice, in

    LMG [291](#)  
     Lossy, Narrow Band, Single Line menu choice [290](#)  
     Lossy, Narrow Band, Single Line menu choice, in LMG [290](#)  
     Lossy, Wide Band, Single Line menu choice [290](#)  
     Lossy, Wide Band, Single Line menu choice, in LMG [290](#)  
     Model Type [290](#)  
     Output file Control [292](#)  
     Subckt Format [291](#)  
     Save Current Settings form [204](#)  
     Subcircuit Name form [202, 242](#)  
     Transmission Line Modeler form [287](#)  
 iteratio parameter [106, 108](#)

### M

Macromodel Creation pop-up window [203, 243](#)  
 Magnitude, as Results form modifier [139](#)  
 Math Operations library [676](#)  
 MATLAB  
     design name [680, 681](#)  
     start command [680](#)  
     startup directory [680](#)  
 maxacfreq parameter [108](#)  
 maxiters parameter [113](#)  
 maxperiods parameter [108](#)  
 maxstep parameter [108, 117](#)  
 method parameter [112](#)  
     euler [112, 113](#)  
     gear2 parameter value [112, 113](#)  
     gear2only parameter value [112, 113](#)  
     trap [112, 113](#)  
     traponly [112, 113](#)  
 Microstrip menu choice, in LMG [293](#)  
 Model Type menu choice, in LMG [290](#)  
 modeling transmission lines  
     using LMG outside Analog Artist (use model one)  
         described [193](#)  
     modeling transmission lines from inside Analog Artist (use model two)  
         described [194](#)  
         example [213](#)  
 Modifier specification, on Results forms [139](#)  
     dB10 [139](#)



dB20 [139](#)  
 dBm [139](#)  
 Imaginary [139](#)  
 Magnitude [139](#)  
 Phase [139](#)  
 Real [139](#)

## N

Name field, Fundamental Tones list box [42](#)  
 netlist  
   generating for cosimulation [677](#)  
   generating for lower-level block in  
     cosimulation [677](#)  
   preparing for cosimulation in ADE [677](#)  
   preparing for cosimulation without using  
     a GUI [683](#)  
 newlink Variable [92](#)  
 Newton parameters  
   maxiters [113](#)  
   restart [113](#)  
 no command value, of Start MATLAB  
   field [680](#)  
 nodes\_and\_terminals, stimuli parameter  
   value [115](#)  
 noise temperature [738](#), [1009](#)  
 noisetemp parameter [738](#), [1009](#)  
 now command value  
   in MATLAB cosimulation [679](#)  
   of Start MATLAB field [679](#)  
   using to test whether ADE starts  
     MATLAB [679](#)  
 Number of Conductors type-in field, in  
   LMG [200](#), [221](#), [293](#)

annotation parameters  
   annotate [110](#)  
   stats [110](#)  
 convergence parameters  
   cmin [110](#)  
   gear\_order [110](#)  
   oscsolver [110](#)  
   readns [111](#)  
   solver [111](#)  
   tolerance [111](#)  
 initial condition parameters  
   ic [111](#)  
   readic [111](#)  
   skipdc [111](#)  
 integration method parameters  
   method [112](#)  
 Newton parameters  
   maxiters [113](#)  
   restart [113](#)  
 output parameters  
   compression [113](#)  
   freqaxis [113](#)  
   oppoint [114](#)  
   skipcount [115](#)  
   skipstart [115](#)  
   skipstop [115](#)  
   stimuli [115](#)  
   strobedelay [115](#)  
   strobeperiod [115](#)  
 simulation interval parameters  
   tstart [116](#)  
 state file parameters  
   swapfile [117](#)  
   write [117](#)  
 time step parameters  
   maxstep [117](#)  
   step [118](#)

Options button [62](#)  
 Options form [104](#), [106](#)  
   Accuracy Parameters [106](#)  
   ANNOTATION PARAMETERS [110](#)  
   CONVERGENCE PARAMETERS [110](#)  
   INITIAL CONDITION  
     PARAMETERS [111](#)  
   INTEGRATION METHOD  
     PARAMETERS [112](#)  
   NEWTON PARAMETERS [113](#)  
   OUTPUT PARAMETERS [113](#)  
   SIMULATION BANDWIDTH  
     PARAMETERS [116](#)  
   SIMULATION INTERVAL

oppoint parameter [114](#)  
 options  
   accuracy parameters  
     allglobal [108](#)  
     alllocal [108](#)  
     lteratio [106](#), [108](#)  
     maxacfreq [108](#)  
     maxperiods [108](#)  
     pointlocal [109](#)  
     relref [108](#)  
     sigglobal [109](#)  
     steadyratio [109](#)

- PARAMETERS [116](#)
- STATE FILE PARAMETERS [116](#)
- TIME STEP PARAMETERS [117](#)
- Options menu
  - in LMG [290](#)
- Oscillator button [63](#)
- oscillator noise analysis [689](#)
- oscsolver parameter [110](#)
- Output file Control menu choices, in LMG [292](#)
- Output harmonics selection [67](#)
  - Array of harmonics [70, 82](#)
  - Number of harmonics [67](#)
  - Select from range [68](#)
- output parameters
  - compression [113](#)
  - freqaxis [113](#)
  - oppoint [114](#)
  - skipcount [115](#)
  - skipstart [115](#)
  - skipstop [115](#)
  - stimuli [115](#)
    - nodes\_and\_terminals [115](#)
    - sources [115](#)
  - strobedelay [115](#)
  - strobeperiod [115](#)
- output pins, number of in SpectreRF Engine block [674](#)
- Output specification for Pnoise and PXF voltage [64](#)
- Output specification for Pnoise and QPnoise [65](#)
- Output specification for PXF [64](#)
- overview, Spectre RF analyses [23](#)

## P

- PAC analysis
  - freqaxis parameter [113](#)
- parameters
  - maxstep [108](#)
- Period (Beat) specification, Fundamental Tones [35](#)
- phase noise, discussion of [689](#)
  - and SpectreRF simulation [703](#)
  - frequently asked questions [711](#)
  - further reading [716](#)
  - models [693](#)
  - troubleshooting [706](#)
- Phase, as Results form modifier [139](#)

- pins, input, in SpectreRF Engine block [674](#)
- pins, output, in SpectreRF Engine block [674](#)
- plot mode specification, on Results forms [144](#)
- Pnoise analysis [48, 52](#)
  - Input Source specification [48, 52](#)
  - Input Voltage Source specification [49, 52](#)
  - Output specification [65](#)
- pointlocal parameter [109](#)
- psin, using in SpectreRF simulation [725](#)
- PSP analysis
  - freqaxis parameter [114](#)
- PSS analysis
  - spectral plots [122](#)
  - time waveform plots [125](#)
- PXF analysis
  - freqaxis parameter [114](#)
  - Output specification [64](#)

## Q

- QPnoise analysis
  - Output specification [65](#)
- QPSS
  - harmonics [46](#)
- Quit menu choice, in LMG [288, 290](#)

## R

- readic parameter [111](#)
- readns parameter [111](#)
- Real, as Results form modifier [139](#)
- Reference Harmonic specification, on Results forms [135](#)
- relref parameter [108](#)
- restart parameter [113](#)
- Results forms [118, 128](#)
  - Add to Outputs specification [128](#)
  - Analysis type specification [129](#)
  - Extrapolation Point specification [130](#)
  - first-order harmonic specification [130](#)
  - function specification [132](#)
  - Modifier specification [139](#)
    - dB10 [139](#)
    - dB20 [139](#)
    - dBm [139](#)
  - Imaginary [139](#)

## Virtuoso Spectre Circuit Simulator RF Analysis User Guide

---

- Magnitude [139](#)
  - Phase [139](#)
  - Real [139](#)
  - plot mode specification [144](#)
  - Reference Harmonic specification [135](#)
  - Signal -value specification [149](#)
  - Sweep specification [149](#)
  - RF Library
    - Bottom-Up Elements [787](#)
    - Measurement
      - CDMA [769](#)
      - DQPSK [776](#)
      - Editing FIR filters with Modelwriter [782](#)
      - eye-diagram generator [779](#)
      - GSM [773](#)
    - Original\_RFAHDL\_lib
      - balun [746](#)
      - balun\_com [748](#)
      - filter [749](#)
      - low noise amplifier [752](#)
      - mixer [755](#)
      - phase shifter [766](#)
      - power amplifier [759](#)
      - quadrature signal generator [765](#)
    - Testbenches Elements [786](#)
    - Uncategorized Elements [787](#)
  - running PSS effectively [719](#)
    - convergence aids [719](#)
      - for oscillators [721](#)
    - running PSS hierarchically [722](#)
- ## S
- sample time, setting for block used in cosimulation [675](#)
  - Save Current Settings
    - form, in LMG [204](#)
    - menu choice, in LMG [289](#)
  - Save Initial Transient Results
    - specification [73](#)
  - Select Design Variable form [93](#)
  - Select Ports, Choosing Analyses form [73](#)
  - setting up the software [177](#)
    - setting up the Cadence libraries [179](#)
    - using the UNIX shell window [181](#)
  - Setup — Matlab/Simulink — Log file [680](#)
  - Setup — Matlab/Simulink — Setting [678](#)
  - Sidebands specification, Choosing Analyses form [78, 82](#)
  - Maximum clock order [83](#)
  - Maximum sideband [37, 79](#)
  - Select from range [80, 83](#)
  - sigglobal parameter [109](#)
  - Signal <variable> value specification, on Results form [149](#)
  - Signal field, Fundamental Tones list box [42](#)
  - Simulation — Configuration
    - Parameters [676](#)
  - Simulation — Start [686, 687](#)
  - simulation interval parameters
    - tstart [116](#)
  - simulation response timeout for cosimulation [675](#)
  - Single - Point specification, Frequency Sweep Range (Hz) fields [39](#)
  - skipcount parameter [115](#)
  - skipdc parameter [111](#)
    - no [111](#)
    - yes [111](#)
  - skipstart parameter [115](#)
  - skipstop parameter [115](#)
  - Small Signal Periodic Analysis list box, Choosing Analyses form [92](#)
  - socket mode, for cosimulation [675](#)
  - socket port, for cosimulation [675](#)
  - solver
    - ira [110](#)
    - std [110, 111](#)
    - turbo [111](#)
  - solver parameter [111](#)
  - sources, stimuli parameter value [115](#)
  - S-parameter file format translator [1063](#)
  - S-parameter simulation data, plotting [897](#)
    - equations for S-parameter waveform calculator [897](#)
  - Spectre command, for cosimulation [675](#)
  - Spectre RF
    - analyses, overview [23](#)
  - SpectreRF Engine
    - block, number of input pins in [674](#)
    - block, number of output pins in [674](#)
    - showing port labels for [675](#)
  - Srcld field, Fundamental Tones list box [42](#)
  - Start - Stop specification, Frequency Sweep Range (Hz) [38, 159](#)
  - Start MATLAB field
    - before Simulation value [680](#)
    - no value [680](#)
    - now value [679](#)
    - values for [679](#)

- state file parameters
  - swapfile [117](#)
  - write [117](#)
- stats parameter [110](#)
- std parameter value [110](#), [111](#)
- steadyratio parameter [109](#)
- step parameter [118](#)
- stimuli parameter [115](#)
  - nodes\_and\_terminals [115](#)
  - sources [115](#)
- stop time, for cosimulation [681](#)
- Stripline menu choice, in LMG [293](#)
- strobedelay parameter [115](#)
- strobeperiod parameter [115](#)
- Subcircuit Name
  - form, in LMG [202](#), [242](#)
  - menu choice, in LMG [288](#), [289](#)
- Subckt Format menu choice, in LMG [291](#)
- swapfile parameter [117](#)
- Sweep Range (Hz) specification
  - Center - Span [95](#)
  - Start - Stop [95](#)
- Sweep specification
  - Choosing Analyses form [92](#)
  - on Results forms [149](#)
  - variable [92](#)
- Sweep Type [95](#)
  - Add Specific Points [41](#), [98](#)
  - Logarithmic [41](#), [97](#), [161](#)

## T

- time step parameters
  - maxstep [117](#)
  - step [118](#)
- tolerance parameter [111](#)
- Tools — Analog Environment [678](#)
- Transmission Line Model Generator, See LMG
- Transmission Line Modeler form
  - data entry section [292](#)
  - display section [292](#)
- Transmission Line Modeler form, in LMG [287](#)
- transmission lines
  - accuracy of LMG models [284](#)
  - modeling transmission lines from inside Analog Artist (use model two)
    - described [194](#)
    - example [213](#)

- using LMG outside Analog Artist (use model one)
  - described [193](#)
- trap parameter value [112](#), [113](#)
- traonly parameter value [112](#), [113](#)
- tstart parameter [116](#)
- turbo parameter value [111](#)

## U

- Update from Schematic button [45](#)
- use external model file button [210](#)
- using in SpectreRF simulation
  - port parameter types [986](#)
  - psin parameter types [727](#)
- Using psin in SpectreRF simulations [725](#)

## V

- Values field, Fundamental Tones list box [42](#)
- View — Simulink library [676](#)
- Voltage Source specification, Pnoise analysis [49](#), [52](#)

## W

- write parameter [117](#)