

# **SIMetrix / SIMPLUS**

## **USER'S MANUAL**

**VERSION 8.5**

JUNE 2021

# **SIMETRIX/SIMPLIS USER'S MANUAL**

**COPYRIGHT © SIMETRIX TECHNOLOGIES LTD. 1992-2021**

**COPYRIGHT © SIMPLIS TECHNOLOGIES INC. 1992-2021**

**Trademarks:**

PSpice is a trademark of Cadence Design Systems Inc.

Hspice is a trademark of Synopsis Inc.

SIMetrix Technologies Ltd.,  
78 Chapel Street,  
Thatcham,  
Berkshire  
RG18 4QN  
United Kingdom

SIMPLIS Technologies, Inc.  
P.O. Box 40084  
Portland, OR 97240-0084  
USA

Tel: +44 1635 866395

Fax: +44 1635 868322

Email: [support@simetrix.co.uk](mailto:support@simetrix.co.uk)

Web: <http://www.simetrix.co.uk>

Tel: +1 503 766 3928

Fax: +1 503 296 5674

Email: [info@simplistechnologies.com](mailto:info@simplistechnologies.com)

Web: <http://simplistechnologies.com>



# Contents

<b>1</b>	<b>Welcome</b>	<b>1</b>
<b>2</b>	<b>Introduction</b>	<b>2</b>
2.1	About SIMetrix/SIMPLIS Documentation . . . . .	2
2.1.1	HTML Documentation . . . . .	2
2.1.2	PDF Documentation . . . . .	2
2.2	What is Simetrix? . . . . .	2
2.3	What is SIMPLIS? . . . . .	3
2.4	Why Simulate? . . . . .	4
2.5	System Requirements . . . . .	4
2.5.1	Operating System . . . . .	4
2.5.2	Hardware . . . . .	5
2.5.3	Recommended System . . . . .	5
2.5.4	Multi-core Processors . . . . .	5
2.6	Acknowledgements . . . . .	5
<b>3</b>	<b>Quick Start</b>	<b>7</b>
3.1	Introduction . . . . .	7
3.1.1	Examples and Tutorials - Where are They? . . . . .	7
3.1.2	Simulation for the Novice . . . . .	7
3.2	Tutorial 1 - A Simple Ready to Run Circuit . . . . .	8
3.3	Tutorial 2 - A Simple SMPS Circuit . . . . .	18
3.4	Tutorial 3 - Installing Third Party Models . . . . .	26
3.4.1	Notes . . . . .	29
<b>4</b>	<b>Getting Started</b>	<b>31</b>
4.1	Simulation Modes - SIMetrix or SIMPLIS . . . . .	31
4.2	Using the Schematic Editor . . . . .	32
4.2.1	Creating a Schematic . . . . .	32
4.3	Circuit Rules . . . . .	34
4.4	Analysis Modes . . . . .	34
4.4.1	Overview . . . . .	34
4.4.2	Setting up a SIMetrix Simulaton . . . . .	34
4.4.3	Setting up a SIMPLIS Simulation . . . . .	39
4.4.4	Manual Entry of Simulator Commands . . . . .	42
4.5	Running the Simulator . . . . .	43
4.5.1	SIMetrix . . . . .	43
4.5.2	SIMPLIS . . . . .	43
4.6	Plotting Simulation Results . . . . .	43
4.6.1	Overview . . . . .	43
4.6.2	Fixed Probes . . . . .	44
4.6.3	Random Probes . . . . .	45
<b>5</b>	<b>SIMetrix GUI Environment</b>	<b>47</b>

5.1	Overview	47
5.2	Interaction	48
5.3	Directory Viewer	51
5.4	Interface Styles	52
<b>6</b>	<b>Schematic Editor</b>	<b>54</b>
6.1	Fundamentals	54
6.1.1	Schematic Windows and Sheets	54
6.1.2	Schematic Editor Window	55
6.1.3	Editing Operations	55
6.1.4	Wiring	60
6.1.5	Bus Connections	62
6.1.6	Copying to the Clipboard	63
6.1.7	Annotating a Schematic	63
6.1.8	Assigning Part References	63
6.1.9	Checking the Schematic	63
6.1.10	Schematic Preferences	64
6.1.11	Adding and Removing Worksheets	64
6.1.12	Finding and Specifying Net Names	64
6.2	Hierarchical Schematic Entry	65
6.2.1	Top-Down Method	65
6.2.2	Bottom-up method	66
6.2.3	Navigating Hierarchical designs	66
6.2.4	Placing - Full vs Relative Path	66
6.2.5	Connecting Buses in a Hierarchy	67
6.2.6	Global Nets	68
6.2.7	Global Pins	68
6.2.8	Passing Parameters Through a Hierarchy	70
6.2.9	Missing Hierarchical Blocks	71
6.2.10	Highlighting	71
6.2.11	Copying a Hierarchy	71
6.3	Schematic Annotations	72
6.3.1	Line Based Annotations	72
6.3.2	Shape Based Annotations	73
6.3.3	Image Annotations	76
6.3.4	Text Annotations	77
6.3.5	Title Block	77
6.4	Schematic Styles	78
6.4.1	Style Editor	78
6.4.2	Simplified Style Editing and Creation	81
6.4.3	Advanced Style Settings	82
6.5	Schematic Grouping	84
6.5.1	Group Breaking	84
6.6	Schematic Printing	85
6.6.1	Printing a Single Schematic Sheet	85
6.6.2	Printing a Hierarchical Schematic	85
6.7	File Operations	85
6.7.1	Saving	86
6.7.2	Exporting Schematic Graphics	86
6.7.3	Exporting to Earlier Versions of SIMetrix	86
6.7.4	Binary format	87
6.7.5	Autosave	87
6.8	Properties	87
6.8.1	Overview	87
6.8.2	What is a Property?	87
6.8.3	Template Property	89

6.8.4	Editing Properties in a Schematic	89
6.8.5	Restoring Properties	89
6.9	Template Property	90
6.9.1	Overview	90
6.9.2	Template Property Format	90
6.9.3	Template Scripts	99
6.10	Load Component Values	99
6.10.1	Overview	100
6.10.2	Menus	100
6.10.3	Configuration File Format	100
6.10.4	Creating a Configuration File from the Schematic	101
6.10.5	Loading a Configuration File	101
6.10.6	Automating Load Component Values with Command line Switches	102
6.10.7	Automating Load Component Values with a Script	102
6.11	CMOS IC Design	102
6.11.1	MOSFET Symbols	102
6.11.2	Edit Length and Width	104
6.11.3	Automatic Area and Perimeter Calculation	104
6.11.4	Editing the MOS Symbols	104
6.12	Further Information	105
6.12.1	How Symbols are Stored	105
6.12.2	Summary of Simulator Devices	105
<b>7</b>	<b>Symbol Editor and Library</b>	<b>109</b>
7.1	Creating Schematic Symbols	109
7.2	Graphical Symbol Editor	109
7.2.1	Notes	109
7.2.2	Symbol Editor Window	109
7.2.3	The Elements of a Symbol	110
7.2.4	Creating a New Symbol	110
7.2.5	Editing an Existing Symbol	111
7.2.6	Drawing Straight Line Segments	111
7.2.7	Drawing Arcs, Circles and Ellipses	111
7.2.8	Placing and Defining Pins	111
7.2.9	Defining Properties	114
7.2.10	Saving Symbols	117
7.3	Creating a Symbol from a Script	119
7.4	Symbol Library Manager	119
7.4.1	Operations	120
7.4.2	Editing System Symbol Libraries	122
<b>8</b>	<b>Parts</b>	<b>123</b>
8.1	How to Find and Place Parts	123
8.2	Part Selector	124
8.3	Part Search	125
8.4	Model Library Browser	125
8.4.1	Selecting a Model by Specification	126
8.4.2	Viewing and Editing Models	127
8.5	Numbered Parts in SIMPLIS	129
8.6	Circuit Stimulus	129
8.6.1	Waveform Generator	129
8.6.2	PWL Source	130
8.6.3	Power Supply/Fixed Current Source	131
8.6.4	AC Source	131
8.6.5	Universal Source	131
8.6.6	Digital Sources	145

8.6.7	Other Sources . . . . .	146
8.7	Generic Parts . . . . .	147
8.7.1	SIMPLIS Primitive Parts . . . . .	148
8.7.2	Saturable Inductors and Transformers . . . . .	149
8.7.3	Ideal Transformers . . . . .	150
8.7.4	DC Transformers . . . . .	151
8.7.5	Coupling Factor . . . . .	152
8.7.6	Mutual Inductors . . . . .	152
8.7.7	Resistors . . . . .	153
8.7.8	Ideal Capacitors and Inductors . . . . .	154
8.7.9	Infinite Capacitors and Inductors . . . . .	154
8.7.10	Multi-Level Capacitors . . . . .	155
8.7.11	Electrolytic Capacitors . . . . .	159
8.7.12	Potentiometer . . . . .	161
8.7.13	Lossless Transmission Line . . . . .	161
8.7.14	Lossy Transmission Line . . . . .	162
8.7.15	Fixed Voltage and Current Sources . . . . .	162
8.7.16	Controlled Sources . . . . .	162
8.7.17	Voltage Controlled Switch . . . . .	162
8.7.18	Switch with Hysteresis . . . . .	163
8.7.19	Delayed Switch . . . . .	164
8.7.20	Parameterised Opamp . . . . .	164
8.7.21	Parameterised Opto-coupler . . . . .	165
8.7.22	Parameterised Comparator . . . . .	166
8.7.23	VCO . . . . .	166
8.7.24	Verilog-A Library . . . . .	166
8.7.25	ISO 16750-2 and 7637-2 Waveform Sources . . . . .	167
8.7.26	Generic ADCs and DACs . . . . .	168
8.7.27	Generic Digital Devices . . . . .	170
8.7.28	Arbitrary Non-linear Passive Devices . . . . .	170
8.8	Creating Models . . . . .	170
8.8.1	Overview . . . . .	170
8.8.2	Creating Soft Recovery Diode Models . . . . .	171
8.8.3	Creating Power Inductor Models . . . . .	172
8.8.4	Digitising Data Sheet Curves . . . . .	175
8.9	Subcircuits . . . . .	181
8.9.1	Overview . . . . .	181
8.9.2	Creating a Sub-circuit from a Schematic . . . . .	182
8.9.3	Calling a Sub-circuit in a Schematic . . . . .	183
8.10	Special Parts . . . . .	185
8.10.1	Initial Conditions . . . . .	185
8.10.2	Nodesets . . . . .	185
8.11	Parameters and Expressions . . . . .	186
8.11.1	Example . . . . .	186
<b>9</b>	<b>Analog Behavioural Modelling</b> . . . . .	<b>188</b>
9.1	Introduction . . . . .	188
9.2	Non-linear Transfer Function . . . . .	188
9.2.1	Expression . . . . .	190
9.2.2	Examples . . . . .	197
9.2.3	Verilog-A Implementation . . . . .	198
9.3	Laplace Transfer Function . . . . .	198
9.3.1	Overview . . . . .	199
9.3.2	User Interface . . . . .	199
9.3.3	Laplace Expression . . . . .	201
9.3.4	Device type . . . . .	204

9.3.5	Implementation . . . . .	204
9.3.6	Impulse Response . . . . .	206
<b>10</b>	<b>Device Library, Parts Management</b>	<b>209</b>
10.1	Using Model Library Browser . . . . .	209
10.2	Installing Models . . . . .	209
10.2.1	Overview . . . . .	210
10.2.2	Procedure . . . . .	210
10.2.3	Full Model Installation Procedure . . . . .	210
10.2.4	Removing Model Libraries . . . . .	213
10.3	Part Selector . . . . .	213
10.3.1	Overview . . . . .	213
10.3.2	Category Editor . . . . .	213
10.3.3	Editing the Part Selector Catalog . . . . .	215
10.4	Advanced Topics . . . . .	216
10.4.1	Associating Multiple Models with Symbols . . . . .	216
10.4.2	Embedded Association . . . . .	217
10.4.3	Catalog Files . . . . .	218
10.4.4	Importing Models to a Schematic . . . . .	219
10.5	Sundry Topics . . . . .	219
10.5.1	.LIB Control . . . . .	219
10.5.2	Drag and Drop to Schematic . . . . .	219
10.5.3	Library Diagnostics . . . . .	220
10.5.4	Local Models . . . . .	220
10.5.5	Library Indexing Mechanism . . . . .	220
10.5.6	Duplicate Model Names . . . . .	220
<b>11</b>	<b>Analysis Modes</b>	<b>222</b>
11.1	Running Simulations . . . . .	223
11.1.1	Overview . . . . .	223
11.1.2	Starting, Pausing and Aborting Analyses . . . . .	223
11.1.3	Running Analyses in Asynchronous Mode . . . . .	224
11.1.4	Running an Analysis on a Netlist . . . . .	224
11.1.5	Simulation and Multi-core Processors . . . . .	224
11.2	Transient Analysis . . . . .	225
11.2.1	Setting up a Transient Analysis . . . . .	225
11.2.2	Restarting a Transient Run . . . . .	228
11.2.3	Transient Snapshots . . . . .	228
11.3	Operating Point . . . . .	230
11.4	Sweep Modes . . . . .	230
11.4.1	Device Sweep . . . . .	231
11.4.2	Temperature . . . . .	231
11.4.3	Model Parameter . . . . .	231
11.4.4	Parameter . . . . .	232
11.4.5	Frequency . . . . .	233
11.4.6	Monte Carlo . . . . .	233
11.4.7	Sensitivity/Worst-case . . . . .	234
11.4.8	Setting up a Swept Analysis . . . . .	234
11.5	DC Sweep . . . . .	236
11.5.1	Setting up a DC sweep . . . . .	236
11.5.2	Example . . . . .	237
11.6	AC Sweep . . . . .	238
11.6.1	Setting up an AC sweep . . . . .	238
11.6.2	Example . . . . .	239
11.7	Noise Analysis . . . . .	241
11.7.1	Setting up an AC Noise Analysis . . . . .	241

11.7.2	Plotting Results of Noise Analysis . . . . .	242
11.7.3	Example 1 . . . . .	242
11.7.4	Example 2 . . . . .	244
11.8	Real Time Noise . . . . .	246
11.8.1	Setting Up a Real Time Noise Analysis . . . . .	246
11.9	Transfer Function . . . . .	247
11.9.1	Setting up a Transfer Function Analysis . . . . .	248
11.9.2	Plotting Transfer Function Analysis Results . . . . .	249
11.9.3	Example . . . . .	249
11.10	DC Sensitivity . . . . .	250
11.10.1	Setting up a DC Sensitivity Analysis . . . . .	250
11.11	Convergence . . . . .	251
11.11.1	Convergence Options . . . . .	251
11.11.2	Convergence Failure Report . . . . .	254
11.11.3	Convergence Static Analysis . . . . .	254
11.12	Data Handling and Keeps . . . . .	255
11.12.1	Scope Options . . . . .	256
11.12.2	Signals Options . . . . .	256
11.12.3	To Add a Voltage Keep to a Schematic . . . . .	257
11.12.4	To Add a Current Keep to a Schematic . . . . .	257
11.13	Simulator Options . . . . .	257
11.13.1	Setting Simulator Options . . . . .	257
11.14	Multi-step Analyses . . . . .	259
11.14.1	Setting up a Multi-step Analysis . . . . .	260
11.14.2	Using Multiple Cores for Multi-step Analyses . . . . .	262
11.14.3	Example 1 . . . . .	264
11.14.4	Example 2 . . . . .	266
11.15	Safe Operating Area Testing . . . . .	268
11.15.1	Overview . . . . .	268
11.15.2	Defining Simple Limit Tests . . . . .	268
11.15.3	Advanced SOA Limit Testing . . . . .	269
<b>12</b>	<b>SIMPLIS Analysis Modes</b> . . . . .	<b>270</b>
12.1	Transient Analysis . . . . .	271
12.1.1	Setting up a Transient Analysis . . . . .	271
12.2	Periodic Operating Point (POP) . . . . .	273
12.2.1	Setting up a POP Analysis . . . . .	273
12.3	AC Analysis . . . . .	276
12.3.1	Setting up an AC Analysis . . . . .	276
12.4	SIMPLIS Options . . . . .	277
12.5	Multi-step and Monte Carlo . . . . .	277
12.5.1	Overview . . . . .	277
12.5.2	Comparison Between SIMetrix and SIMPLIS . . . . .	278
12.5.3	Setting up a SIMPLIS Multi-step Parameter Analysis . . . . .	278
12.5.4	Setting Up a SIMPLIS Monte Carlo Analysis . . . . .	280
12.5.5	Multi-core Multi-step SIMPLIS Analyses . . . . .	280
12.5.6	Tolerances and Distribution Functions . . . . .	282
12.5.7	Monte Carlo Log File . . . . .	282
12.5.8	Monte Carlo Seed Values . . . . .	283
12.5.9	Performance Analysis and Histograms . . . . .	284
12.6	Back-Annotation . . . . .	285
12.6.1	Overview . . . . .	285
12.6.2	How to Back-annotate a Schematic . . . . .	285
12.6.3	Disable/Enable Initial Conditions . . . . .	285
12.6.4	Back-annotation Errors . . . . .	285
12.6.5	Editing Back-annotated Initial Conditions . . . . .	285

12.6.6	How Does it Work? . . . . .	286
12.6.7	Hierarchical Blocks and Subcircuits . . . . .	286
<b>13</b>	<b>Graphs, Probes and Data Analysis</b>	<b>287</b>
13.1	Elements of the Graph Window . . . . .	287
13.1.1	Main Window . . . . .	287
13.1.2	Windows and Tabbed Sheets . . . . .	287
13.1.3	Graph Toolbar . . . . .	288
13.2	Fixed vs. Random . . . . .	288
13.3	Fixed Probes . . . . .	288
13.3.1	Fixed Probe Options . . . . .	290
13.3.2	Curve History . . . . .	294
13.3.3	Fixed Bus Probe Options . . . . .	295
13.3.4	Using Fixed Probes in Hierarchical Designs . . . . .	295
13.3.5	Adding Fixed Probes After a Run has Started . . . . .	295
13.3.6	Per-cycle Probes . . . . .	296
13.3.7	XY Probes . . . . .	304
13.3.8	Bode Plot Probe with Measurements . . . . .	310
13.3.9	Bode Plot Probe - Basic Version . . . . .	313
13.3.10	Fourier Analysis . . . . .	316
13.3.11	Arbitrary Fixed Probe . . . . .	319
13.3.12	Changing Update Period and Start Delay . . . . .	322
13.3.13	Auto Probe Colouring . . . . .	322
13.4	Random Probes . . . . .	322
13.4.1	General Behaviour . . . . .	322
13.4.2	Functions . . . . .	322
13.4.3	Notes on Probe Functions . . . . .	324
13.4.4	Plotting Noise Analysis Results . . . . .	324
13.4.5	Plotting Transfer Function Analysis Results . . . . .	325
13.4.6	Fourier Analysis . . . . .	325
13.4.7	Probing Buses . . . . .	329
13.4.8	Bus Probe Options . . . . .	329
13.4.9	Plotting an Arbitrary Expression . . . . .	331
13.4.10	Curve Arithmetic . . . . .	336
13.4.11	Using Random Probes in Hierarchical Designs . . . . .	336
13.4.12	Previous Results . . . . .	338
13.5	Updating Curves . . . . .	340
13.5.1	Overview . . . . .	340
13.5.2	Update Curves . . . . .	340
13.5.3	Plot Journals . . . . .	341
13.6	Graph Layout . . . . .	341
13.6.1	AutoAxis Feature . . . . .	343
13.6.2	Manually Creating Axes and Grids . . . . .	344
13.6.3	Selecting Axes . . . . .	344
13.6.4	Stacking Curves to Multiple Grids . . . . .	344
13.6.5	Moving Curves to Different Axis or Grid . . . . .	344
13.6.6	Detached X-Axes . . . . .	344
13.6.7	Deleting Axes . . . . .	345
13.6.8	Editing Axes . . . . .	345
13.6.9	Reordering Grids and Digital Axes . . . . .	345
13.7	Curve Operations . . . . .	345
13.7.1	Selecting Curves . . . . .	345
13.7.2	Deleting Curves . . . . .	346
13.7.3	Hiding and Showing Curves . . . . .	346
13.7.4	Re-titling Curves . . . . .	346
13.7.5	Highlighting Curves . . . . .	346

13.7.6	Moving and Copying Curves Between Graphs . . . . .	346
13.8	Graph Cursors . . . . .	347
13.8.1	Overview . . . . .	347
13.8.2	Cursor Operations . . . . .	348
13.8.3	Additional Cursors . . . . .	349
13.8.4	Cursor Readout . . . . .	350
13.8.5	Cursor Functions . . . . .	352
13.9	Curve Measurements . . . . .	352
13.9.1	Overview . . . . .	352
13.9.2	Available Measurements . . . . .	352
13.9.3	Using the Define Measurement GUI . . . . .	352
13.9.4	Measurement Definitions Manager . . . . .	354
13.9.5	Repeating the Same Measurement . . . . .	356
13.9.6	Applying Measurements to Fixed Probes . . . . .	356
13.9.7	Notes on Curve Measurement Algorithms . . . . .	357
13.9.8	Plots from curves . . . . .	359
13.10	Efficiency Calculator . . . . .	359
13.10.1	Overview . . . . .	359
13.10.2	Step1: Place Power Probes . . . . .	360
13.10.3	Step 2: Define Input and Output probes . . . . .	361
13.10.4	Step 3: Calculate Efficiency . . . . .	361
13.11	Zoom and Scroll . . . . .	361
13.11.1	Zooming with the Mouse . . . . .	361
13.11.2	Zooming with the Keyboard . . . . .	361
13.11.3	Recovering an Earlier Zoom . . . . .	362
13.11.4	Scrolling with the Keyboard . . . . .	362
13.11.5	Zooming with Multiple Grids . . . . .	362
13.11.6	Adding new Curves to a Zoomed Grid . . . . .	362
13.11.7	Zoom to Fit Y-axis . . . . .	362
13.12	Annotating a Graph . . . . .	362
13.12.1	Curve Markers . . . . .	362
13.12.2	Legend Box . . . . .	364
13.12.3	Text Box . . . . .	365
13.12.4	Caption and Free Text . . . . .	365
13.12.5	Graph Symbolic Values . . . . .	366
13.13	Exporting/Importing . . . . .	368
13.13.1	Clipboard . . . . .	368
13.13.2	Exporting Graphics . . . . .	369
13.13.3	Data Import and Export . . . . .	369
13.14	Saving Graphs . . . . .	372
13.14.1	Saving . . . . .	372
13.14.2	Restoring . . . . .	372
13.15	Saving Data . . . . .	372
13.15.1	Saving the Data of a Simulation . . . . .	372
13.15.2	Restoring Simulation Data . . . . .	373
13.16	DC Operating Point . . . . .	373
13.16.1	Schematic Annotation . . . . .	373
13.16.2	Displaying Device Operating Point Info . . . . .	373
13.16.3	List File Data . . . . .	373
13.16.4	Other Methods of Obtaining Bias Data . . . . .	373
13.16.5	Bias Annotation in SIMPLIS . . . . .	374
13.16.6	Bias Annotation Display Precision . . . . .	374
13.16.7	Bias Annotation and Long Transient Runs . . . . .	374
13.17	Performance Analysis . . . . .	374
13.17.1	Overview . . . . .	374
13.17.2	Example . . . . .	374

13.17.3 Histograms . . . . .	376
13.17.4 Goal Functions . . . . .	379
<b>14 The Command Shell</b>	<b>389</b>
14.1 Command Line . . . . .	389
14.1.1 Command History . . . . .	389
14.1.2 Message Window . . . . .	390
14.1.3 Multiple commands on one line . . . . .	390
14.1.4 Scripts . . . . .	390
14.1.5 Command Line Editing . . . . .	390
14.1.6 Command Line Switches . . . . .	390
14.2 Editing the Menu System . . . . .	390
14.2.1 Overview . . . . .	390
14.2.2 Procedure . . . . .	391
14.3 User Defined Toolbars, Buttons . . . . .	393
14.4 Message Window . . . . .	393
<b>15 Command–Function Reference</b>	<b>394</b>
15.1 Introduction . . . . .	394
15.1.1 Notation . . . . .	394
15.2 Command Summary . . . . .	395
15.3 Reference . . . . .	395
15.3.1 OpenGroup . . . . .	395
15.3.2 Set . . . . .	396
15.3.3 Show . . . . .	396
15.3.4 Unset . . . . .	397
15.4 Function Summary . . . . .	397
15.5 Function Reference . . . . .	399
15.5.1 abs(real/complex) . . . . .	399
15.5.2 arg(real/complex) . . . . .	399
15.5.3 arg_rad(real/complex) . . . . .	399
15.5.4 atan(real/complex) . . . . .	400
15.5.5 cos(real/complex) . . . . .	400
15.5.6 db(real/complex) . . . . .	400
15.5.7 diff(real) . . . . .	400
15.5.8 exp(real/complex) . . . . .	400
15.5.9 fft(real [, string]) . . . . .	400
15.5.10 FIR(real, real [, real]) . . . . .	400
15.5.11 Floor(real) . . . . .	401
15.5.12 GroupDelay(real/complex) . . . . .	401
15.5.13 Histogram(real, real) . . . . .	402
15.5.14 Iff(real, any, any) . . . . .	403
15.5.15 IIR(real, real [, real]) . . . . .	403
15.5.16 im(real/complex), imag(real/complex) . . . . .	404
15.5.17 integ(real) . . . . .	404
15.5.18 Interp(real, real [, real, real]) . . . . .	404
15.5.19 IsComplex(any) . . . . .	404
15.5.20 length(any) . . . . .	405
15.5.21 ln(real/complex) . . . . .	405
15.5.22 log10(real/complex), log(real/complex) . . . . .	405
15.5.23 mag(real/complex) , magnitude(real/complex) . . . . .	406
15.5.24 maxidx(real/complex) . . . . .	406
15.5.25 Maxima(real [, real, string]) . . . . .	406
15.5.26 Maximum(real/complex [, real, real]) . . . . .	406
15.5.27 mean(real/complex) . . . . .	407
15.5.28 Mean1(real [, real, real]) . . . . .	407

15.5.29	minidx(real/complex)	407
15.5.30	Minima(real [, real, string])	407
15.5.31	Minimum(real/complex [, real, real])	407
15.5.32	norm(real/complex)	407
15.5.33	ph(real/complex) , phase(real/complex)	408
15.5.34	phase_rad(real/complex)	408
15.5.35	Range(real/complex [, real, real])	408
15.5.36	re(real/complex), real(real/complex)	408
15.5.37	Ref(real/complex)	408
15.5.38	Rms(real)	408
15.5.39	RMS1(real [, real, real])	408
15.5.40	rnd(real)	408
15.5.41	RootSumOfSquares(real [, real, real])	409
15.5.42	sign(real)	409
15.5.43	sin(real/complex)	409
15.5.44	sqrt(real/complex)	409
15.5.45	SumNoise(real [, real, real])	409
15.5.46	tan(real/complex)	409
15.5.47	Truncate(real [, real, real])	409
15.5.48	unitvec(real)	410
15.5.49	vector(real)	410
15.5.50	XFromY(real, real [, real, real])	410
15.5.51	XY(real, real)	410
15.5.52	YFromX(real, real [, real])	410
<b>16</b>	<b>Monte Carlo, Sensitivity and Worst-case</b>	<b>411</b>
16.1	A Monte Carlo Example	411
16.2	Sensitivity/Worst-case Example	415
16.3	Part Tolerance Specification	418
16.3.1	Setting Device Tolerances	418
16.3.2	Model Tolerances	419
16.3.3	Matching Devices	419
16.3.4	Random Distribution	420
16.4	Running Monte Carlo	420
16.4.1	Overview	420
16.4.2	Setting up a Multi Step Monte Carlo Run	421
16.4.3	Setting up a Single Step Monte Carlo Sweep	422
16.4.4	Setting the Seed Value	422
16.5	Running Sensitivity and Worst-case	423
16.5.1	Overview	423
16.5.2	Setting up a Multi-step Sensitivity Analysis	424
16.5.3	Setting up a Single Step Sensitivity Analysis	425
16.6	Analysing Monte-Carlo Results	425
16.6.1	Plots	425
16.6.2	Creating Histograms	426
16.6.3	Log File	426
16.7	Analysing Sensitivity and Worst-case Results	426
16.7.1	Plots	426
16.7.2	Reports	427
<b>17</b>	<b>Verilog-HDL Simulation</b>	<b>430</b>
17.1	Overview	430
17.2	Documentation	430
17.3	Supported Verilog Simulators	430
17.4	Basic Operation	430
17.5	Using Verilog-HDL in SIMetrix Schematics	431

17.5.1	Creating Schematic Symbols . . . . .	431
17.5.2	Editing Parameters . . . . .	431
17.6	Module Cache . . . . .	431
17.6.1	Operation . . . . .	432
17.7	Simulation Options . . . . .	432
17.7.1	Verilog Simulator . . . . .	432
17.7.2	Timing Resolution . . . . .	432
17.7.3	Open Console for Verilog Process . . . . .	432
17.8	Tutorial . . . . .	433
17.8.1	Procedure . . . . .	434
17.9	Verilog Simulator Interface . . . . .	437
17.9.1	VPI . . . . .	437
17.9.2	Interface Configuration . . . . .	438
17.9.3	Launch Script . . . . .	438
17.9.4	Verilog Simulation Preparation . . . . .	438
<b>18</b>	<b>Sundry Topics</b>	<b>439</b>
18.1	Saving and Restoring Sessions . . . . .	439
18.1.1	Overview . . . . .	439
18.1.2	Saving a Session . . . . .	439
18.1.3	Restoring a Session . . . . .	439
18.1.4	Where is Session Data Stored? . . . . .	439
18.2	Symbolic Path Names . . . . .	440
18.2.1	Overview . . . . .	440
18.2.2	Definition . . . . .	440
18.2.3	Configuration File Example . . . . .	441
18.2.4	Using Symbolic Names . . . . .	441
18.3	SIMetrix Command Line Parameters . . . . .	442
18.3.1	Using startup.ini . . . . .	443
18.4	Configuration Settings . . . . .	443
18.4.1	Overview . . . . .	443
18.4.2	Default Configuration Location . . . . .	444
18.4.3	Application Data Directory . . . . .	444
18.4.4	Specifying Other Locations for Config Settings . . . . .	444
18.5	Options . . . . .	444
18.5.1	Overview . . . . .	445
18.5.2	Using the Options Dialog . . . . .	445
18.5.3	Using the Set and Unset commands . . . . .	453
18.6	Startup Auto Configuration . . . . .	470
18.6.1	Overview . . . . .	470
18.6.2	What is Set Up . . . . .	470
18.6.3	Auto Configuration Options . . . . .	470
18.6.4	Installation - Customising . . . . .	472
18.7	Colours and Fonts . . . . .	472
18.7.1	Colours . . . . .	472
18.7.2	Fonts . . . . .	472
18.8	Startup Script . . . . .	473
18.9	International Character Sets . . . . .	473
18.9.1	Character Encoding . . . . .	473
18.9.2	SIMetrix Character Encoding . . . . .	473
18.9.3	Compatibility with Older Versions . . . . .	474
18.9.4	Text Editors . . . . .	474
18.10	Installation and Licensing . . . . .	474
18.10.1	Install CD . . . . .	474

# Chapter 1

## Welcome

For a simple tutorial - see [“Tutorial Introduction”](#) on page 7

For basic guidance on the main features - see [“Getting Started”](#) on page 31

For more detailed documentation refer to one of the following:

- [“Schematic Editor”](#) on page 54
- [“GUI Environment”](#) on page 47
- [“SIMetrix Simulator Analysis Modes”](#) on page 222
- [“SIMPLIS Simulator Analysis Modes”](#) on page 270
- [“Installing Models”](#) on page 209
- [“Finding and Placing Parts”](#) on page 123
- [“Waveform Viewer”](#) on page 287

## Chapter 2

# Introduction

## 2.1 About SIMetrix/SIMPLIS Documentation

### 2.1.1 HTML Documentation

All SIMetrix and SIMPLIS documentation is available in HTML form and may be viewed through your web browser. This can be accessed through the menus **Help | SIMetrix Documentation** and **Help | SIMPLIS Documentation**.

You can select whether you access this documentation on-line via the Internet or locally stored on your system. Documentation on-line has the benefit of always being the latest version but has the drawback of requiring a fast Internet connection. Local documentation has the benefit of being always available even if there is no Internet connection, but may become out of date.

To choose the documentation source, select menu **Help | Select Help Source...**. If you opt for local help and you have not selected this previously, the help files will first need to be installed. This operation is performed automatically but will take a few minutes.

### 2.1.2 PDF Documentation

Much of the documentation is also available in the form of PDF manuals. The PDF manuals may be accessed locally through the menus under **Help | PDF Manuals** or can be downloaded. These are described below:

Manual name and web link	Description
<i>User's Manual</i>	General operation of SIMetrix and SIMPLIS
<i>Simulator Reference Manual</i>	Detailed reference for the SIMetrix simulator
<i>SIMPLIS Reference Manual</i>	Detailed reference for the SIMPLIS simulator
<i>Script Reference Manual</i>	SIMetrix script language reference manual
<i>Verilog-A Reference Manual</i>	Verilog-A language reference manual

## 2.2 What is Simetrix?

SIMetrix is a mixed-signal circuit simulator designed for ease and speed of use.

The core algorithms employed by the SIMetrix analog simulator are based on the SPICE program developed by the CAD/IC group at the department of Electrical Engineering and Computer Sciences, University of California at Berkeley. The digital event driven simulator is derived from XSPICE developed by the Computer Science and Information Technology Laboratory, Georgia Tech. Research Institute, Georgia Institute of Technology.

Although originally derived from these programs only a tiny fraction of the overall application code can be traced to them. Nearly all of the simulator code is either new or has been rewritten in order to provide new analysis features and to resolve convergence problems.

### Features

- Closely coupled direct matrix analog and event driven digital simulator.
- Fully integrated hierarchical schematic editor, simulator and graphical post-processor.
- Superior convergence for both DC and transient analyses.
- Advanced swept analyses for AC, DC, Noise and transfer function. 6 different modes available.
- Real time noise analysis allowing noise simulation of oscillators and sampled data systems.
- Support for IC design models such as BSIM3/4, VBIC and Hicem.
- Cross probing of voltage, current and device power from schematic. Current and power available for sub-circuits.
- Monte Carlo analysis including comprehensive tolerance specification features.
- Full featured scripting language allowing customised waveform analysis and automated simulation
- Verilog-A Analog Hardware Description Language
- Mixed signal simulation using Verilog-HDL
- Functional modelling with arbitrary non-linear source and arbitrary linear s-domain transfer function.
- Arbitrary logic block for definition of any digital device using descriptive language. Supports synchronous, asynchronous and combinational logic as well as ROMs and RAMs.
- Models for saturable magnetic parts including support for air-gaps.
- User definable fixed and popup menus and key definitions.

## 2.3 What is SIMPLIS?

SIMPLIS is a circuit simulator designed for rapid modelling of switching power systems. An acronym for “SIMulation for Piecewise LInear System”, it is supplied with our SIMetrix/SIMPLIS product range.

SIMPLIS is a component level simulator like SPICE but is typically 10 to 50 times faster when simulating switching circuits. It achieves its speed by modelling devices using a series of straight-line segments rather than solving non-linear equations as SPICE does. By modelling devices in this way, SIMPLIS can characterise a complete system as a cyclical sequence of linear circuit topologies. This is an accurate representation of a typical switching power system where the semiconductor devices function as switches. However, a linear system can be solved very much more rapidly than the non-linear systems that SPICE handles. The end result is accurate, but extremely fast simulations, allowing the modelling of complex topologies that would not be viable with SPICE.

SIMPLIS has three analysis modes: Transient, Periodic Operating Point and AC. Transient analysis is similar to the SPICE equivalent but is typically 10-50 times faster. Periodic Operating Point is a unique analysis mode that finds the steady-state operating waveforms of switching systems. AC analysis finds the frequency response of a switching system without needing to use averaged models. This is especially

useful for what-if studies on new circuit topologies or control schemes where the small-signal averaged model has not yet been derived.

Because non-linear devices are defined using a sequence of straight line segments, models for such devices are quite different from SPICE models. There are of course many SPICE models available and so in order to retain compatibility with these, SIMetrix/SIMPLIS has the ability to convert models for some types of device into SIMPLIS format. This conversion is performed when the device is placed on the schematic. Devices currently supported are MOSFETs, BJTs and diodes. In the case of MOSFETs and Zener diodes, the conversion is achieved by performing a sequence of simulations using the SIMetrix-SPICE simulator. This method is independent of the method of implementation of the device.

## 2.4 Why Simulate?

Integrated circuit designers have been using analog simulation software for nearly five decades. The difficulty of bread-boarding and high production engineering costs have made the use of such software essential.

For board level designers the case has not been so clear cut. For them prototyping is straightforward, inexpensive and generally provides an accurate assessment of how the final production version of a circuit will behave. By contrast computer simulation has been seen as slow and prone to inaccuracies stemming from imperfect models.

In recent years, however, the simulation of discrete analog circuits has become more viable. This has come about because of advances in CPU power, the increased availability of device models from their manufacturers and the introduction of easy to use and affordable simulation tools such as SIMetrix.

The pressure to reduce product development time-scales has meant that for many projects the traditional bread-boarding phase is skipped altogether - with or without simulation - and circuit development is carried out on the first revisions of PCB. The use of simulation on a circuit or parts of a circuit can help to eliminate errors in a circuit design prior to this stage and reduce the number of PCB revisions required before the final production version is reached. Of course, to be useful, the simulation process must therefore not be too time consuming.

Computer simulation, does however, have many more uses. There are some things you can do with a simulator which cannot be achieved with practical approaches. You can remove parasitic elements, you can make non-invasive measurements that are impossible in real-life or you can run components outside of their safe operating area. These abilities make simulation a valuable tool for finding out why a particular design does not behave as expected. If the problem can be reproduced on a simulator then its cause can be much more easily identified. Even if a problem cannot be reproduced then this gives some clues. It means that it is caused by something that is not modelled, a wiring parasitic perhaps.

Simulation is extremely useful for testing ideas at the system level. Sometimes it is not easy to test a concept because the hardware to implement it is very costly or time consuming to build. It may even be that you don't know how to implement the idea in hardware at all. The alternative is to design a model and simulate it with a computer. Once it has been established that the concept is viable then attention can be given to its implementation. If it proves not to be viable, then a great deal of time will have been saved.

## 2.5 System Requirements

### 2.5.1 Operating System

#### **SIMetrix and SIMetrix/SIMPLIS 64 bit version**

The following are supported:

Windows 10 Home, Pro and Enterprise - 64 bit editions

Windows 8.1, Windows 8.1 Pro, Windows 8.1 Enterprise - 64 bit editions

Windows 8, Windows 8 Pro, Windows 8 Enterprise - 64 bit editions

Windows 7 Home Premium/Professional/Enterprise/Ultimate - 64 bit editions

## SIMetrix and SIMetrix/SIMPLIS 32 bit versions

The following is supported

Windows 7 Home Premium/Professional/Enterprise/Ultimate

## Windows RT

SIMetrix and SIMetrix/SIMPLIS will **not** run on any version of Windows RT.

### 2.5.2 Hardware

SIMetrix will run satisfactorily on any system that meets all the following requirements:

1. The system is running one of supported operating systems listed above
2. The system meets the minimum hardware requirement for the operating system
3. The system's display has a minimum vertical resolution of 720 pixels and a minimum horizontal resolution of 1024 pixels

### 2.5.3 Recommended System

If you regularly run large circuit simulations or long runs on smaller circuits, we recommend investing in the most powerful CPU available. A large RAM system can be useful as this will allow caching of simulation data. This will speed up plotting results if a large amount of data is generated. The data is stored to disk in an efficient manner and therefore substantial RAM is not essential unless the circuits being simulated are very large indeed. A high performance bus mastering SCSI disk system will improve simulation performance a little.

### 2.5.4 Multi-core Processors

SIMetrix can exploit multiple core CPUs in a number of ways and will benefit from multiple core processors. Note, however, that you will require a SIMetrix Pro or SIMetrix Elite license to be able to use the multiple core features. Note also that SIMetrix cannot effectively use Hyper-threading.

For more information, see the following sections:

[“Simulation and Multi-core Processors” on page 224](#)

[“Using Multiple Cores for Multi-step Analyses” on page 262](#)

[“Multi-core Multi-step SIMPLIS Analyses” on page 280](#)

## 2.6 Acknowledgements

We acknowledge the following code sources and copyright owners:

1. SPICE3, BSIM3 and BSIM4: The Regents of the University of California at Berkeley

## *2.6. Acknowledgements*

2. XSpice: Computer Science and Information Technology Laboratory, Georgia Tech. Research Institute, Georgia Institute of Technology
3. KLU Matrix Solver: University of Florida Research Foundation, Inc.
4. HiSim HV Model: Hiroshima University & Semiconductor Technology Academic Research Center (STARC)
5. SimKit models; NXP Semiconductors

# Chapter 3

## Quick Start

### 3.1 Introduction

This chapter covers a number of tutorials that will help you get started with SIMetrix.

Tutorial 1 is designed for total novices. You may wish to skip to tutorial 2 if you already have experience with SPICE type programs.

Tutorial 2 assumes you have grasped the basics of using the schematic editor. You don't have to worry about setting up analyses or the characteristics of any input stimulus such as V2 in tutorial 1; these procedures will be explained.

If you are an experienced circuit designer but have never used a circuit simulator before, we recommend you read [“Simulation for the Novice” on page 7](#). This will familiarise you with a few concepts about simulation that may be alien to you if you are used to traditional methods of evaluating circuits.

#### 3.1.1 Examples and Tutorials - Where are They?

On Windows the example files are first installed under the main installation root (e.g. under C:\Program Files\SIMetrix820\support\examples) but it is not intended that they are used from that location. Instead they will be copied to your “My Documents” folder when SIMetrix starts for the first time, but only if you accept the option to do so. If you can't find the examples files, you may need to manually copy them from the installation root to a suitable location of your choice.

In the following tutorial discussions, the examples directory is referred to as 'EXAMPLES'.

#### 3.1.2 Simulation for the Novice

When measuring a real circuit, you would probably connect up a power source - bench power supply perhaps - maybe also some signal source, then switch it on and take measurements using - probably - an oscilloscope. You can also make adjustments to the circuit and see the effects straight away.

For simulation, you have a choice of analysis modes and not all of them have an exact real life equivalent. The analysis mode that most closely resembles the method of bench testing a circuit described above is *transient analysis*. This performs an analysis over a specified (by you) time period and saves all the results - i.e. all the voltages and currents in the circuit - to your hard disk. The difference between real life testing and simulation is that the simulation is not running all the time. If you want to see the effects of changing a part value, you have to change it then re-run the simulation. (But note there is a potentiometer device that automates this procedure see [“Potentiometer” on page 161](#)).

In order to solve the circuit, the simulator has to calculate the voltage at every node at every time point. Disk space is cheap and plentiful so SIMetrix saves all these values as well as the device currents. Not all simulators do this, some require you to state in advance what you want saved.

After the run is complete, you can then randomly probe the circuit to look at any voltage, current or device power over the analysis time period. You can also place fixed probes on the circuit before running the analysis which will cause the waveform at that point of the circuit to be automatically displayed while the simulation is running or optionally after its completion.

Some of the other analysis modes are: AC analysis which performs a frequency sweep, DC sweep which ramps a voltage or current source and noise analysis which calculates total noise at a specified point and which parts are responsible for that noise.

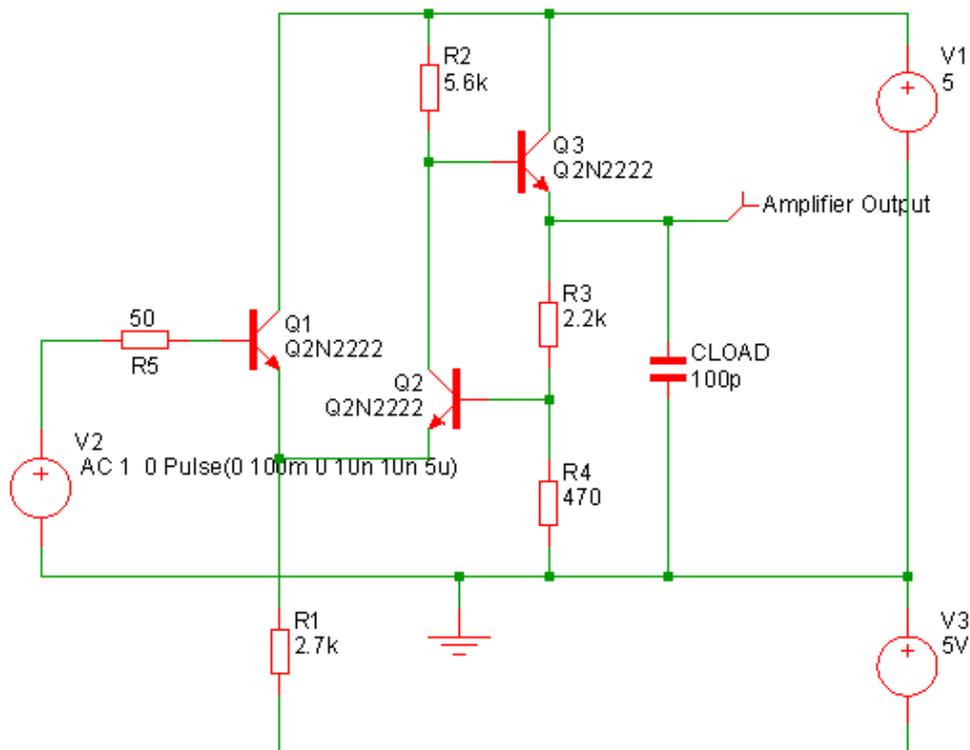
## 3.2 Tutorial 1 - A Simple Ready to Run Circuit

This tutorial demonstrates a basic simulation on a ready to run circuit. All you need to do is load the circuit and press **F9** to run it. We will then make a few changes to the circuit and make some extra plots.

This tutorial demonstrates the basic features without having to get into the details of setting up a simulation. Proceed as follows:

1. Select the menu **File | Open...** and if not already select **Schematic Files** in the file type drop down box. Select the schematic file TUTORIAL1 which you should find in the folder EXAMPLES/SIMetrix/Tutorial (See “[Examples and Tutorials - Where are They?](#)” on page 7). Click on **Open** to open this file.

A schematic window will open with the following circuit:



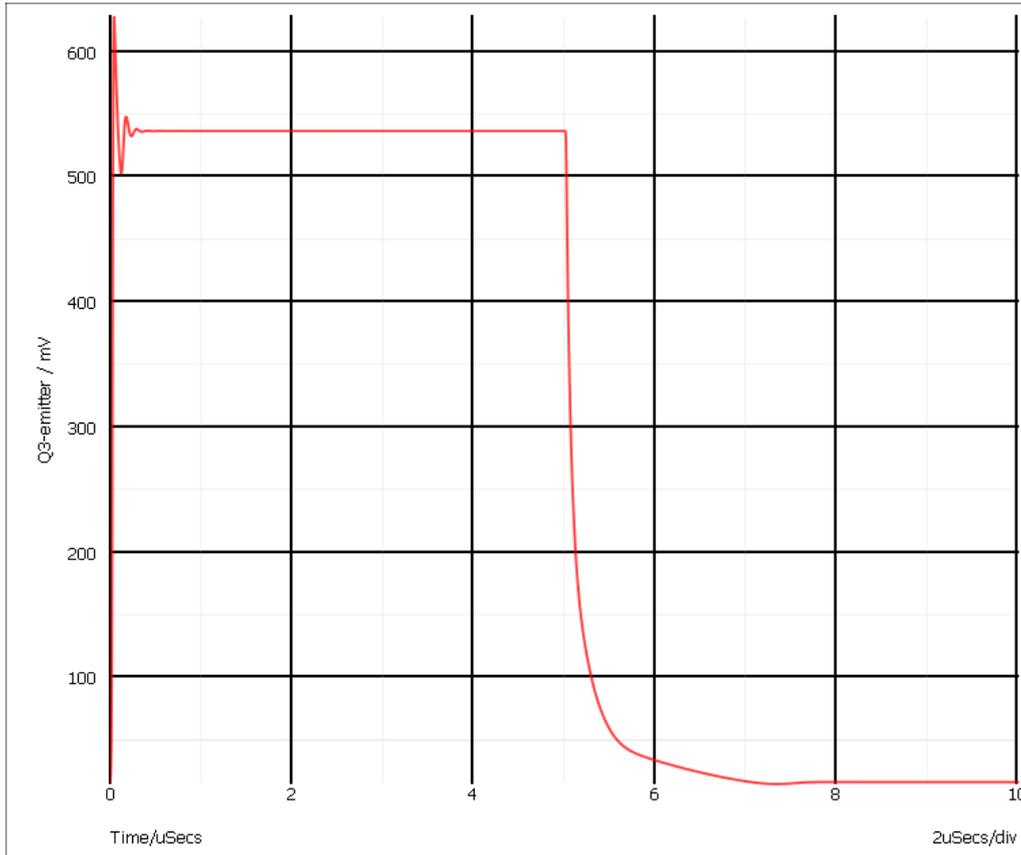
This is a simple feedback amplifier designed to amplify a 100mV pulse up to around 500mV. The

basic requirement of the design is that the pulse shape should be preserved, DC precision is important but is not critical. The above is our first attempt at a design but has not yet been optimised.

This example circuit has been setup to be 'ready to run'.

2. To start the simulation, select from the schematic window **Simulator | Run Schematic** or press **F9**. The simulation will not take long, on a modern machine less than half a second.

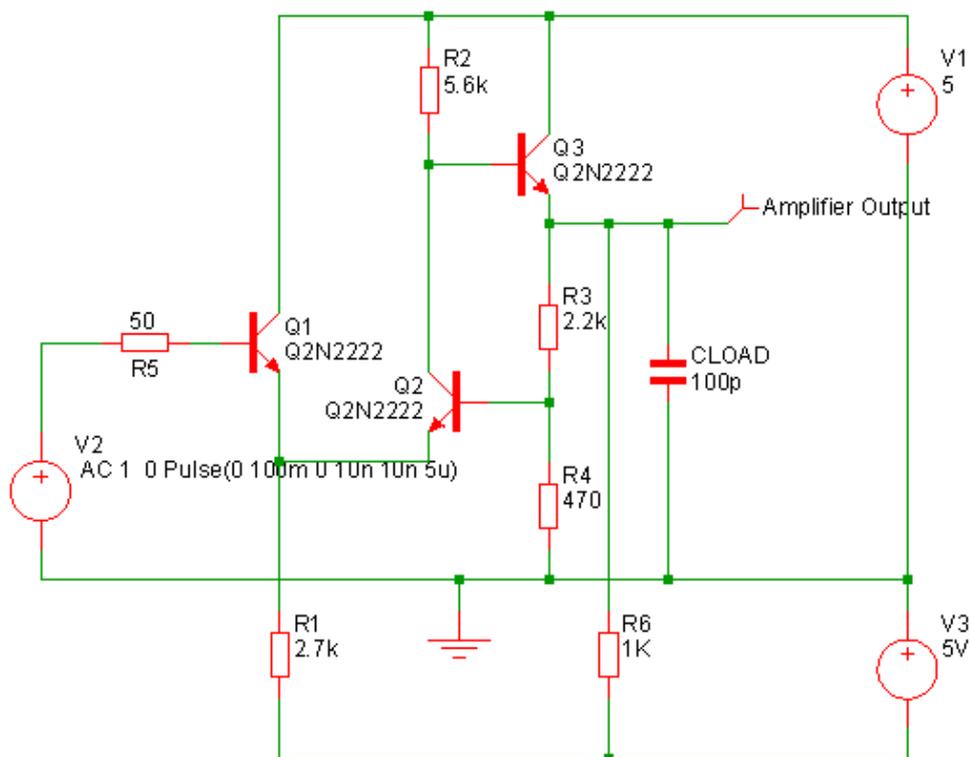
You will see a graph of the output voltage appear:



As can be seen, our amplifier doesn't work all that well. There are two problems.

1. There is substantial ringing on the rising edge, probably caused by the capacitive load.
2. The falling edge is somewhat sluggish.

The sluggish falling edge is caused by the absence of any standing current in the output emitter follower, Q3. To rectify this, we will place a resistor from the emitter to the -5V rail. The resulting schematic is shown below:



To make this modification, proceed as follows:

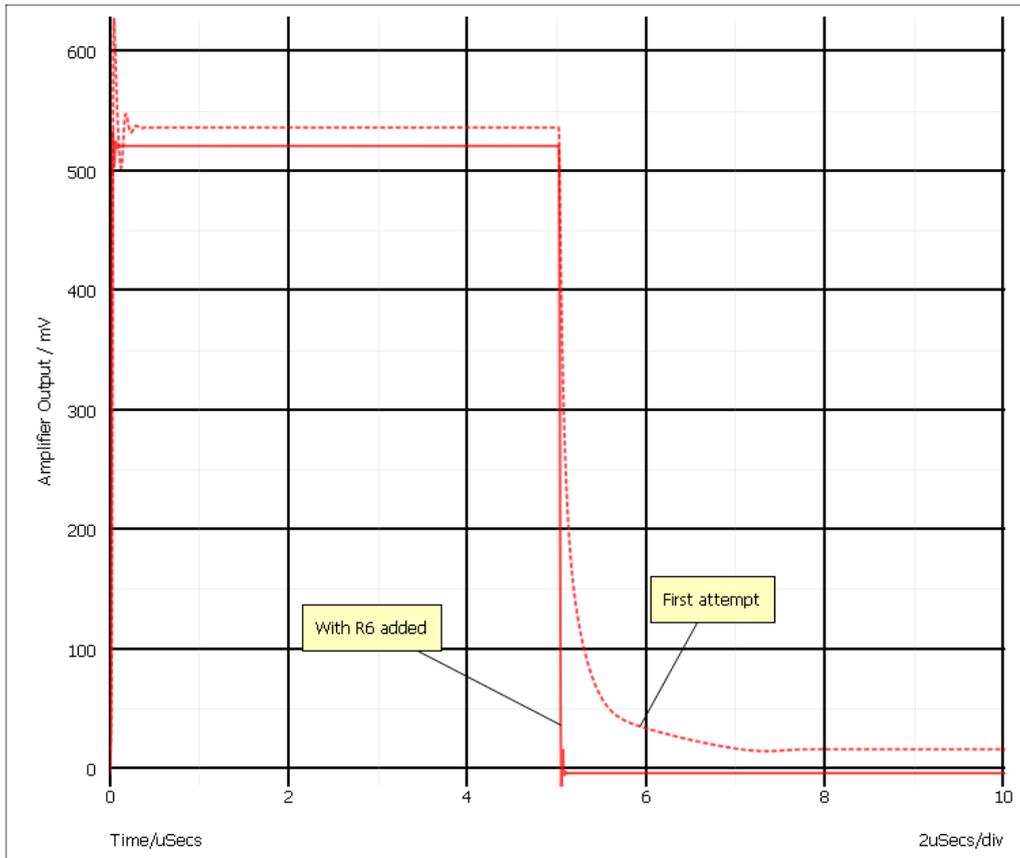
1. Press the Resistor button in the parts toolbar. Alternatively, select the menu **Place | Passives | Resistor (Box shape)** or, if you prefer, **Place | Passives | Resistor (Z shape)**. A resistor symbol will appear. Place this in the location shown in the diagram above. Click the left mouse button to fix it to the schematic. You may now see another resistor symbol appear (depending on how the system options are set). Cancel this by clicking the right mouse button.
2. Now wire up the resistor. There are a number of ways of doing this. If you have a three button mouse or wheel mouse, one way is to use the middle button (or wheel). Pressing it once will start a wire. Pressing it again will fix it and start a new one. Pressing the right button will terminate it.

If enabled you can also use the 'smart-wiring' method. Just take the mouse pointer to the pin of the resistor. You will see a pen symbol appear as the mouse gets close to the pin. Left click then move the mouse cursor to the destination then left click again. This method will automatically locate a route for the wire if one exists.

You can also enter wiring mode by selecting the toolbar wire button. This puts schematic into a permanent wiring mode where the left key is always used for wiring. Revert to normal mode by pressing the wire button again.

3. Re run the simulation by pressing **F9**.

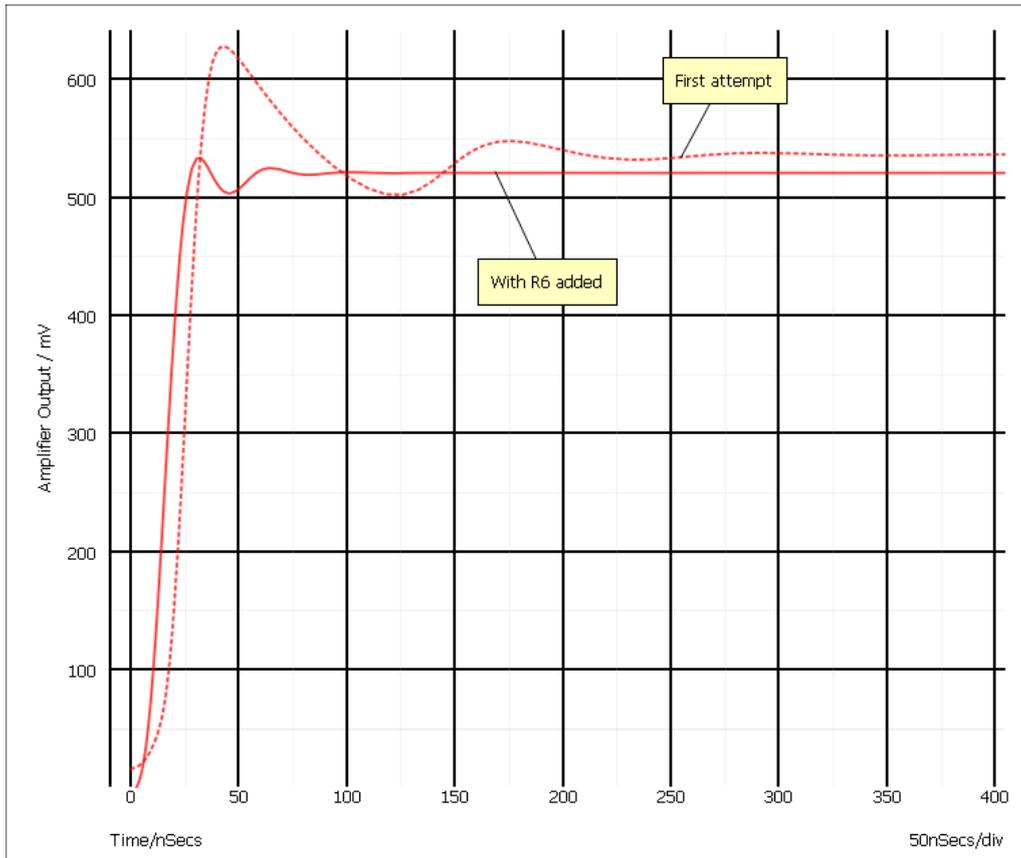
The graph will now be updated to:



As you can see, The problem with the trailing edge has been fixed and the ringing is much improved.

Now let's have a look at the ringing in more detail. To do this, we need to zoom in the graph by adjusting the limits of the axes. There are two ways of doing this. The quickest is to simply drag the mouse over the region of interest. The other method is to manually enter the limits using the Edit Axis Dialog Box. To zoom with the mouse, proceed as follows:

1. Make sure that the graph window is selected by clicking in its title bar.
2. Place the cursor at the top left of the region of interest i.e to the left of the y-axis and above the top of the red curve.
3. Press the left mouse key and while holding it down, drag the mouse to the bottom right of the area you wish to zoom in. You should see a rectangle appear as you drag the mouse.
4. Release the mouse key. You should see something like:



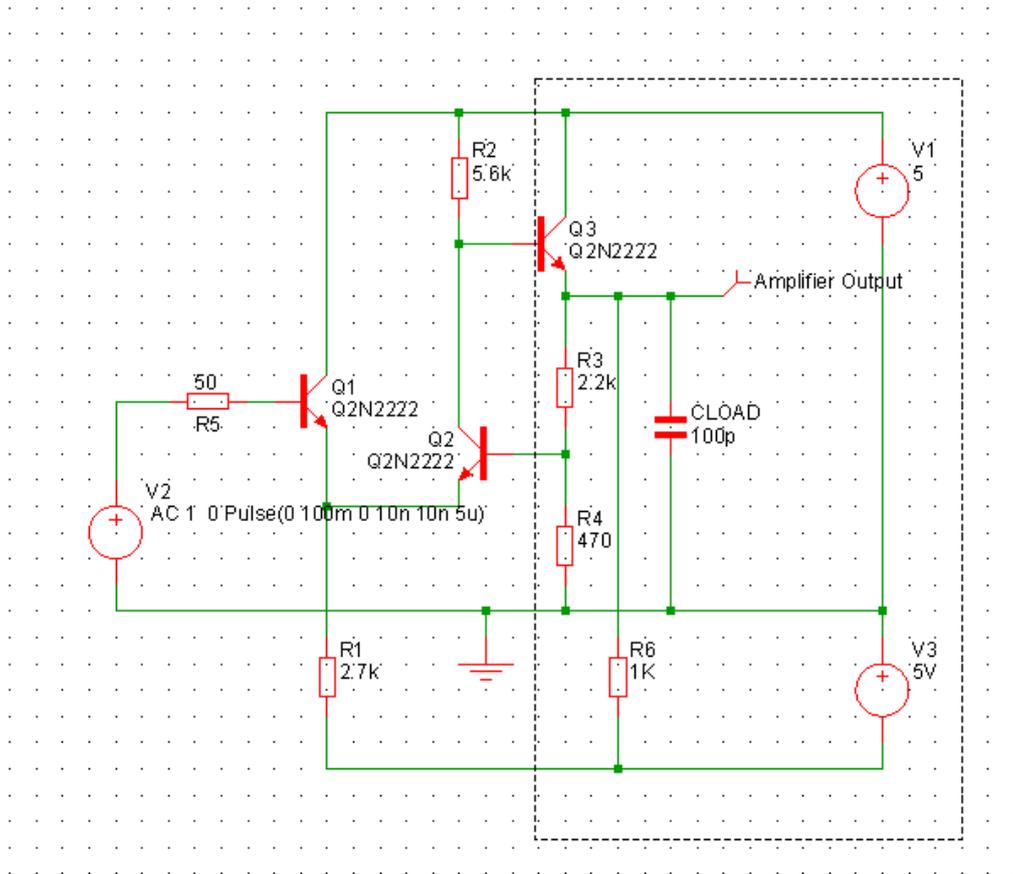
If you don't get it quite right, press the Undo Zoom button:



to return to the previous view. You can also use the left, right, up and down arrow keys to shift the position of the curves.

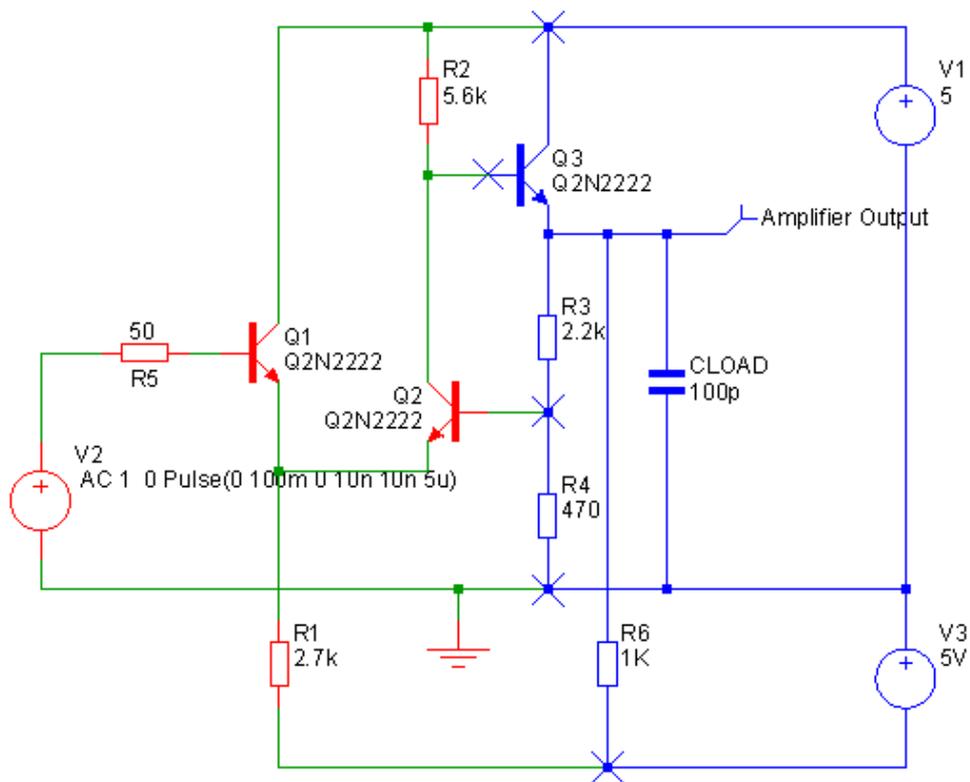
We can probably improve the ringing by adding a small phase lead in the feed back loop. This can be done by connecting a small capacitor between the emitter of Q3 and the base of Q2. There isn't room to add this tidily at present, so first, we will move a few parts to make some space. Proceed as follows:

1. In the schematic window, drag the mouse with the left key pressed over the region shown by the dotted lines below:



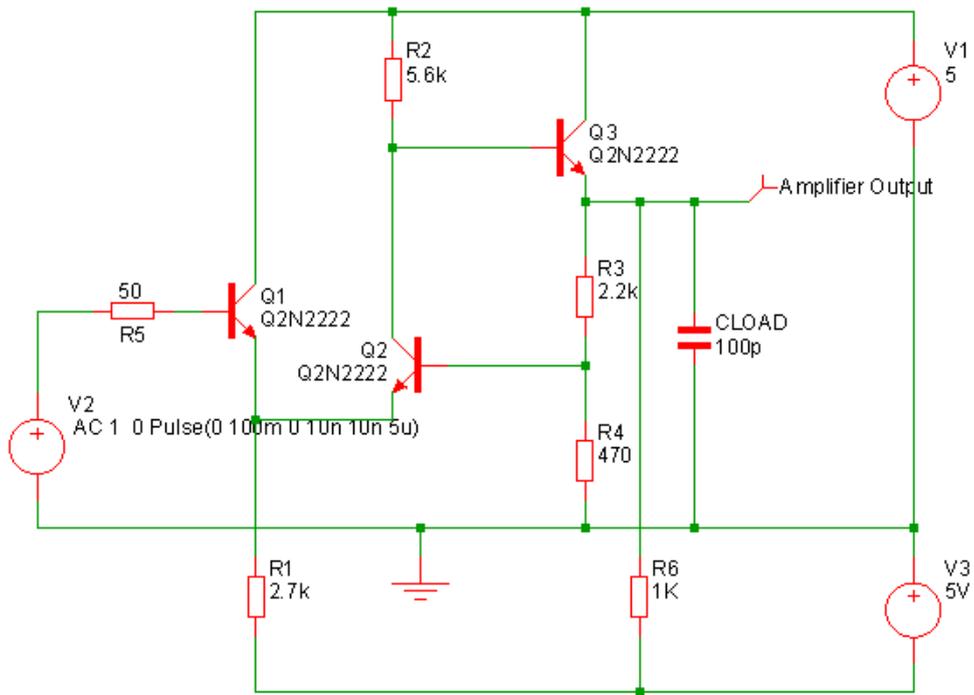
As you drag the mouse, a rectangle should appear.

2. Release the mouse. The area enclosed will turn blue:

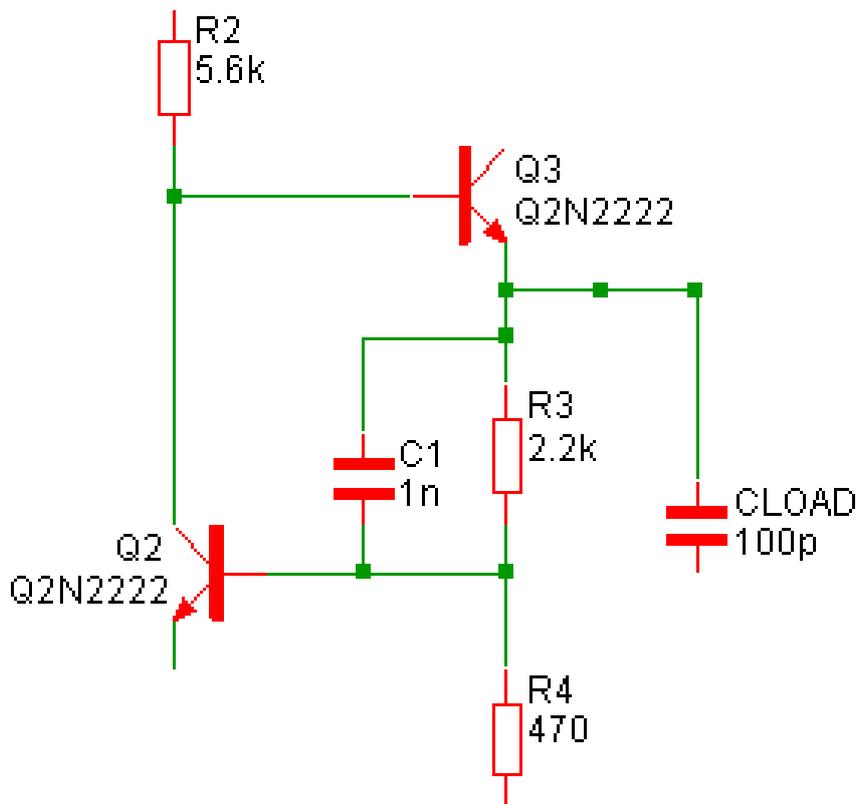


The blue wires and parts are said to be *selected*. To move them...

3. Place the cursor within one of the selected parts - V1 say - then press and hold the left mouse key.
4. Move the mouse to the right by two grid squares then release the left key.
5. Unselect by left clicking in an empty area of the schematic. This is what you should now have:

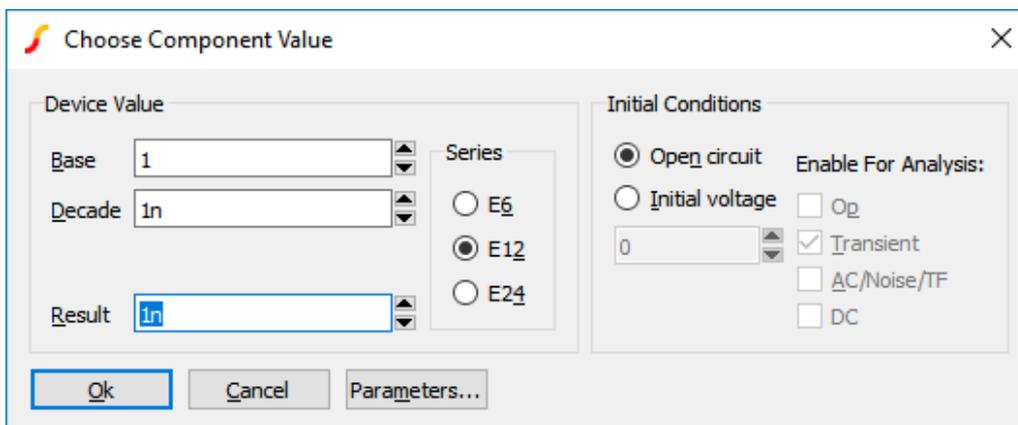


6. Wire in the capacitor C1 as shown below using a similar procedure as for the resistor R6.



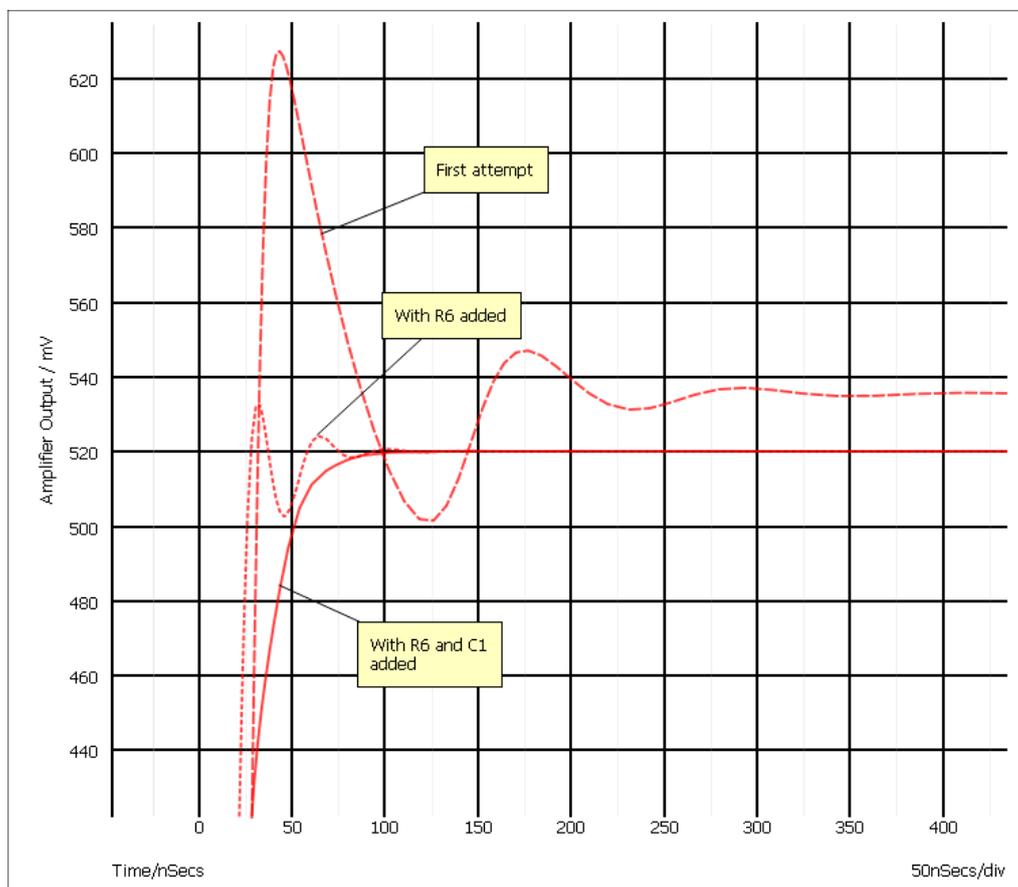
1nF is obviously far too high a value so we will try 2.2pF. To change the part's value proceed as follows:

1. Double click C1. You should see the following dialog box appear:



You can type the new value in directly in the Result box or you can select a value using the mouse alone with the up and down arrow buttons. Leave the Initial Conditions setting at its default (Open Circuit)

2. Now re-run the simulation. This is the result you should see:



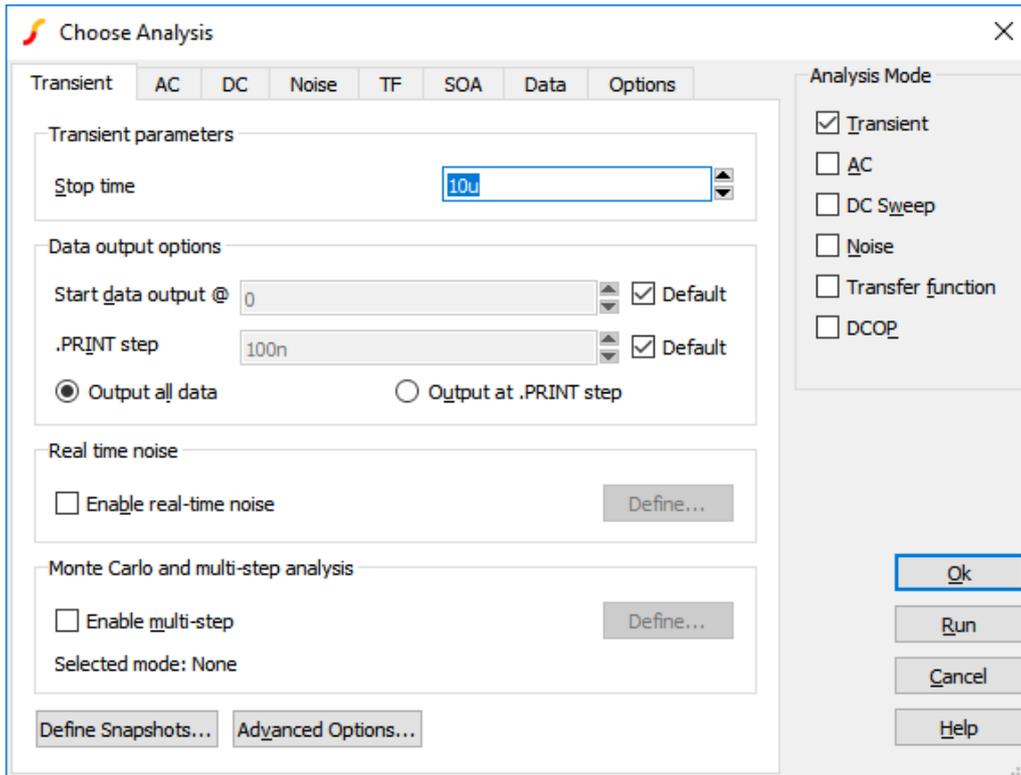
The solid curve is the latest result. This is now a big improvement on our first attempt.

You will notice that a new curve is displayed each time you run a new simulation. This is the default behaviour but this can be changed so that, for example, old curves are deleted leaving only the latest on view. To do this, double click the probe - that is the object labelled "Amplifier Output". Set History to 1 and close the box. (For more information see "[Probe Options Sheet](#)" on page 291). You can change the number of curves that are shown at any time using the popup menu **History | Show/Hide....**

We will now round off tutorial 1 by introducing AC analysis.

AC analysis performs a frequency sweep over a specified frequency range. To set one up, follow these instructions:

1. In the schematic window, select the menu **Simulator | Choose Analysis....** This is what you will see

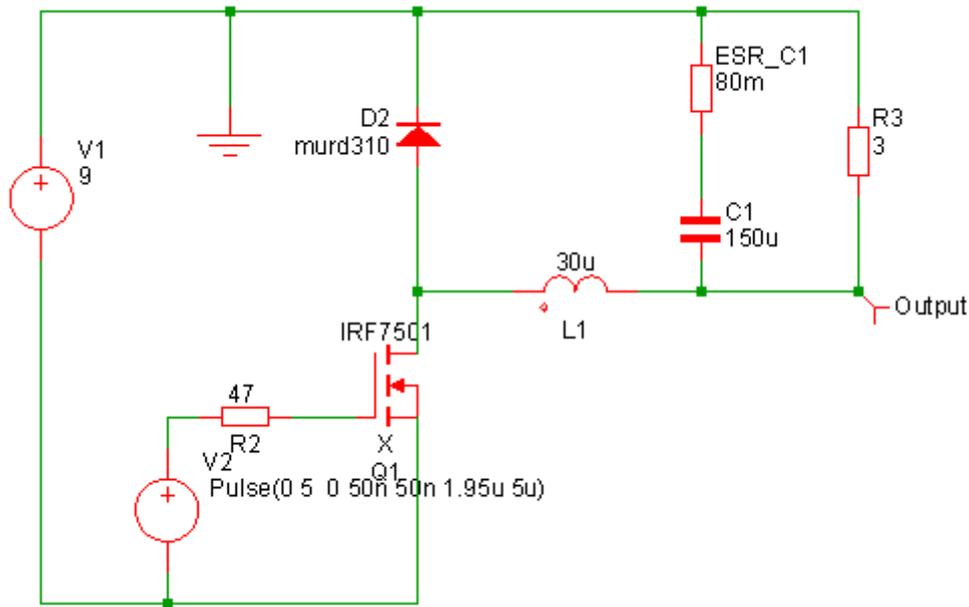


Click AC check box and uncheck the Transient check box. The details of the AC sweep have already been set up - click the AC tab at the top to see them.

2. Run the simulation - this will open a new graph sheet.

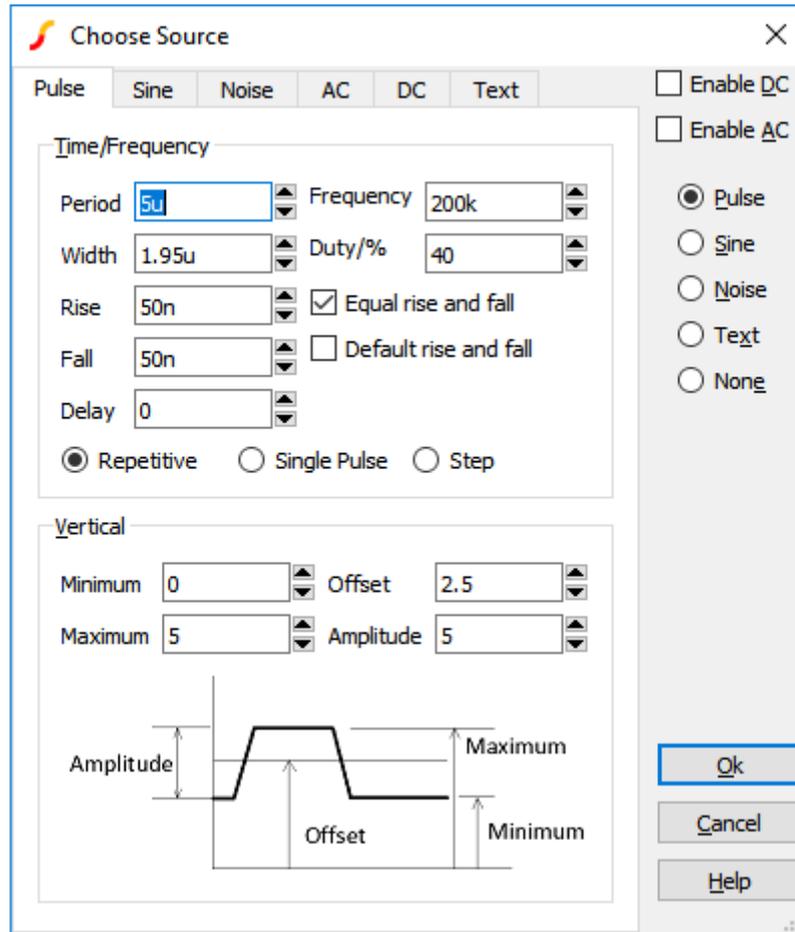
### 3.3 Tutorial 2 - A Simple SMPS Circuit

In this tutorial we will simulate a simple SMPS switching stage to demonstrate some of the more advanced plotting and waveform analysis facilities available with SIMetrix.



You can either load this circuit from EXAMPLES/SIMetrix/Tutorials/Tutorial2 (see [“Examples and Tutorials - Where are They?”](#) on page 7) or alternatively you can enter it from scratch. The latter approach is a useful exercise in using the schematic editor. To do this follow these instructions:

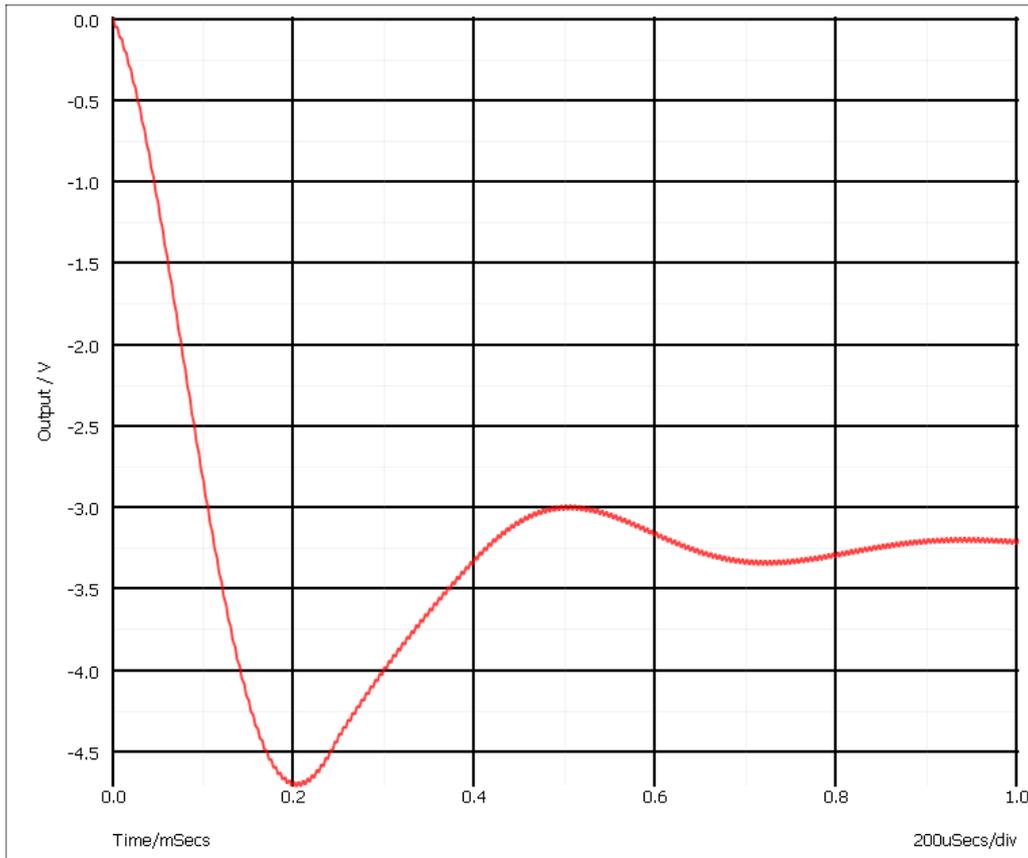
1. Place the parts and wires as shown above.
2. The probe labelled ‘Output’ can be selected from the following locations:  
 The Part selector. If this is not already showing, select menu **View | Show Part Selector**.  
 Navigate to Probes → Voltage Probe  
 OR Menu **Place | Probe | Voltage Probe**  
 OR Menu **Probe | Place Fixed Voltage Probe**  
 OR by pressing ‘B’ in the schematic editor.
3. After placing the output probe, double click to edit its label. Enter `Output` in the box titled `Curve Label`. All the other options may be left at their defaults.
4. For the pulse source V2, you can use **Place | Voltage Sources | Universal Source** or the **Universal Source** tool bar button.
5. Double click V2. Edit the settings as shown below:



This sets up a 200kHz 5V pulse source with 40% duty cycle and 50nS rise time.

- Set up the simulation by selecting the schematic menu **Simulator | Choose Analysis....** In the dialog box, check **Transient**. Usually we would set the **Stop** time but on this occasion, the default 1mS is actually what we want. Now select the **Advanced Options** button. In the **Integration method** box, select **Gear integration**. This improves the simulation results for this type of circuit. You will still get sensible results without checking this option, they will just be a little better with it. (For more information, see *Simulator Reference Manual/Convergence, Accuracy and Performance*.)

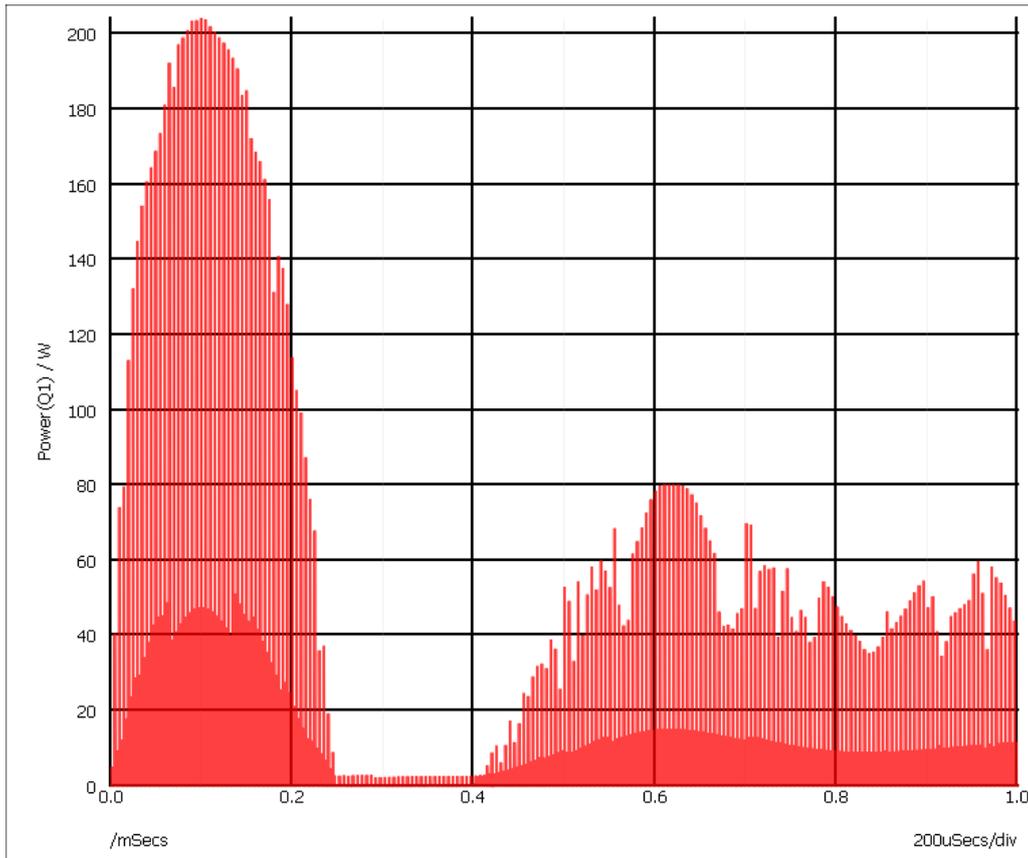
If you have any graph windows open, you should now close them. Once you have loaded or entered the circuit, press **F9** or use the schematic **Simulator | Run Schematic** menu to start the simulation. This will take somewhat longer than the previous tutorial but still less than 1 second on a modern machine. This is the graph you will see



The circuit is the switching stage of a simple step-down (buck) regulator designed to provide a 3.3 V supply from a 9V battery. The circuit has been stripped of everything except bare essentials in order to investigate power dissipation and current flow. Currently, it is a little over simplified as the inductor is ideal. More of this later. We will now make a few measurements. First, the power dissipation in Q1:

1. Create an empty graph sheet by pressing **F10**
2. Select schematic menu **Probe | Power In Device....** Left click on Q1.

This is what you should see:

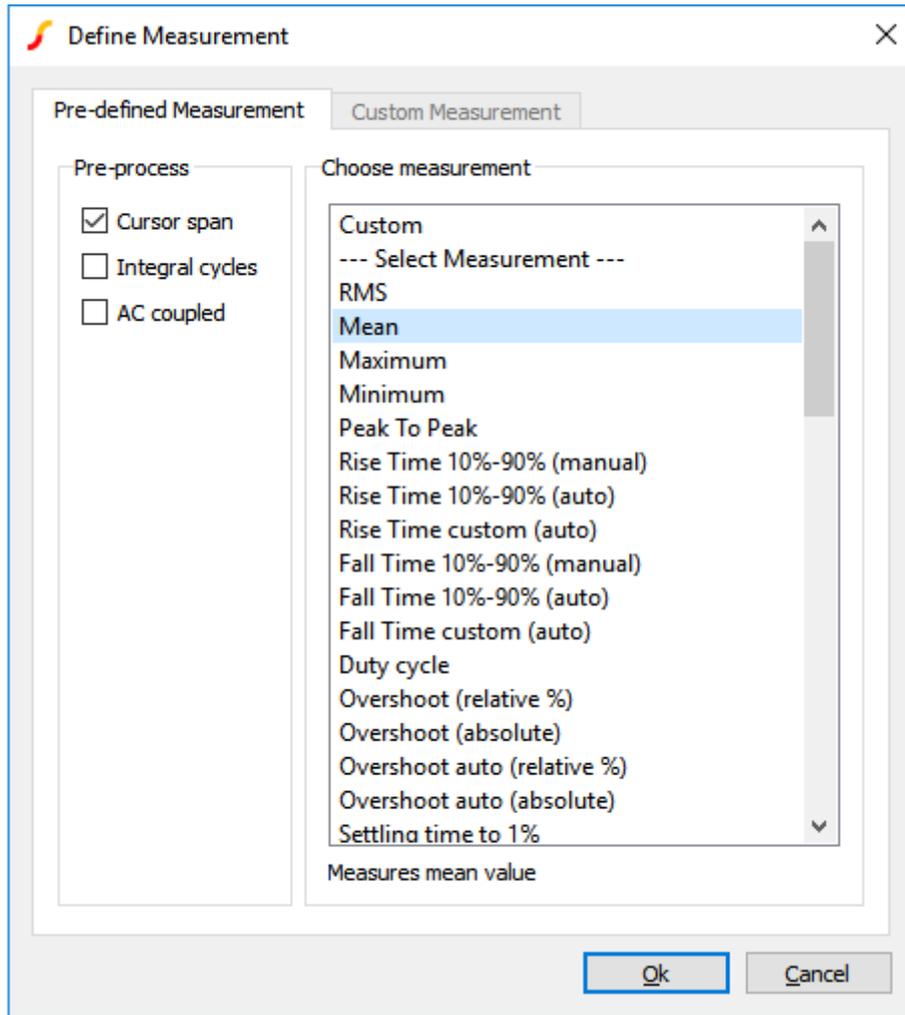


This shows a peak power dissipation of 200W although you are probably more interested in the average power dissipation over a specified time. To display the average power dissipation over the analysis period:

1. Select menu **Measure | Mean**

This should display a value of about 517mW. This will be shown in the measurement window below and to the right of the graph plot area. This is the average power over the whole analysis period of 1mS. You can also make this measurement over any period you select using the cursors as described below:

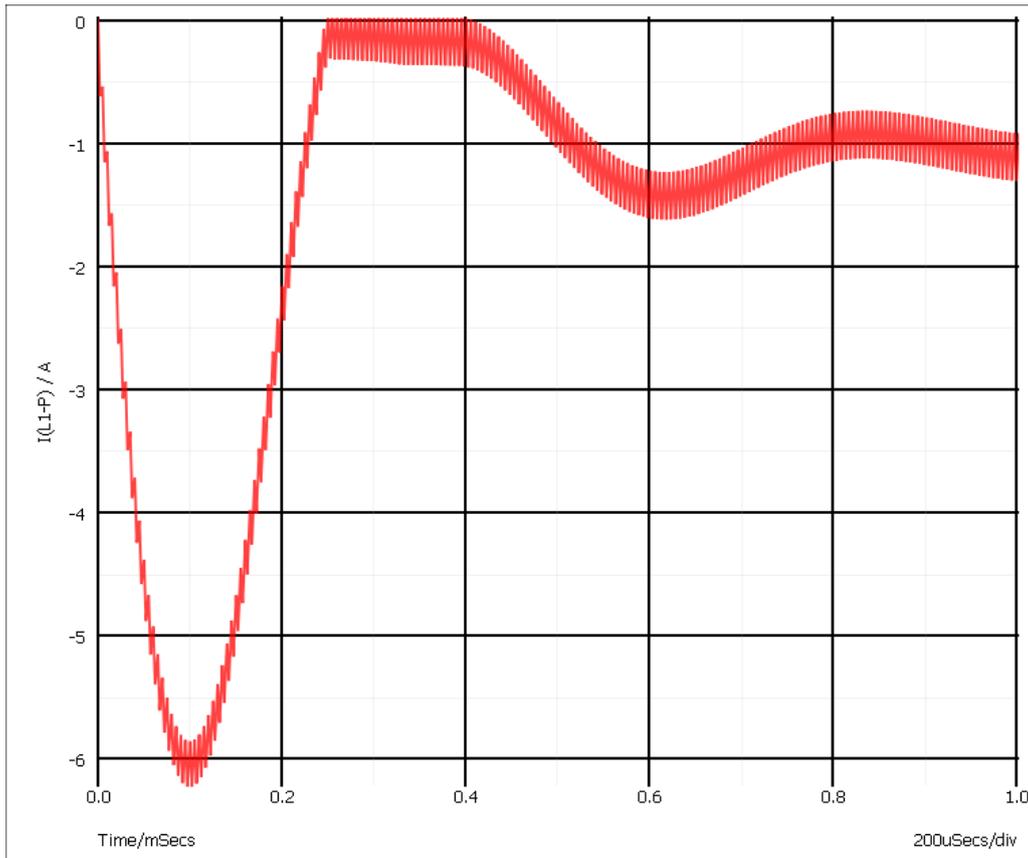
1. Zoom in the graph at a point around 100uS, i.e. where the power dissipation is at a peak.
2. Switch on graph cursors with menu **Cursors | Toggle On/Off**. There are two cursors represented by cross-hairs. One uses a long dash and is referred to as the reference cursor, the other a shorter dash and is referred to as the main or measurement cursor. When first switched on the reference cursor is positioned to the left of the graph and the main to the right.
3. Position the cursors to span a complete switching cycle. There are various ways of moving the cursors. To start with the simplest is to drag the vertical hairline left to right. As you bring the mouse cursor close to the vertical line you will notice the cursor shape change. See [“Graph Cursors” on page 347](#) for other ways of moving cursors.
4. Press **F3** or select analysis menu **Measure | More Functions...**:



In the right hand pane labelled Choose Measurement select Mean. In the Pre-process group, select Cursor span. Click Ok to close the dialog box.

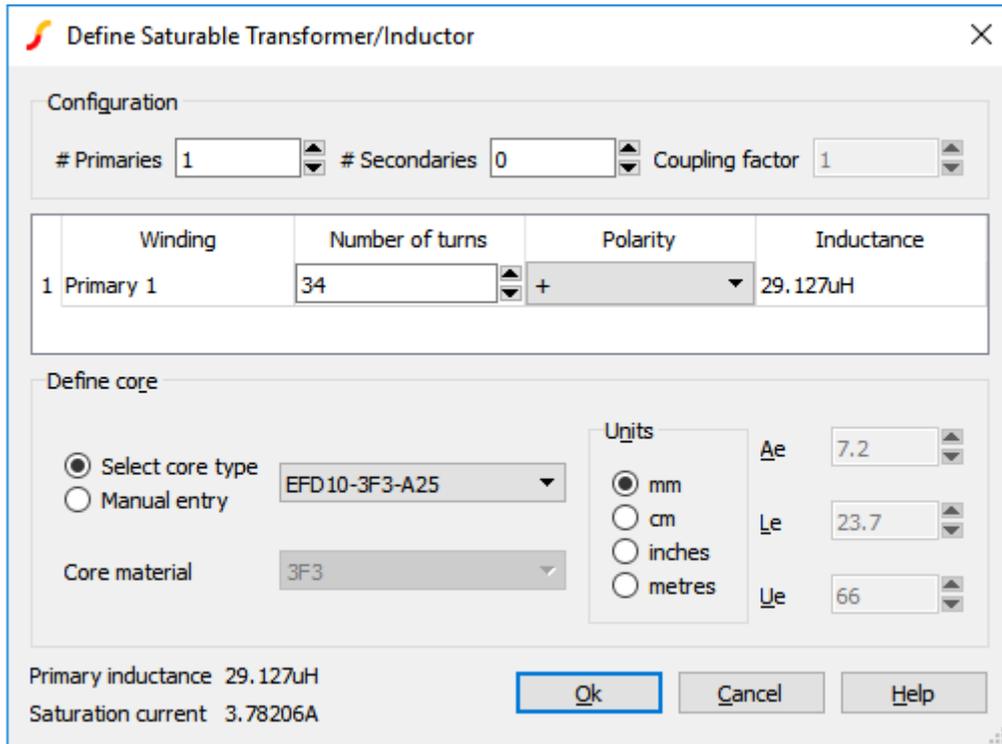
You should see a value of about 2.8W displayed. This is somewhat more than the 517mW average but is still well within the safe operating area of the device. However, as we noted earlier, the inductor is ideal and does not saturate. Lets have a look at the inductor current.

1. Select schematic menu **Probe | Current in Device Pin (New graph sheet)...**
2. Left click on the left pin of the inductor L1. This is what you will see:



This shows that the operating current is less than 1.5A but peaks at over 6A. In practice you would want to use an inductor with a maximum current of around 2A in this application; an inductor with a 6A rating would not be cost-effective. We will now replace the ideal part, with something closer to a real inductor.

1. Delete L1.
2. Select schematic menu **Place | Magnetics | Saturable Transformer/Inductor...** A dialog box will be displayed. (See picture below). Select 0 secondaries then enter 34 for the number of turns in primary 1. Next check **Select Core Type**. Select EFD10-3F3-A25. This is part number for a Ferroxcube ferrite core. This is what you should have:



Click Ok to close the dialog box.

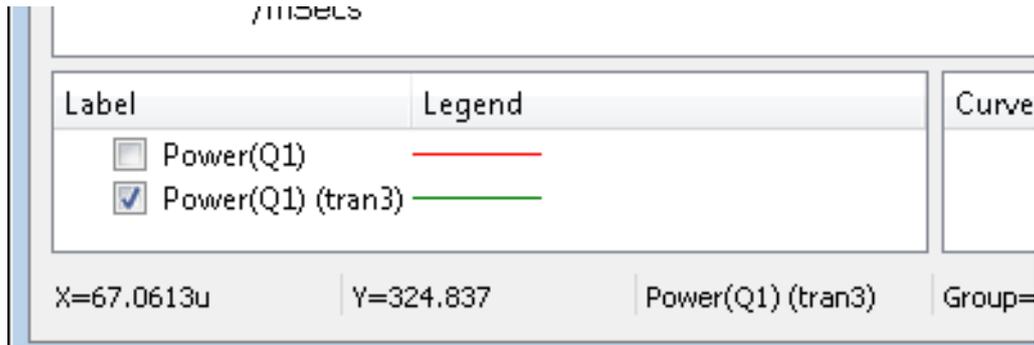
3. Place the inductor in the same place as before.
4. Run a new simulation.
5. Select the graph sheet that displayed the inductor current by clicking on its tab at the top of the graph window. Now select schematic menu **Probe | Current in Device Pin...** and left click on the left hand inductor pin.

You will notice the peak current is now in excess of 45A. This is of course because the inductor is saturating. You can also measure the peak power over 1 cycle:

1. Select the graph sheet with the power plot then select **Probe | Power In Device...**
2. Zoom back to see the full graph using the zoom button.



3. Zoom in on the peak power.
4. Position the cursors to span a full cycle. (The cursors are currently tracking the first power curve. This doesn't actually matter here as we are only interested in the x-axis values. If you want to make the cursors track the green curve, you can simply pick up the cursor at its intersection with the mouse and drag it to the other curve).
5. We now have two curves on the graph so we must select which one with which we wish to make the measurement. You can simply click on the curve to select it or check the box as shown below:



- As before, press **F3** then select **Mean** with the cursor span pre-process option. The new peak power cycle will now be in the 11-12W region - much more than before.

### 3.4 Tutorial 3 - Installing Third Party Models

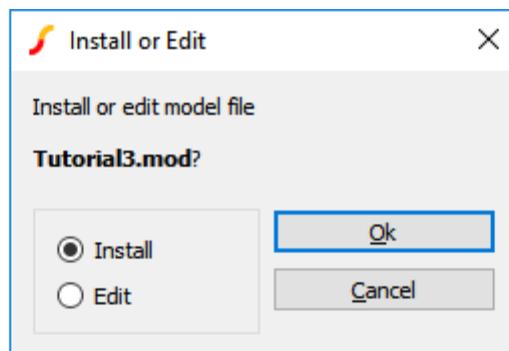
In this tutorial, we will install a device model library. For this exercise, we have supplied a model library file - TUTORIAL3.MOD - with just two devices. These are:

SXN1001 - an NPN bipolar transistor SXOA1000 - an opamp.

Both are totally fictitious.

You will find this file in the tutorial folder i.e. Examples/SIMetrix/Tutorials/Tutorial3.mod. There are two aspects to installing a model. SIMetrix needs to know where within your file system, the model is located. If the model is to be listed in the model library browser system, then SIMetrix also needs to know what symbol to use for it in the schematic and what category it comes under. This is how you do it:

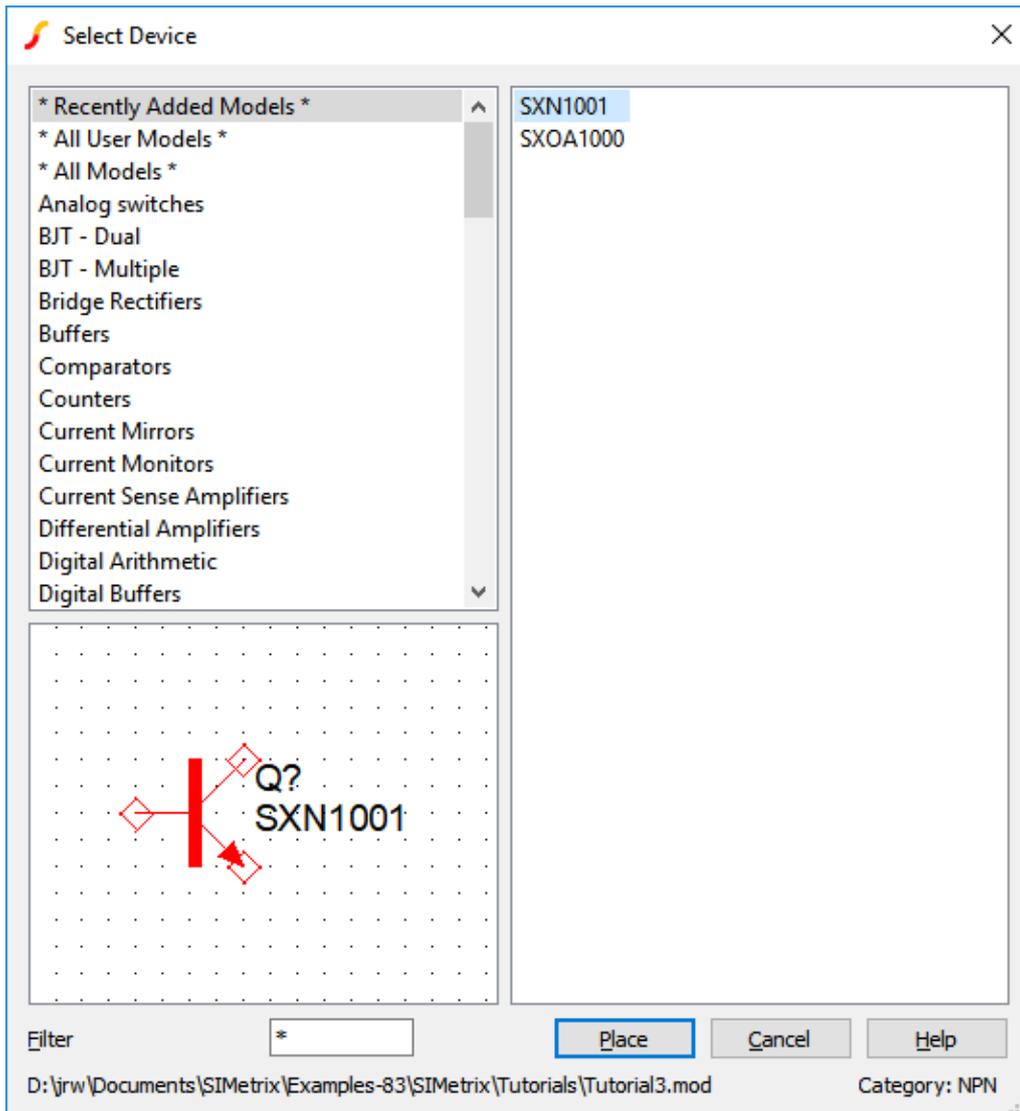
- Open windows explorer or click on My Computer or open other file manager of your choice.
- Locate TUTORIAL3.MOD in EXAMPLES/SIMetrix/Tutorials (see [“Examples and Tutorials - Where are They?”](#) on page 7). Pick the file up and drop into the SIMetrix command shell message window. That is, drop it in the window where SIMetrix messages are displayed. If the command shell is not visible, select menu **View | Show Command Shell**. After dropping the file, this dialog box will appear:



Select **Install** then **Ok**. The message "Model library changed. Rebuilding catalogs, please wait..." will be displayed, followed by "Complete".

At this stage, SIMetrix knows where to find our fictitious devices. You will find that it also knows about the NPN transistor as the following demonstrates:

1. Open an empty schematic.
2. Press **ctrl-G** or select menu **Place | From Model Library**. You should see a window displayed with the caption **Select Device**
3. Select the **\* Recently Added Models \*** category from the top of list shown on the left hand side.
4. Select **SXN1001** from the listed items on the right hand side. This is what you should see:



5. Press **Place** to place the device on the schematic.

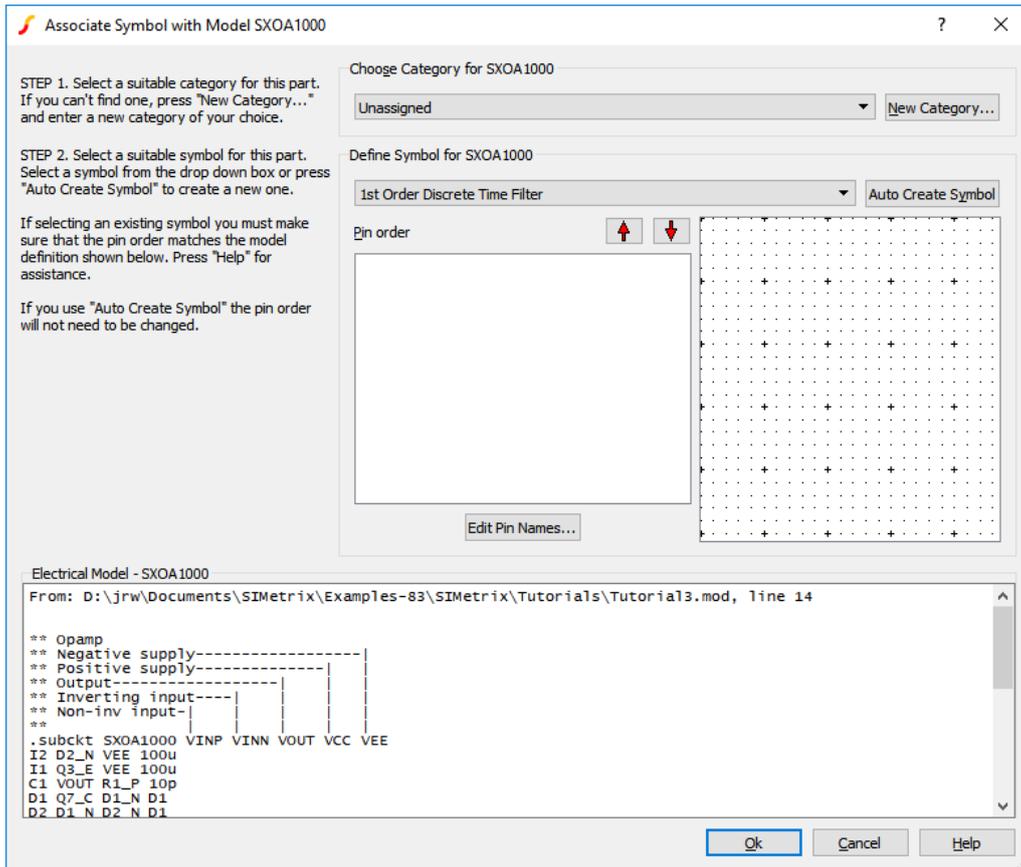
Without you having to tell it, SIMetrix already knew that the SXN1001 is an NPN transistor. This is because it is a primitive device defined using a **.MODEL** control. Such devices are built in to the simulator and SIMetrix can determine the part type simply by reading the **.MODEL** control in **TUTORIAL3.MOD**.

This is not the case with the other device in the model library. This is an opamp and is defined as a subcircuit. This is a module made up of other parts, in this case BJTs, diodes, resistors and current sources. SIMetrix can't tell what type of device this is. It knows that it has five terminals and it knows

where the electrical model is located in the file system, but it doesn't know what schematic symbol to use for this model.

SIMetrix will ask you for this information when you try and place it. Follow this procedure:

1. Repeat the steps 1-5 above but instead select the SXOA1000 device instead of the SXN1000. Notice that when you select the device in the right hand side, you see the message **SIMetrix does not know what symbol to use for this model**. Press "Place" to resolve.
2. After pressing the **Place** button, you should see the following box:



3. First specify a suitable category for the device. In this case it is an operational amplifier, so select Op-amps from the drop down box labelled **Choose Category for SXOA1000**.
4. Next define a symbol for this part. Under **Define Symbol for SXOA1000** select **Operational Amplifier - 5 terminal**.

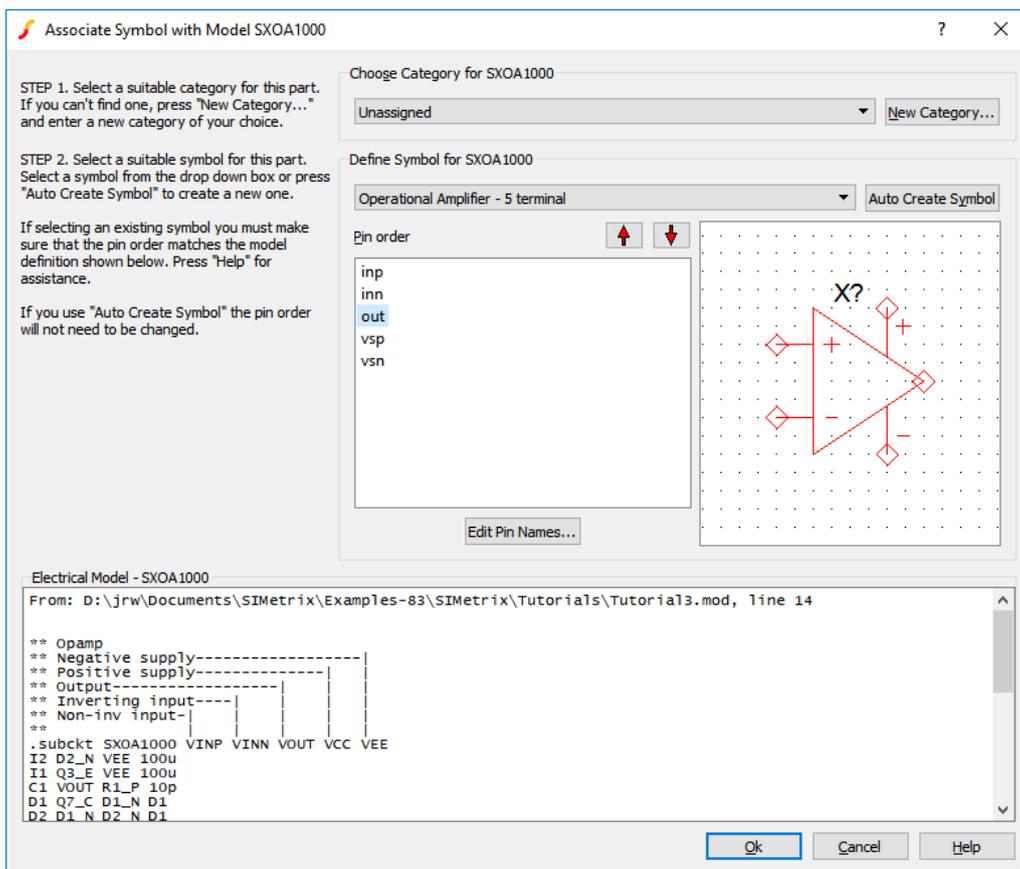
We have not quite finished yet. Our new op-amp has the wrong pin out for the schematic symbol. The pin order for the symbol is shown in the third box down on the right and is as follows:

Pin name	Function
INP	Non-inverting input
INN	Inverting input
VSP	Positive supply
VSN	Negative supply

Pin name	Function
OUT	Output

The text at the bottom of the dialog box shows the actual subcircuit definition. Fortunately, it has been annotated with the function of each of the sub-circuit's terminals. (This is in fact usually the case with third-party models). As you can see, the output terminal is in the wrong place. We can change the pin order using the Pin order up and down buttons:

1. Select out in the pin order box.
2. Click twice on the up button so that it is positioned between inn and vsp. This is what you will now have:



3. Press OK

You will now find our op-amp listed under the Op-amps category in the model library browser.

### 3.4.1 Notes

You will not always need to execute the above procedure to associate models and symbols even for subcircuit devices. SIMetrix is supplied with a data base of over 40000 devices that are already associated. These are devices for which SPICE models are known to be available from some third party source. This database is in the file all.cat which you will find in 'support\devdb' directory. The information you enter in

the associate models dialog is stored in a file called user\_v2.cat which you will find in devdb/user under the application data directory - see [“Application Data Directory” on page 444](#).

You will also not need to perform the above procedure for many 2 and 3 terminal semiconductor parts even if they are not listed in the all.cat database. SIMetrix runs a series of simulations on these models and attempts to determine what the device type is from their results. If successful, the ‘association’ step demonstrated above will be skipped.

Finally, there is a method of embedding association information within the model itself, and such models will not require manual association. The embedding method is described in [“Embedded Association” on page 217](#).

# Chapter 4

## Getting Started

This chapter describes the basic operation of SIMetrix and is aimed primarily at novice users.

The basic steps to simulate a circuit are as follows:

1. Enter the circuit using the schematic editor. See [“Using the Schematic Editor” on page 32](#)
2. Add signal sources if relevant to your circuit. See [“Circuit Stimulus” on page 129](#).
3. Specify analysis. This includes what type of analysis and over what limits it should run. See [“Analysis Modes” on page 34](#).
4. Run simulator. See [“Running the Simulator” on page 43](#)
5. Graph results. See [“Plotting Simulation Results” on page 43](#)

The following paragraphs briefly describe these steps. More details are given in other sections.

### 4.1 Simulation Modes - SIMetrix or SIMPLIS

If you have SIMetrix/SIMPLIS, you can set the schematic editor to one of two modes to select whether you are using the SIMetrix native (SPICE) simulator or the SIMPLIS simulator. To create a new schematic in SIMPLIS mode use menu **File | New | SIMPLIS Schematic** and to create a new schematic in SIMetrix mode, use **File | New | SIMetrix Schematic**. To switch to from SIMPLIS mode to SIMetrix mode, use menu **Simulator | Switch to SIMetrix mode** and to switch from SIMetrix mode to SIMPLIS mode use menu **Simulator | Switch to SIMPLIS mode**.

If the schematic is not empty and you change modes, the program will check that all parts entered on the sheet are compatible with the newly selected simulation mode as not all parts will work in both modes. Any that are believed not to be compatible will be highlighted and a warning will be issued. To clear the highlighting, select **Edit | Unhighlight (All Open Sheets)**. You will most likely need to replace those parts but in some cases you may simply need to re-enter the same part.

If you wish to enter a circuit that will work in both modes, you should enter it in SIMPLIS mode and not use any of the parts in the menu **Place | SIMPLIS Primitives** or **Place | Digital** except **Place | Digital | SIMetrix Compatible Logic Gates**. Following this advice will not guarantee a circuit with dual mode simulation ability but will minimise the chance of placing a device that is compatible with only one of the simulators.

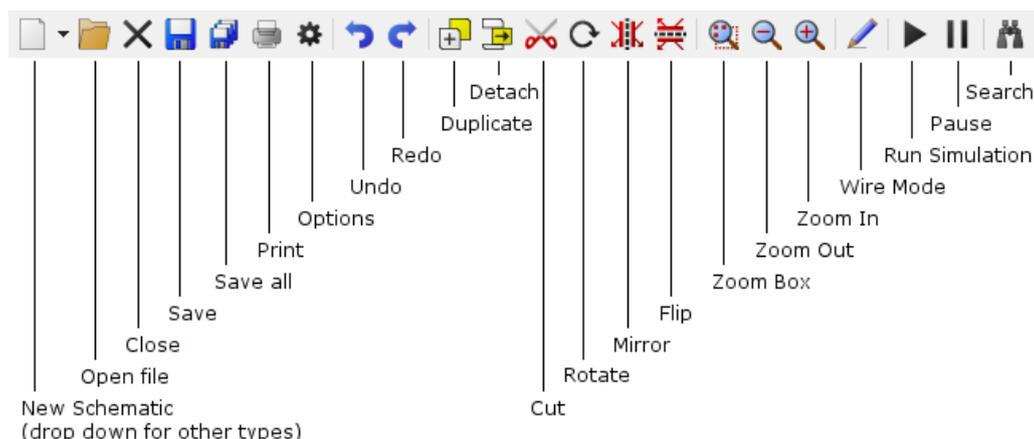
If you use SIMPLIS predominantly, you may wish to select SIMPLIS as the default simulator for all new schematics. To do this, select menu **File | Options | General...** then select SIMPLIS for the Initial Simulator setting.

## 4.2 Using the Schematic Editor

### 4.2.1 Creating a Schematic

The schematic editor has been designed to be intuitive to use and you may not need to read too much about it. Here we describe the less obvious procedures. If you have SIMetrix/SIMPLIS, make sure you are in the correct mode before entering a schematic. See above section.

In the following notes references are made to the schematic tool bar. The diagram above shows the standard toolbar and the function of each button.



### To Place a Part

If it is a simple device which only needs a value such as a resistor or capacitor, select the appropriate symbol from the tool bar or **Place** menu. For other devices that require a part number, it is easiest to use the model library browser. Select menu **Place | From Model Library** and select your desired device.

### To Change Value or Device Type for a Part

First select it then double click or select schematic popup **Edit Part...** or press **F7**. A dialog box appropriate for the type of part will be displayed. For devices requiring a model name, a list of available types will appear.

### To Rotate, Mirror or Flip a Part

Use the **Rotate** toolbar button (see diagram above) or key **F5** to rotate a part.

This operation can be performed while a part is being placed or while a block is being moved or copied (see below).

You can also select a part or block then press the rotate button/key to rotate in-situ.

To mirror a part or block through the y-axis, press the **Mirror** toolbar button or **F6** key.

To flip a part or block (mirror about x-axis), press **Flip** button or press **shift-F6**.

## Wiring

There are a number of ways of placing a wire:

Method 1: Place the mouse cursor close to an unselected symbol pin or wire end. Notice the cursor shape change to depict a pen symbol. Now left click to mark the start point then left click again to mark the final point. SIMetrix will automatically route the wire for you. You can also mark intermediate points if you would prefer to define the precise route rather than accept the auto-routed connection.

Method 2: If you have a three button mouse or scroll wheel mouse you can start a wire by clicking the middle button/scroll wheel. Clicking the middle button or scroll wheel a second time will complete the wire and start a new one. Click the right button or press escape to terminate wiring.

Method 3: Start a wire by pressing **F3** or double clicking the left button. Single clicking the left button will complete the wire and start a new one. Click the right button or press escape to terminate wiring.

Method 4: Press the **Wiring** tool button on the toolbar. You can start a wire by single clicking the left button, otherwise continue as described above. Press the **Wire** button again to cancel this mode.

## Disconnecting Wires

In most cases just selecting the wire then deleting it with the delete key or button is the easiest way.

In some cases, especially if the wire is short, it is difficult to select it. In this instance, hold down the **shift** key then select area enclosing the wire. Press delete button.

## To Move Items Disconnected

Select items then schematic menu **Edit | Detach** or the **Detach** toolbar button. Move items to desired location then press left mouse key. You can rotate/flip/mirror the items (see above) while doing so.

## To Copy Across Schematics

Select block you wish to copy. Choose menu **Edit | Copy**. In second schematic choose **Edit | Paste**.

## Multiple Selection

Individual items which do not lie within a single rectangle can be selected by holding down the **control** key while using the mouse to select the desired items in the usual way.

## Selecting Wires Only

Hold down **shift** key while performing select operation.

## Holding Down the ALT Key...

... while selecting will limit part selection to only devices that are wholly enclosed by the selection box.

## Zoom Area

Press the **Zoom** box button on schematic. Drag mouse to zoom in on selected area.

## Zoom Full (Fit to Area)

Press the **HOME** key to fit the whole schematic in the current window size.

## 4.3 Circuit Rules

The following design rules must be observed for the simulation to run correctly. Note that most circuits obey them anyway and they do not impose serious limitations on the capability of the simulator.

- There must always be at least one ground symbol on every circuit.
- Every node on the circuit must have a dc path to ground. For example, two capacitors in series form a node which does *not* have DC path to ground. If you do have a floating node, you can provide a DC path to ground by connecting a high value resistor (e.g. 1G) between it and ground. Capacitors without initial conditions do not have a DC path. But if you set an initial condition on a capacitor a DC path is formed and this method is an alternative to using a resistor to provide a DC path.

Also note that inductors with an initial condition do not have a DC path. This is because they are treated as a constant current during the calculation of the DC bias point.

If using a high value resistor to make a DC path for a transformer winding, we recommend that you also place a low value capacitor in parallel with it. This is not always necessary but can help avoid problems with transient analysis convergence. It is best to choose a realistic value that would represent what the capacitance would really be in the real-world circuit.

- There must not be any zero resistance loops constructed from voltage sources, inductors without initial conditions or capacitors with initial conditions. If you do have such loops you can insert a low value resistor. You should choose a value that represents what the resistance would be in the real world, e.g. the actual winding resistance of an inductor, and never use an unrealistically small value. Very small resistances (e.g. 1 fempto-Ohm) can cause convergence problems.

For loops containing inductors you can break the loop by adding an initial condition to the inductor instead of adding a resistor.

Failure to observe the above usually leads to a *Singular Matrix* error.

## 4.4 Analysis Modes

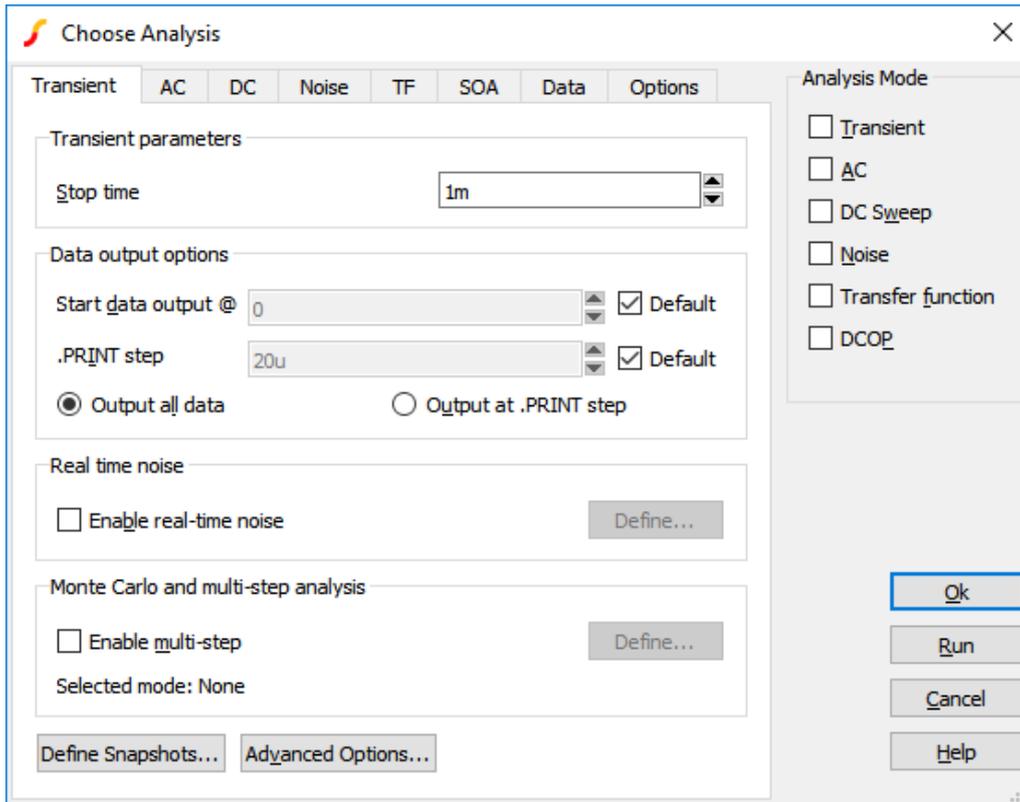
### 4.4.1 Overview

In this section we explain how to setup the most commonly used analysis modes in both SIMetrix and SIMPLIS (SIMetrix/SIMPLIS product only)

For more comprehensive details on analysis modes, see [“Analysis Modes” on page 222](#) for SIMetrix and [“SIMPLIS Analysis Modes” on page 270](#) for SIMPLIS .

### 4.4.2 Setting up a SIMetrix Simulaton

Analysis mode is setup by selecting the menu **Simulator | Choose Analysis....** In SIMetrix mode this displays the following dialog box:



To set up the analysis, first check the box on the right according to which analysis you wish to perform. You can select more than one, but usually it is easier to do just one at a time.

The following describes the most commonly used modes and how to set one up.

## Transient

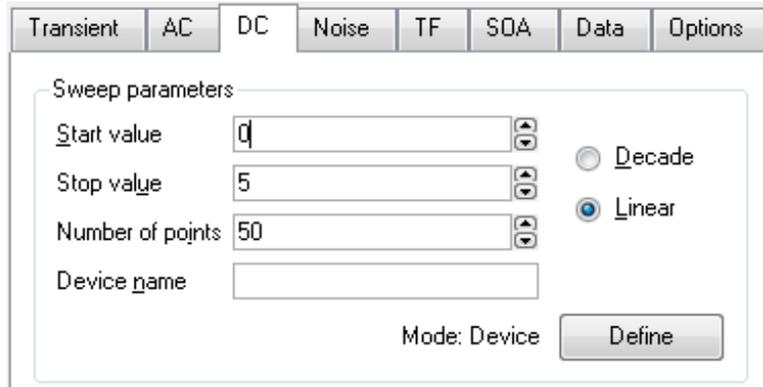
The most useful and general mode. First the bias point is found. Then the circuit is simulated over a fixed time interval in steps of varying size according to circuit activity. The circuit may contain any number of time varying voltage and current sources (stimuli see [“Circuit Stimulus” on page 129](#)) to simulate external signals, test generators etc.

Usually you only need to specify the Stop time specified at the top of the dialog box. For information on the remaining options see [“Transient Analysis” on page 225](#).

## DC Device Sweep

A DC device sweep will sweep a specified device over a defined range and compute the DC operating point of the circuit at each point. This allows, for example, the DC transfer function of the circuit to be plotted. Note that all reactive elements are ignored in DC sweep.

To set up a DC Sweep, select the DC Sweep check box at the right and the DC tab at the top. You will need to enter some values in the Sweep Parameters section:



The analysis will sweep the device you specify in the Device name box over the range specified by Start value, Stop value and Number of points or Points per decade if you select a decade sweep.

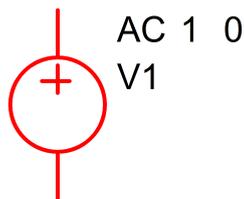
The entry in the Device name box is the part reference of the device to be swept and for DC sweep would usually be a voltage source, a current source or a resistor.

Device sweep is just 1 of 5 modes available with DC sweep. The Define... button allows you to specify one of the others. See “DC Sweep” on page 236 for details.

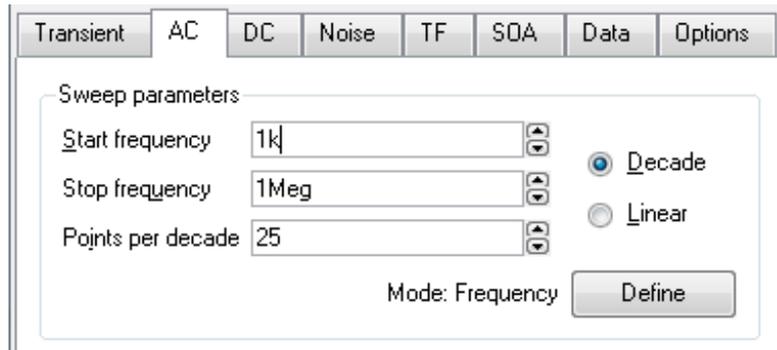
## AC Frequency Sweep

An AC Frequency Sweep calculates the small signal response of a circuit to any number of user defined inputs over a specified frequency range. The small signal response is computed by treating the circuit as linear about its DC operating point.

There must be at least one input source for AC analysis for the results to be meaningful. Connect a voltage or current source to the circuit, select it then press **F7**. In the dialog box select the Enable AC check box. On the circuit, an AC input voltage source will look something like this:



To set up an AC Frequency Sweep, select the AC check box at the right and the AC tab at the top. You will need to enter some values in the Sweep Parameters section:



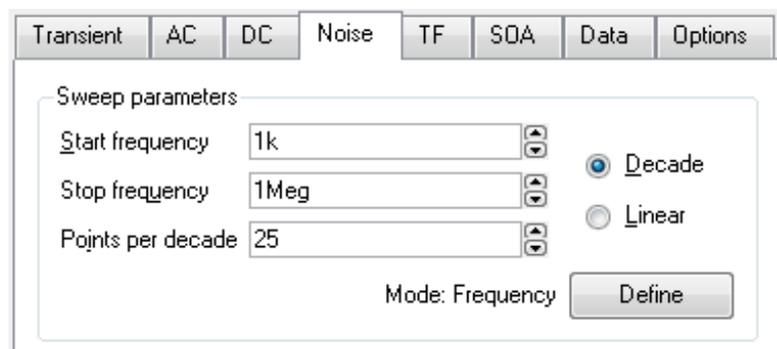
The analysis will sweep the frequency over the range specified by Start frequency, Stop frequency and Number of points or Points per decade if you select a decade sweep.

Frequency sweep is just 1 of 6 modes available with AC sweep. The Define... button allows you to specify one of the others. See [“AC Sweep” on page 238](#) for details.

## Noise Frequency Sweep

Like AC analysis, Noise analysis is a small signal mode. Over a user defined frequency range, the circuit is treated as linear about its DC operating point and the contribution of all noisy devices to a designated output is computed. The total noise at that output is also calculated and optionally the noise referred back to an input source may also be computed.

To set up a Noise Frequency Sweep, select the Noise check box at the right and the Noise tab at the top. You will need to enter some values in the Sweep Parameters section:



The analysis will sweep the frequency over the range specified by Start frequency, Stop frequency and Number of points or Points per decade if you select a decade sweep.

You will also need to enter some additional parameters:

Noise parameters	
Output node	<input type="text"/>
Ref node (optional)	<input type="text"/>
Source name (optional)	<input type="text"/>

Use "Terminal" symbol to assign names for output and optional ref nodes

An entry in the **Output node** box is compulsory. It is the name of the circuit node as it appears in the netlist. Usually the schematic's netlist generator chooses the node names but we recommend that when running a noise analysis that you assign a user defined name to your designated output node. You can do this using a terminal symbol (**Place | Connectors | Terminal**) To find out more see ["Finding and Specifying Net Names"](#) on page 64.

An entry in the **Ref node** box is optional. It is the node to which the output node is referenced. If omitted it is assumed to be ground.

An entry in the **Source name** box is optional. If specified the noise referred back to it will be calculated. Enter the part reference of the voltage or current source that is used as the input to your circuit.

Frequency sweep is just 1 of 6 modes available with Noise Analysis. The **Define...** button allows you to specify one of the others. See ["Noise Analysis"](#) on page 241 for details.

## DC Operating Point

To specify a DC operating point analysis, check the **DCOP** box on the right of the Choose Analysis Dialog.

Note that the DC operating point is calculated automatically for all the other analysis modes described above although for noise analysis the results are not stored.

After a DC operating point has been completed, you can annotate your schematic with markers to display the voltages at each node. Press **ctrl-M** on the schematic to place a single marker or select the popup menu **Place | Bias Annotation | Auto Place Markers** to automatically place markers on all nodes. See ["Viewing DC Operating Point Results"](#) on page 373 for full details.

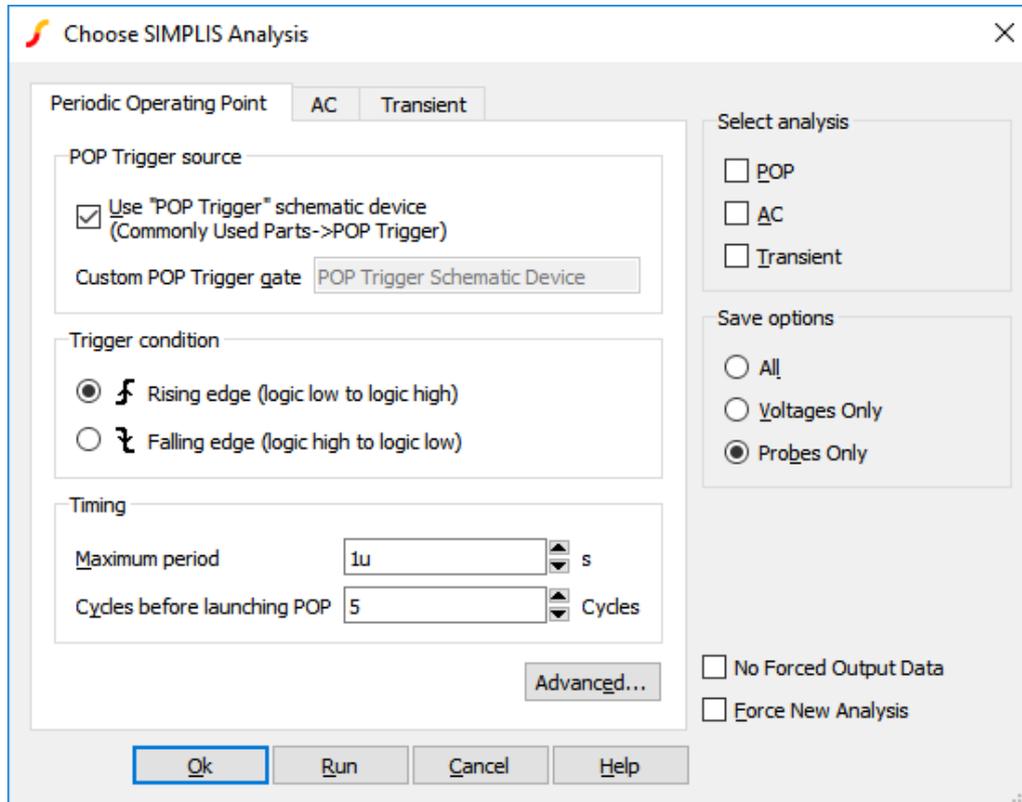
## Other Analysis Modes

Real-time noise	An extension of transient analysis which enables noise generators for noisy devices using the same equations used for small signal noise analysis. See <a href="#">"Real Time Noise"</a> on page 246.
Transfer function	Similar to AC but instead of calculating the response to a (usually) single input, it calculates the response from all signal sources to a single output. See <a href="#">"Transfer Function"</a> on page 247.
Sensitivity	Calculates the sensitivity of a specified output to device and model parameters. See <a href="#">"Sensitivity"</a> on page 250.
Multi-step Analyses	Transient, AC, DC, Noise and Transfer Function analyses can be run in an auto-repeat mode while stepping a user-defined parameter. See <a href="#">"Multi-step Analyses"</a> on page 259.
Monte Carlo Analysis	See <a href="#">"Monte Carlo Analysis"</a> on page 411.

### 4.4.3 Setting up a SIMPLIS Simulation

SIMPLIS analyses are setup using the same menu as SIMetrix but you must first set the schematic to SIMPLIS mode. See “Simulation Modes - SIMetrix or SIMPLIS” on page 31 for details.

Select menu **Simulator | Choose Analysis...** You will see the following dialog box:

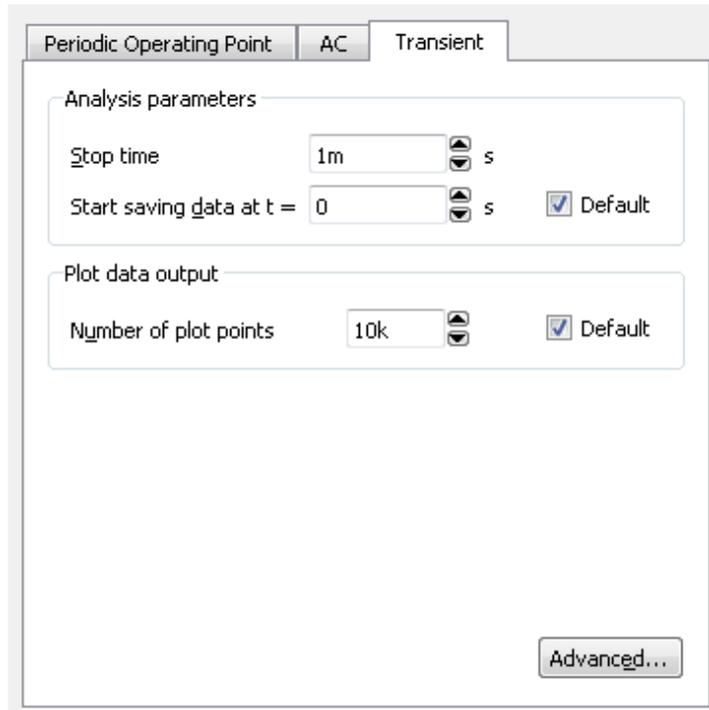


SIMPLIS has three analysis modes, namely Transient, Periodic Operating Point (POP) and AC. Transient is similar to SIMetrix transient analysis. POP is a unique analysis mode that finds the steady state of a switching circuit. AC finds the small signal response of a periodic system.

### Transient Analysis

To setup a basic transient analysis:

1. Click on the Transient tab:



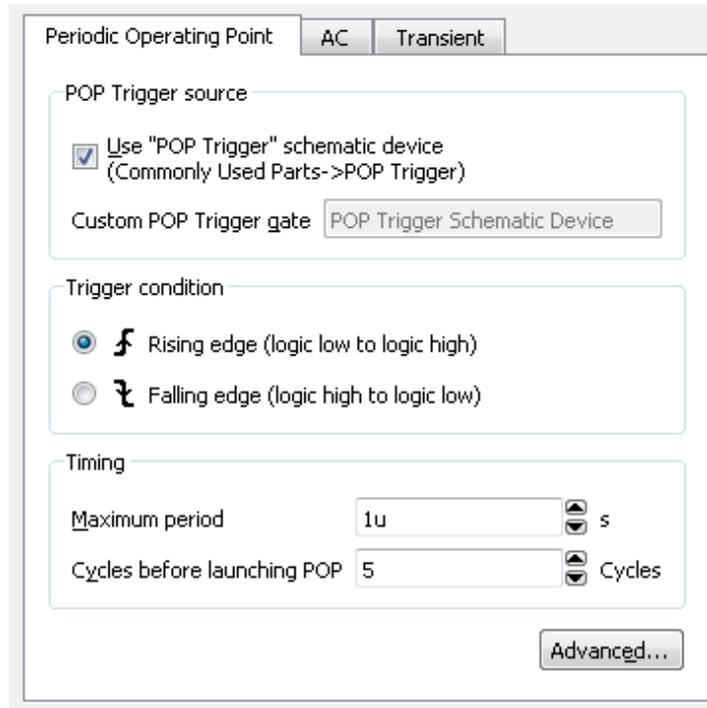
2. Check the Transient box in the Select analysis section.
3. Enter an appropriate stop time under Analysis parameters.
4. Enter an appropriate selection under Save Options. Its usually best to select All. This will instruct SIMPLIS to save all data for subsequent plotting.

In most cases the above is all you need to do. For information on the remaining transient analysis settings, see [“Transient Analysis” on page 225](#).

## Periodic Operating Point Analysis (POP)

To setup a POP analysis:

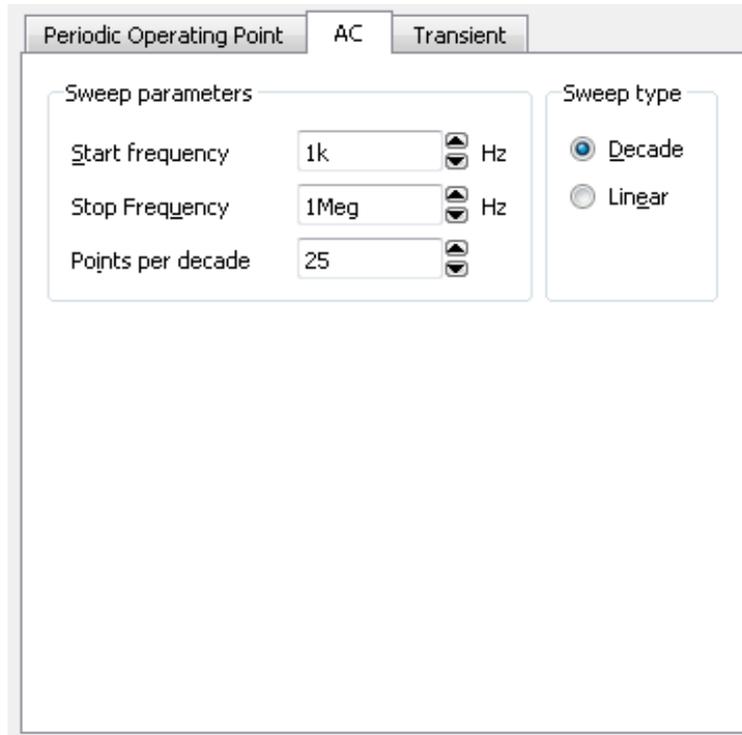
1. Select Periodic Operating Point sheet:



2. Check the POP box under Select analysis.
3. Check the Use "POP Trigger" schematic device box. You will need to place a POP trigger device on your schematic. See below.
4. In the Maximum period box, enter a value that is larger than the largest possible value of your circuits switching period.

You must place on your schematic a POP trigger device. Select menu **Place | Analog Functions | POP Trigger**. After placing the device, connect its input to a switching frequency signal. You do not need to connect the output of this device. Select the trigger device then press **F7**. Enter suitable values for Ref. Voltage and Hysteresis so that it will always reliably trigger on the switching waveform. If you don't use the output, there is no need to change the other parameters.

## AC Analysis



To setup an AC analysis:

1. Select AC sheet.
2. Check the AC box under Select analysis.
3. Enter parameters in Sweep Parameters section. These have the same meaning as the equivalent SIMetrix analysis.

### 4.4.4 Manual Entry of Simulator Commands

The analysis mode selected using **Simulator | Choose Analysis...** is stored in text form in the schematic's simulator command window also known as the **F11** window. If you wish, it is possible to edit this directly. Users familiar with the simulator's syntax may prefer this approach. Note that the text entered in the simulator command window and the Choose Analysis dialog box settings remain synchronised so you can freely switch between the two methods.

To open the simulator command window, select the schematic then press the **F11** key. It has a toggle action, pressing it again will hide it. If you have already selected an analysis mode using the Choose Analysis dialog box, you will see the simulator controls already present.

The window has a popup menu selected with the right key. The last item - **Edit file at cursor** - will open a text editor window with the file name pointed to by the cursor or selected text item if there is one.

The simulator command window can be resized using the splitter bar between it and the schematic drawing area.

If you have SIMetrix/SIMPLIS you should use the `.SIMULATOR` control to mark SIMetrix and SIMPLIS entries. If `.SIMULATOR SIMetrix` is encountered, all following lines will only work in SIMetrix mode

and will be ignored by SIMPLIS. Conversely, any lines following .SIMULATOR SIMPLIS will only be accepted by SIMPLIS and will be ignored by SIMetrix. All lines before any occurrence of .SIMULATOR or after .SIMULATOR DEFAULT will be accepted by both simulators.

## 4.5 Running the Simulator

### 4.5.1 SIMetrix

You can run the simulator using a number of methods:

- Select the **Simulator | Run Schematic** menu item
- Press **F9**
- Click the Run button in the **Simulator | Choose Analysis...** dialog box
- Click on the play button



A dialog box will show the status of the simulation.

You can pause the simulation by selecting the **Pause** button on the simulator status dialog box. To restart select the **Resume** button (the **Pause** button changes name when simulation pauses) or the **Simulator | Resume** menu item. There is no obligation to resume a simulation that has been paused. If you start a new run after having paused the previous one, you will be asked whether you wish to abandon the pending simulation run.

### Notes

1. There is no need to specify in advance of the simulation what voltages, currents and/or powers you wish to look at. By default everything except signals internal to some device models are stored in a disk file. You can decide after the run is complete what you wish to look at.
2. It is recommended that any schematics are saved before a run is commenced especially if the run is expected to take a long time.

### 4.5.2 SIMPLIS

If the schematic is in SIMPLIS mode, the procedure described above will start the SIMPLIS simulator. A window showing the progress of the SIMPLIS simulation will be displayed.

SIMPLIS can be aborted by pressing the **Abort** button in the progress window. SIMPLIS, cannot however, be paused and resumed.

## 4.6 Plotting Simulation Results

### 4.6.1 Overview

SIMetrix provides two methods of creating plots of simulated results.

The first approach is to fix voltage or current probes to the schematic before or during a run. SIMetrix will then generate graphs of the selected voltages and/or currents automatically. The probes have a wide range

of options which allow you to specify - for example - how the graphs are organised and when and how often they are updated.

The second approach is to randomly probe the circuit after the run is complete. (You can also do this during a run by pausing first). With this approach, the graph will be created as you point the probe but will not be updated on a new run.

You do not need to make any decisions on how you wish to probe your circuit before starting the run. You can enter a circuit without any fixed probes, run it, then randomly probe afterwards. Alternatively, you can place - say - a single fixed probe on an obvious point of interest, then randomly probe to investigate the detailed behaviour of your circuit.

Fixed schematic probes are limited to single ended voltages and currents and differential voltages. The random probing method allows you to plot anything you like including device power, FFTs, arbitrary expressions of simulation results and X-Y plots such as Nyquist diagrams. It is possible to set up fixed probes to plot arbitrary expressions of signals but this requires manually entering the underlying simulator command, the .GRAPH control. There is no direct schematic support for this. For details of the .GRAPH control see the “Command Reference” chapter of the *Simulator Reference Manual*.

### 4.6.2 Fixed Probes

There are several types of fixed probe. Three of the commonly used probes are:

1. Voltage. Plots the voltage on a net.
2. Current. Plots the current in a *device pin*.
3. Differential voltage. Plots the voltage difference between two points.

They are simply schematic symbols with special properties. When you place a fixed probe on the schematic, the voltage or current at the point where you place the probe will be plotted each time you run the simulation. The probes have a wide range of options which can be set by double clicking it. These options are covered in detail in section “Fixed Probes” on page 288.

There are more fixed probes available in addition to those described above. See “Fixed Probes” on page 288 for details.

### Fixed Voltage Probes

You can place these on a schematic with the single hot key ‘B’ or with one of the menus:

**Probe | Place Fixed Voltage Probe...**

**Place | Probe | Voltage Probe**

schematic popup **Probe Voltage**

### Hint

If you place the probe immediately on an existing schematic wire, SIMetrix will try and deduce a meaningful name related to what it is connected to. If you place the probe at an empty location, its name will be a default (e.g. PROBE1-NODE) which won’t be meaningful and you will probably wish to subsequently edit it.

### Fixed Current Probes

You can place these on a schematic with the single hot key ‘U’ or with one of the menus **Probe | Place Fixed Current Probe... Place | Probe | Current Probe** schematic popup **Probe Current...**

Current probes must be placed directly over a part pin. They will have no function if they are not and a warning message will be displayed.

## Fixed Differential Voltage Probes

These can be placed using one of the menus

**Probe | Place Fixed Diff. Voltage Probe...**

**Place | Probe | Differential Voltage Probe**

### 4.6.3 Random Probes

Most of the entries in the schematic's **Probe** menu are for random probing. You can probe, voltage, current, differential voltage, device power, dB, phase, Nyquist diagrams and much more. You can also plot arbitrary expressions of any circuit signal and plot signals from earlier simulation runs. Just a few of the possibilities to get you started are explained below. For a full reference see "[Random Probes](#)" on page 322.

### Random Voltage Probing

1. Select the schematic menu item **Probe | Voltage...**
2. Using the mouse, place the cursor over the point on the circuit you wish to plot.
3. Press the left mouse button. A graph of the voltage at that point will be created. The new curve will be added to any existing graph if the X-axis has the same units. Otherwise, a new graph sheet will be created.

### Random Voltage Probing - On New Graph Sheet

1. Select the schematic menu item **Probe | Voltage (New Graph Sheet)...**
2. Using the mouse, place the cursor over the point on the circuit you wish to plot.
3. Press the left mouse button. A graph of the voltage at that point will be created. A new graph sheet will be created for it unconditionally.

### Random Current Probing

1. Select the schematic menu item **Probe | Current...**
2. Using the mouse, place the cursor at the *device pin* whose current you wish to plot.
3. Press the left mouse button. A graph of the current at that point will be created. The new curve will be added to any existing graph if the X-axis has the same units. Otherwise, a new graph sheet will be created.

### Random Current Probing - On New Graph Sheet

1. Select the schematic menu item **Probe | Current in Device Pin (New Graph Sheet)...**
2. Using the mouse, place the cursor at the *device pin* whose current you wish to plot.
3. Press the left mouse button. A graph of the current at that point will be created. A new graph sheet will be created for it unconditionally.

## Probing dB and Phase for AC Analysis

In AC analysis you will probably want to plot signals in dB and you may also want to plot the phase of a signal.

1. Select the schematic menu item **Probe AC/Noise | db - Voltage...** for dB or **Probe AC/Noise | Phase - Voltage....**
2. Using the mouse, place the cursor over the point on the circuit you wish to plot.
3. Press the left mouse button. The new curve will be added to any existing graph if the X-axis has the same units. Otherwise, a new graph sheet will be created.

## Probing dB and Phase for AC Analysis - On New Graph Sheet

1. Create an empty graph sheet by pressing **F10** or selecting menu **Probe | New Graph Sheet**
2. Proceed as in above section.

## Differential Voltage Probing

The schematic menu **Probe | Voltage Differential...** allows you to plot the voltage difference between two points. When you select this menu click on the schematic twice. The first is the *signal node* and the second the *reference node*.

## Advanced Probing

The menu **Probe | More Probe Functions...** provides many more probing functions selectable from a tree structured list. More advanced plotting can be achieved with the menu **Probe | Add Curve...**. This opens a dialog box allowing you to enter any expression and which also provides a range of options on how you wish the graph to be plotted.

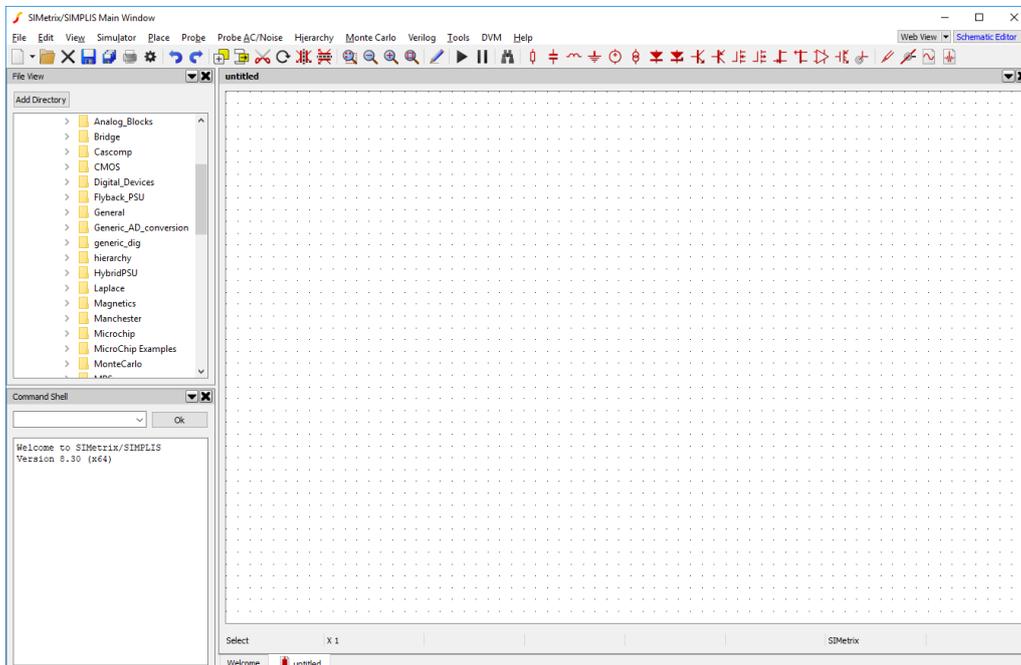
# Chapter 5

## SIMetrix GUI Environment

### 5.1 Overview

The SIMetrix user interface provides an environment where multiple views can be managed within a window. There are a range of views, for example content related views such as the Schematic Editor, Symbol Editor and Waveform Viewer, along with more system related views such as the Command Shell and File Viewer. By dragging a views title bar, views can be rearranged within a window into the layout that best suits your needs.

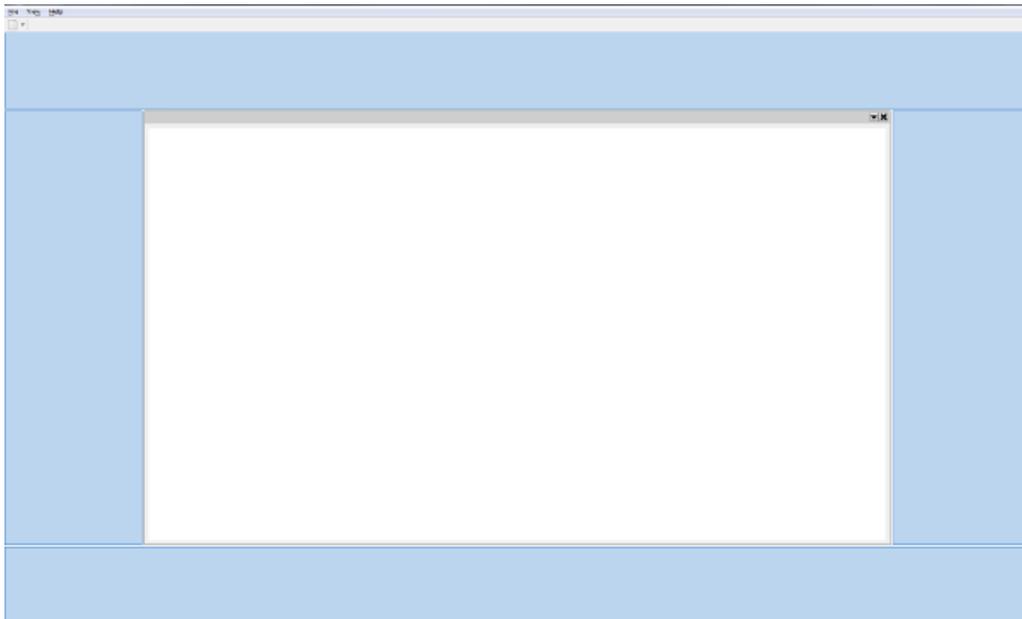
Before we discuss the layout possibilities, we note that there are conceptually two types of views within SIMetrix. The first, *System Views*, provide tools to operate the program, which includes the Command Shell and File Viewer. The second, *Workspace Views*, provide means for developing and reviewing designs, which includes the Schematic Editor and Waveform Viewer.



Overview of the User Interface.

In terms of layout, *System Views* sit on the outside of any *Workspace Views*. A visualisation of the regions that can be used to place a *System View* are shown below. *Workspace Views* will fill in any available space

if *System Views* are not present.

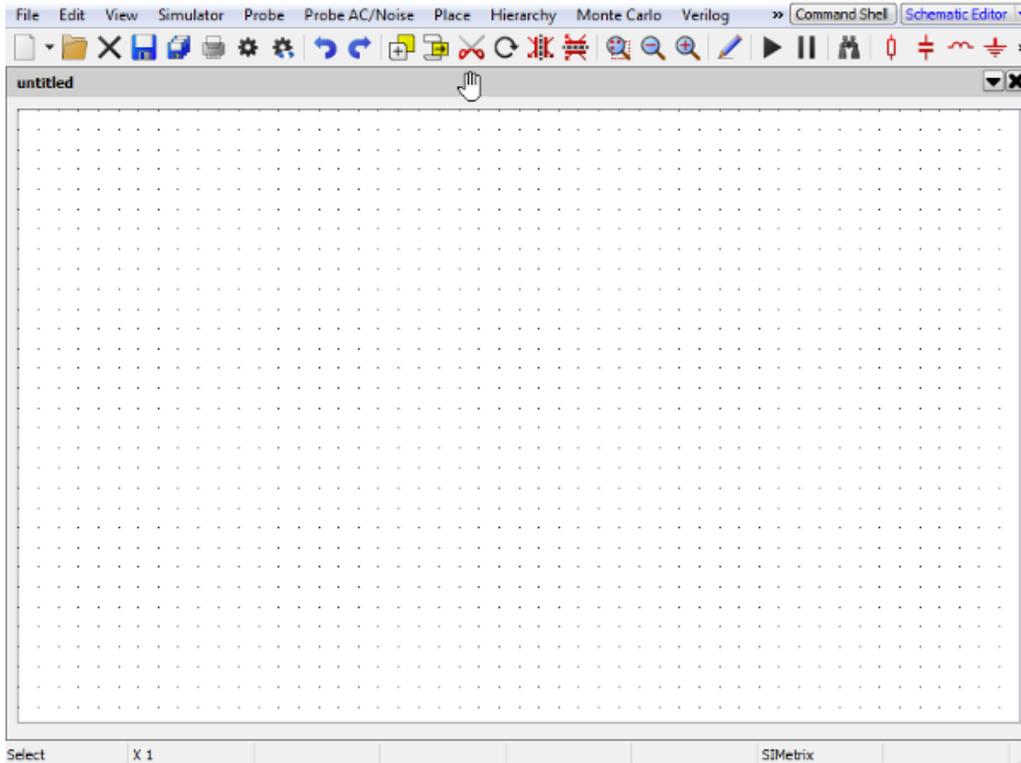


The regions that a *System View* can be placed (shown in blue), in relation to a *Workspace View* (shown in white).

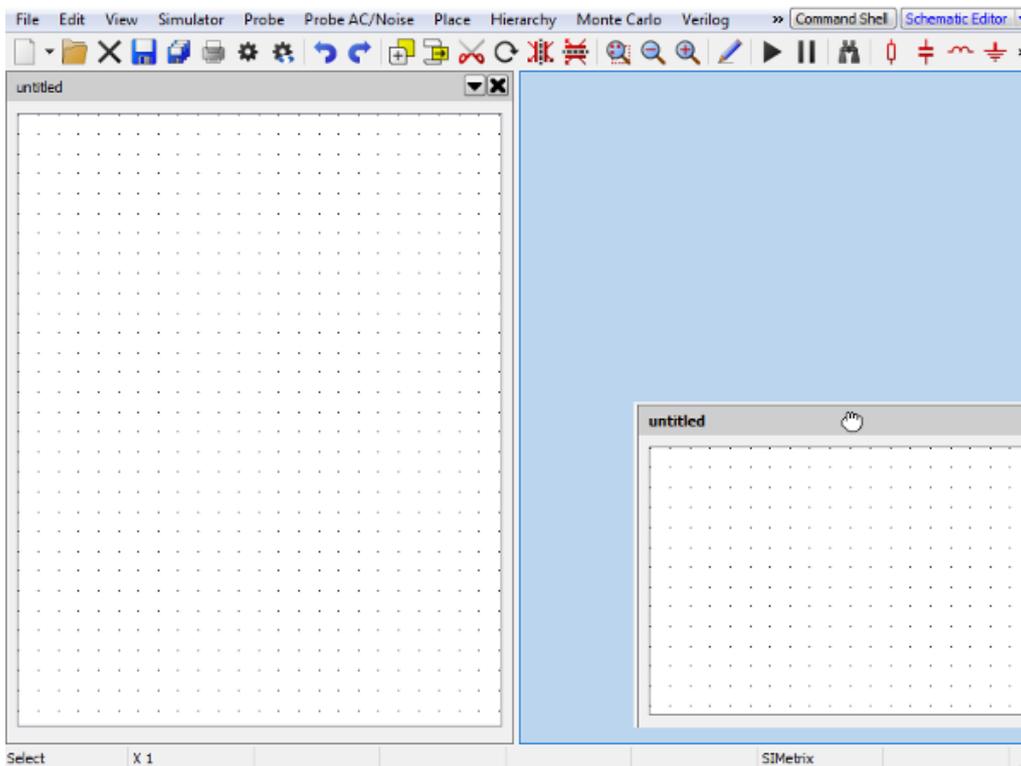
When a view is dragged, these guiding regions shown in blue will be displayed to highlight where a view can be dragged to. Alternatively, views can be moved to new or existing windows. In the [Interaction](#) section, more detail is given about arranging layouts within a window.

## 5.2 Interaction

To move any view, move the mouse to the title bar at the top of the view, shown in grey in the figure of a blank schematic below. The cursor will change to an open hand. Click on the title bar and hold, the cursor will change to a closed hand. Next move the mouse, causing the View to undock from the window and begin to be dragged with the mouse. As you move the View, possible places to redock the View will be highlighted in blue, which will also provide a preview of how the new window layout will look, as shown in the second figure below. Dragging the view on top of another view of the same type (either System or Workspace) will cause those Views to stack on top of each other.

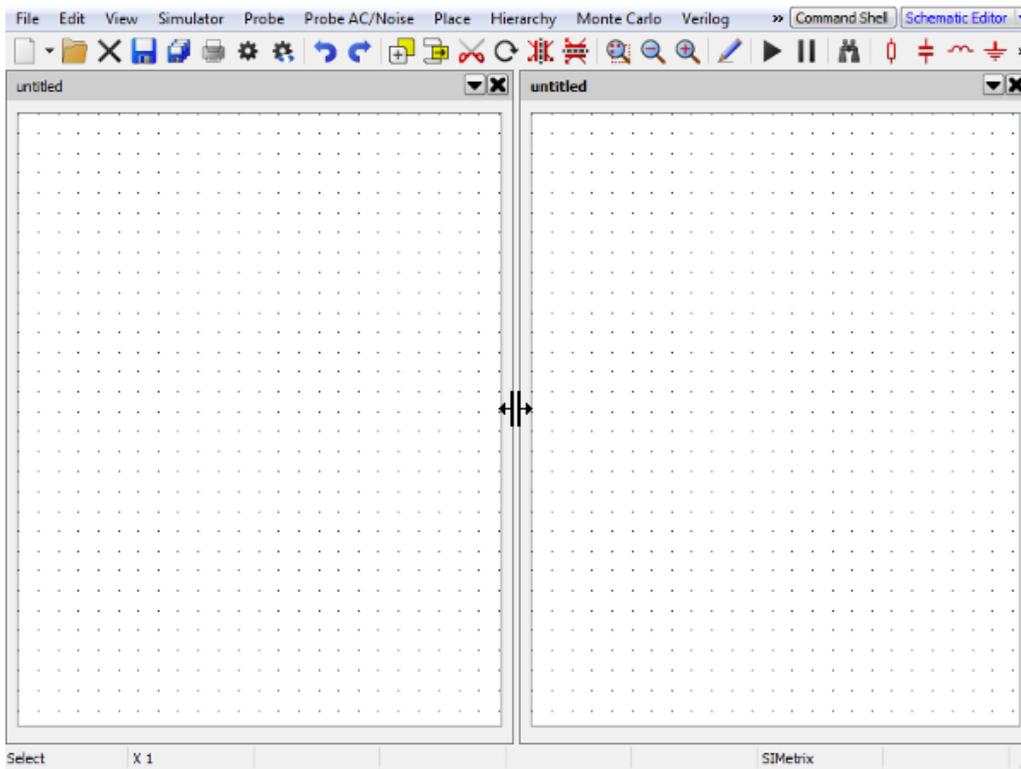


A blank schematic. The draggable title bar is shown shaded grey at the top of the schematic labelled 'untitled', along with the drag available cursor.



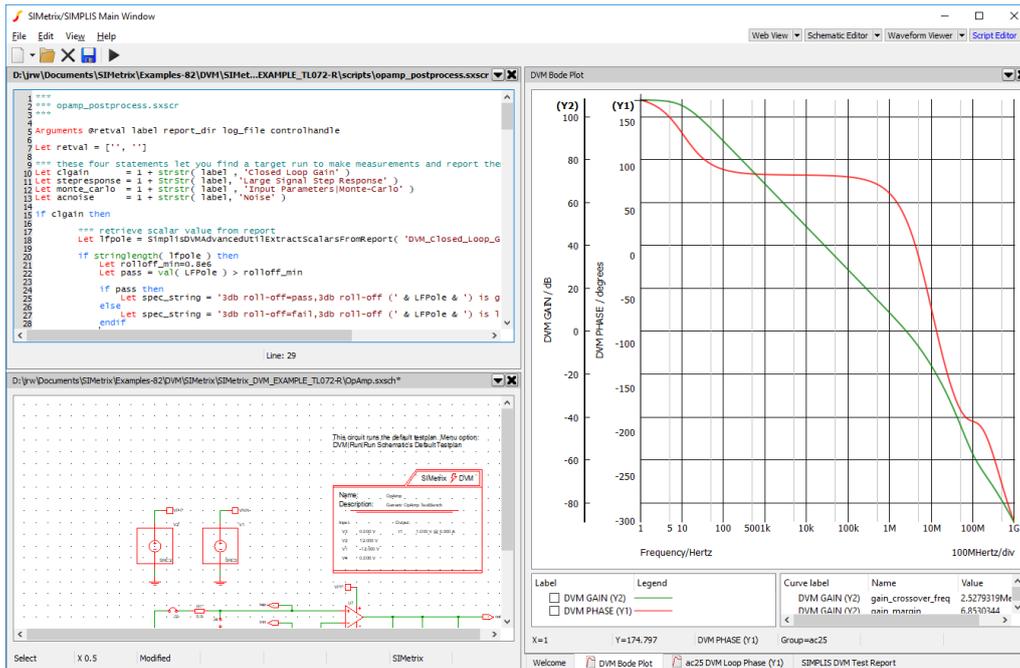
Placement of a View whilst being dragged. If the drag ended at this point in time, the blue region shows where the dragged View would be placed.

By default the View's will split to take up half of the available space when placed next to each other. They can be resized by moving the cursor in-between them until the cursor shown is visible, then dragging them to the required size, as shown in the figure below of two schematics placed side-by-side.



Two schematics placed side by side. The resize cursor is shown between the two Views.

For *Workspace Views*, one View in each window will be selected to have application focus. The focus determines which menus and toolbars are shown and where targeted operations, such as placing a component, will be carried out. Bold title text in the Views title bar, indicates that the view is selected. To quickly determine the type of view that currently has application focus, look for the buttons in the top right of the screen, shown in the figure below of multiple views within the same window. These buttons show the different *Workspace Views* that open in the current window. The type of View that has focus will be highlighted blue. To quickly swap between different *Workspace Views* of different types, click on the button of the type of View you want to see. This will display and focus the application on the last View of that type that was used.



Multiple Views within the same window. The selected View type, Schematic Editor, is shown by the highlighted button in the top right of the window.

Views can be moved to new windows by either dragging them outside of their existing window. Alternately, through the *Undock* or *Dock to...* drop down menu option available by pressing the

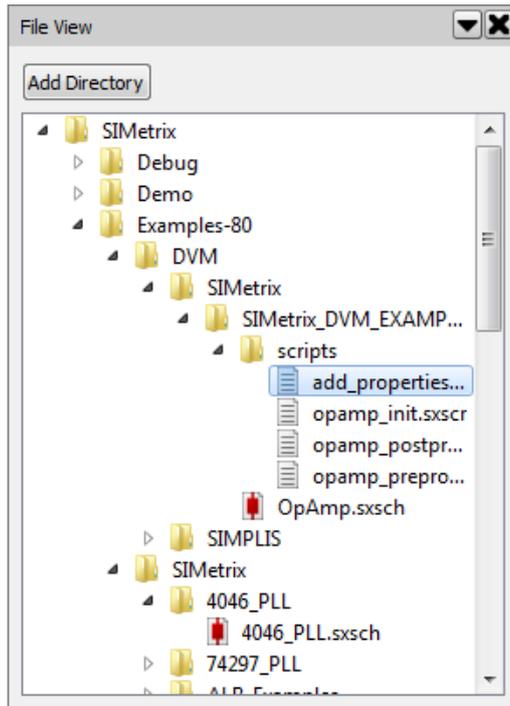


button in the View's title bar. Similarly, all Views of a type can be undocked to another window by using the Undock all drop down menu option of the View button at the top right of the window.

## 5.3 Directory Viewer

The directory viewer is a system view that enables the navigation of SIMetrix related files and directories.

By default, the SIMetrix folder in your Documents directory will appear in this view when you first use SIMetrix, as shown in the figure below. The view will display the folders within the directory along with files that are associated with the program. Double clicking on a file will open it in the program.



The File View. The top level directory, SIMetrix, is the initial default top level directory. Below that, all sub-directories and application related files are shown.

To add a new top-level directory to the viewer, click *Add Directory*, then navigate to the directory to add to the view. When selected, the file viewer will show all sub-directories and files that are associated with SIMetrix or SIMetrix/SIMPLIS.

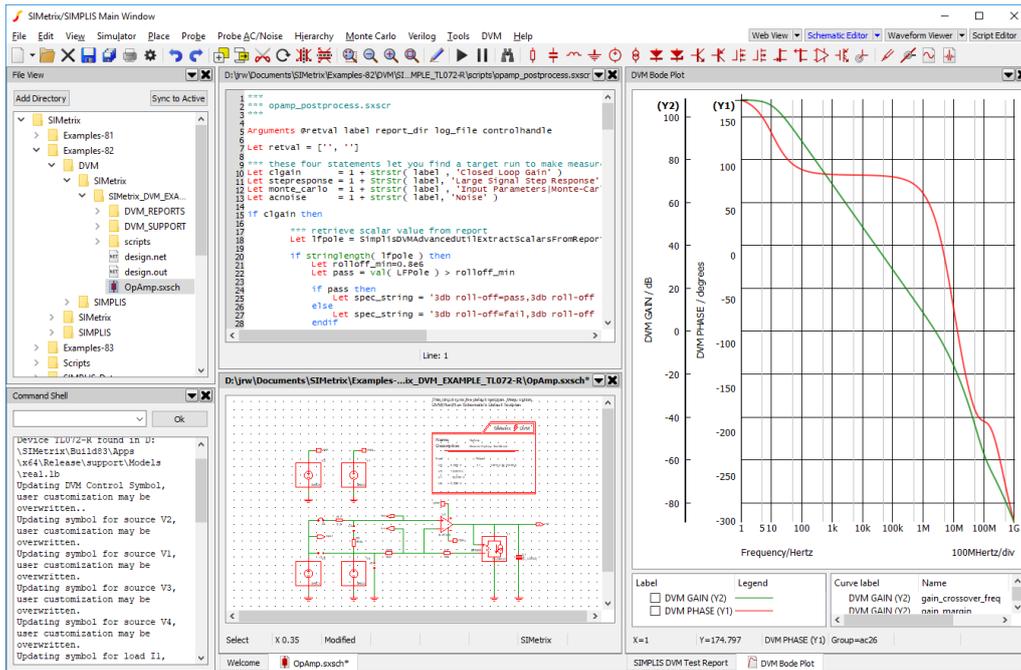
To remove a top-level directory, right click on the directory and press *Unload Directory*.

The right click menu also allows you to open the containing folder for a particular sub-folder or file, create a new folder and refresh the listings (although in most cases the listing will refresh automatically).

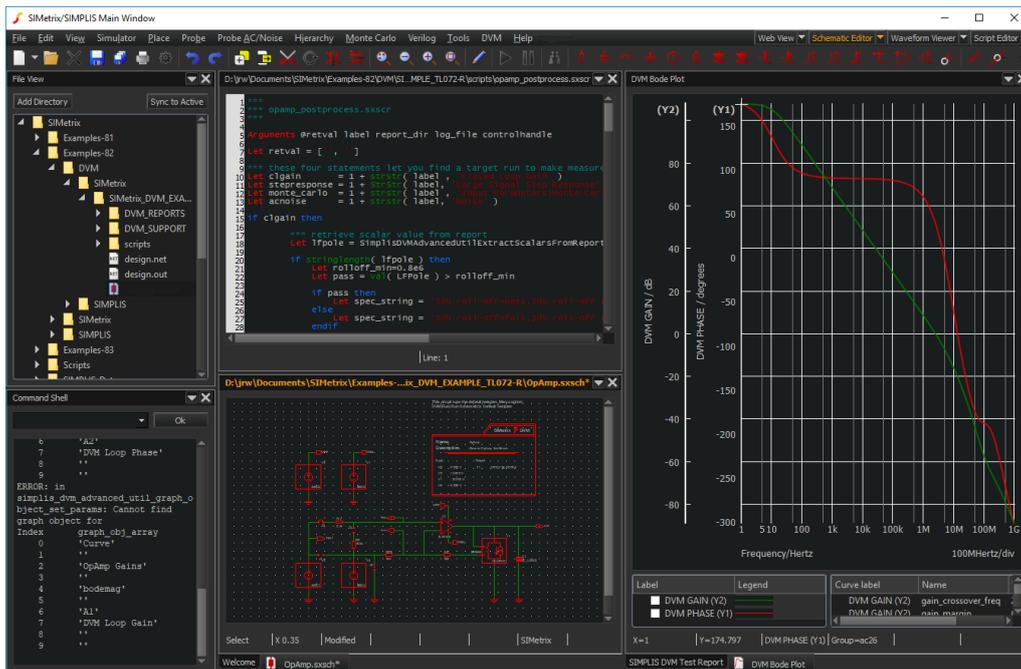
## 5.4 Interface Styles

The style of the user interface can be swapped between two preset styles, a lighter and a darker style. These two styles are shown in the two figures below.

To swap between the two styles use: **File | Options | Interface Colour....** Doing so will also cause some schematic styles to be changed.



Light version of the user interface.



Dark version of the user interface.

# Chapter 6

## Schematic Editor

### 6.1 Fundamentals

#### 6.1.1 Schematic Windows and Sheets

Schematic editor windows are *Content Views* in the SIMetrix GUI Environment. See [“SIMetrix GUI Environment”](#) on page 47.

#### Creating Schematics

If you have SIMetrix/SIMPLIS, the schematic can operate in one of two modes, namely SIMetrix mode or SIMPLIS mode and you can select the mode when creating a new empty schematic as follows:

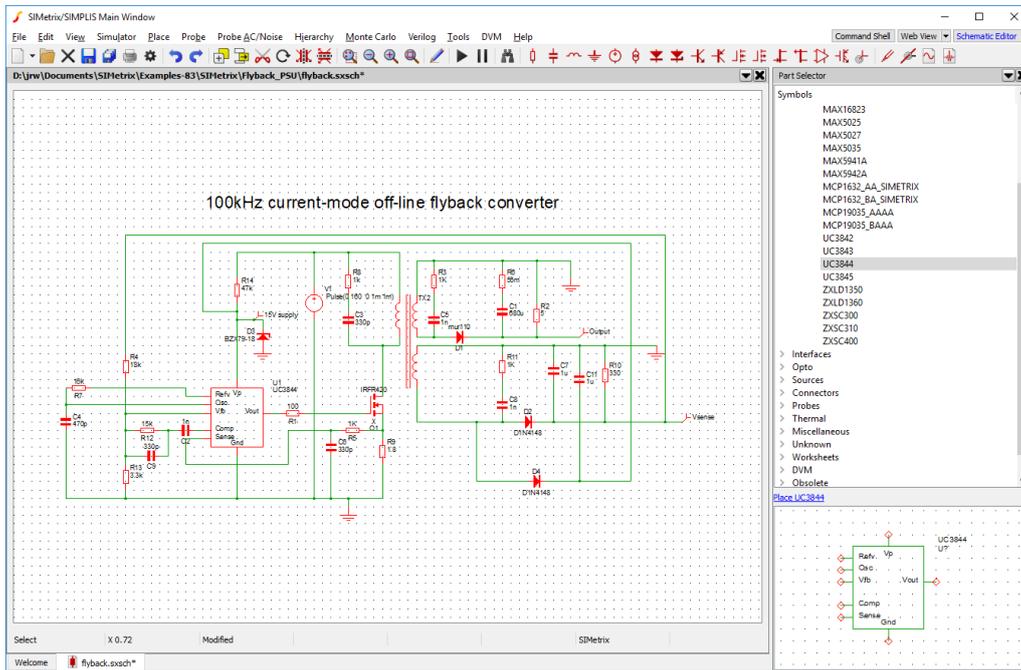
- To create an empty SIMetrix schematic, select menu **File | New | SIMetrix Schematic**
- To create an empty SIMPLIS schematic, select menu **File | New | SIMPLIS Schematic**

If you have a SIMetrix-only product use menu **File | New | Schematic**.

#### Changing Simulator Mode

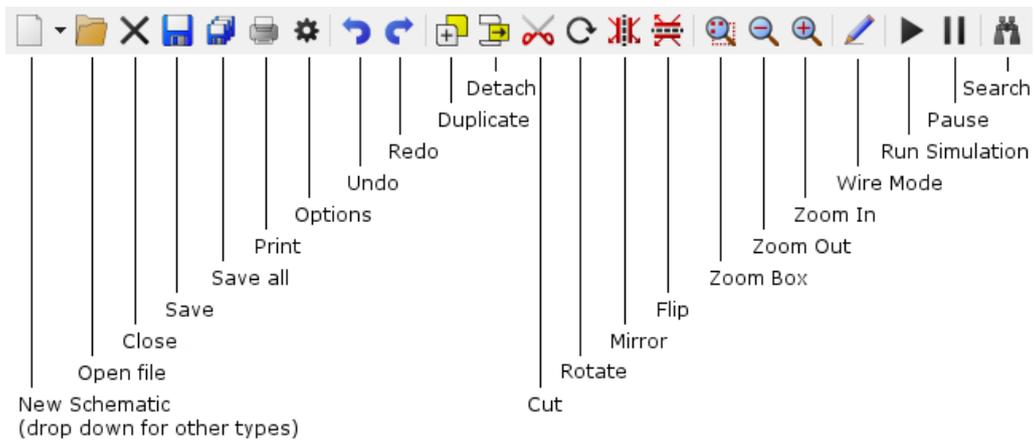
If you have SIMetrix/SIMPLIS and you change the simulator mode, select menu **Simulator | Switch to xxx Mode** where *xxx* is either SIMetrix or SIMPLIS.

### 6.1.2 Schematic Editor Window



### 6.1.3 Editing Operations

In the following notes references are made to the schematic tool bar. The diagram above shows the standard toolbar and the function of each button.



### To Place a Part

Parts are most conveniently placed using the parts selector which is located on the right hand side of the schematic window. If it is not showing, select menu **View | Show Part Selector** to make it visible. The part selector is a hierarchical categorised list containing nearly all parts available. When you find the part you want, click on it then either right click to see menu options or click on the **Place** link between the list and the symbol view.

If you cannot find what you are looking for select menu **Place | Search Part...** to open the SIMetrix search tool. You can also use the **Search** button:



Some commonly used parts can be selected from the parts toolbar.

Once the symbol has been selected, using the mouse, move the image of the part to your desired location then press the left mouse button. This will fix the part to the schematic. Depending on preference settings (menu **File | Options | General...** schematic tab), you may now be presented with another copy of the symbol for placement. Use left key as before to place, press right key to cancel.

You can rotate, mirror or flip the part before placing it on the schematic using the appropriate toolbar button or the keys **F5**, **F6** or **shift-F6** respectively.

If you place a two-terminal part directly over a wire, the wire will broken to allow the part to be connected in-line.

## Selecting a Single Part

Most operations require items to be selected. When an item (part or wire) is selected, it changes colour to blue.

To select a single part, just left click it.

## Selecting an Area

To select all items within a rectangular area of the schematic press the left mouse key in an empty area of the sheet and hold down while dragging mouse to opposite corner of rectangle. A rectangular box should be drawn as you move the mouse. (Note that if the initial cursor position is too close to a wire junction or part, a move operation will be carried out instead of selection.)

## To Change Value or Device Type for a Part

First select it then select schematic popup **Edit Part...** or press **F7**. Alternatively, you can just double click the device. A dialog box appropriate for the type of part will be displayed. For devices requiring a model name, a list of available types will appear

## To Rotate, Mirror or Flip a Part

Click the **Rotate** toolbar button or press key **F5** to rotate a part.

This operation can be performed while a part is being placed or while a block is being moved or copied (see below).

You can also select a part or block then click the **Rotate** button or press the **F5** key to rotate in situ.

To mirror a part or block through the y-axis, click the **Mirror** toolbar button or press the **F6** key.

To flip a part or block (mirror about x-axis), click the **Flip** toolbar button or press **shift-F6**.

## Wiring

See “[Wiring](#)” on page 60.

## Deleting Wires

Select the wire by placing cursor over it clicking left button. Click the **Cut** toolbar button or press delete key.

## Disconnecting Wires

Press the shift key, then select area enclosing the wire or wires to be deleted. Press delete button.

## To Move a Single Part

Place the cursor within it and then drag it using the left mouse key. You can rotate/flip/mirror the part (see above) while doing so.

## To Move More Than One Item

Select items as described above. Place cursor within any of the selected items then drag the items to the desired location. You can rotate/flip/mirror the items (see above) while doing so.

## To Move Items Disconnected

Select items as described above then click the **Detach** toolbar button. Move items to desired location then press left mouse key. You can rotate/flip/mirror the items (see above) while doing so.

## To Move Property Text (Labels)

SIMetrix provides the ability to move property labels simply by dragging them with the mouse but this method is disabled by default. To enable, use menu **File | Options | General...** then in Schematic sheet select **Enable GUI property edits** in the Property editing box.

Although this is of course a convenient method for moving property labels, our recommendation is that this method is kept switched off. Our philosophy is that it is better to move the symbol so that the label is clearly visible rather than move the label itself. See [“Notes on Property Text Position” on page 60](#) for a discussion.

You can also move a part’s value, by pressing **ctrl-F7** and its reference using **ctrl-F8**. To move any other property select device then popup **Properties | Move...** .

## To Duplicate Items

Select items as described above then click the **Duplicate** toolbar button. Move the items to your desired location then press left key to fix. You can rotate/flip/mirror the items (see above) while doing so.

## To Copy Across Schematics

Select block you wish to copy. Choose menu **Edit | Copy**. In second schematic choose **Edit | Paste**.

## To Delete

Select items as described above then click the **Cut** toolbar button or press the delete key.

## Multiple Selection

Individual items which do not lie within a single rectangle can be selected by holding down the **control** key while using the mouse to select the desired items as described above.

## Selecting Wires Only

Hold down shift key while performing select operation.

## Holding Down the ALT Key...

... while selecting will limit part selection to only devices that are wholly enclosed by the selection box.

## Unselecting

Place the cursor in an empty area and press left mouse key.

## Unselect Items Within a Rectangle

You can unselect an area of schematic enclosed by the selection box. Use menu **Edit | Unselect | Rectangle**.

## To Change a Part Reference

Select part(s) then press F8 or select schematic menu **Edit | Change Reference**. Enter new reference.

## To Correct a Mistake

Click the Undo toolbar button. By default you can backtrack up to ten operations (but this can be changed with **File | Options | General...**). If you want to undo the undo operation, select the menu **Edit | Redo** menu item.

## To Add Text To a Schematic

Select the popup menu item **Edit | Add Free Text...** This opens a dialog box prompting you for the text to be entered. After entering text and closing box you can then position the text where you require using the mouse.

## To Change Text Already Entered

Select the text then press **F7** and enter the new text.

## To Hide A Part Value

Select popup menu item **Hide/Show Value**

## To Disable/Enable a Schematic Part

Schematic parts may be disabled so that, for simulation purposes, they behave as if they are not present on the schematic.

To disable one or more parts, first select them then select right click menu **Disable selected**. Each of the disabled parts will grey-out and show with a cross through it.

To re-enable, select then right click menu **Enable selected**.

Any part may be disabled, including regular components, probes, terminals, module ports, ground symbols and bus rippers. Be aware that if you use this method to disable an inline current probe, the probe will become an open circuit.

## To Short-circuit a Schematic Part

Schematic parts may be short-circuited so that they appear as a wire connection for the purposes of simulation. This operation is similar to disabling in that the part is effectively removed from the circuit, but in addition one or more pairs of pins are connected.

To short-circuit one or more parts, first select them then select right click menu **Disable & Short-circuit Selected**. Each of the disabled parts will grey out and show with a thick line connecting one or more pairs of pins.

For two-terminal parts both pins are connected unconditionally. If the part has more than two pins, the connected pair or pairs is determined from the value of the SHORTPINS property. This property is in the form:

$n1-n2[,n3-n4]...$

Where  $n1$ ,  $n2$  etc are pin numbers starting from 1. This is the same as the pin-order as defined in the symbol editor.

For example "1-3" connects pin 1 to pin 3. "1-3,4-6" connects pin 1 to pin 3 and pin 4 to pin 6.

If the part has no SHORTPINS property, no action will be taken and an error message will be displayed in the command shell. Note that the SHORTPINS property will be read directly from the symbol definition if not present on the instance.

## Breaking Wires

If you place a new two pin part onto the schematic and both pins of the part land on the same wire, thus shorting out the part, the wire will automatically be broken and so inserting the part in series.

This will also occur when copying, moving or duplicating existing parts as long as the part was not already short circuited before the operation was started.

The same action will occur for parts with more than two pins if a SHORTPINS property is defined for the part. SHORTPINS is described above [“To Short-circuit a Schematic Part” on page 59](#).

## Zoom Area

Click the Zoom box toolbar button then drag mouse to zoom in on selected area.

## Zoom Full (Fit to Area)

Select popup **View | Zoom to Fit** or press HOME key to fit whole schematic in current window size.

## Zoom Out

Click the **Zoom out** toolbar button or press F12 to zoom out one level.

You may also zoom out by holding down the **control** key and rolling the mouse scroll wheel backwards

## Zoom In

Click the **Zoom in** toolbar button or shift-F12 to zoom in one level.

You may also zoom out by holding down the **control** key and rolling the mouse scroll wheel forwards

## Panning

The easiest way to pan the schematic is with the mouse scroll wheel. Just rotate the wheel for vertical pan. For horizontal pan, hold down the shift key and rotate the wheel.

You may also use the scroll bars, cursor keys and page up and down keys to pan schematic. The left, right, up and down cursor keys pan the schematic one grid square in the relevant direction and the Page up, Page down, control left cursor, control right cursor to pan the schematic 10 grid squares.

## Notes on Property Text Position

The SIMetrix schematic editor has been designed using a basic principle that it is better to move the part to make its property text visible rather than move the property. That way the part's value and other properties will always have a consistent location relative to the symbol body and there will be no confusion as to which part it belongs.

If you have a situation where some device label (=property text) clashes with another, your first thought will probably be to move the label. We ask you instead to think about moving the part that owns the label; it's nearly always a better way. In situations where the label is very long, it might be better to hide it altogether.

If you find that moving the label is the only way then you should be aware of how the positions of property text are defined.

In SIMetrix, property positions can be defined in one of two ways namely *Auto* and *Absolute*. Most of the standard symbols have their properties defined as *Auto*. This means that SIMetrix chooses the location of the property on a specified edge of the symbol and ensures that it doesn't clash with other properties on the same edge. *Auto* properties are always horizontal and therefore easily readable. The position of *Absolute* properties is fixed relative to the symbol body regardless of the orientation of the symbol and location of other properties. When the symbol is rotated through 90 degrees, absolute text will also rotate. *Absolute* properties are intended for situations where the precise location is important, such as in a title block.

When a visible property on a symbol is moved by the method described above, *it and all other visible properties on that symbol* are converted to *Absolute* locations. This is the only way that the positions of all properties can be preserved. This means that once you move a single property on a part, it and all other properties will rotate with the symbol. For this reason, it is better not to move property text until the orientation of the symbol has been finalised.

### 6.1.4 Wiring

#### Overview

SIMetrix offers both manual and smart wiring methods. In smart mode, you select the start and end points and SIMetrix will find a suitable route. In manual mode, you place each wire segment individually in

exactly the locations you require. You don't need to change global settings to select the mode you desire; the procedures for each mode are different and so you can freely switch between them from one wiring operation to the next.

However, in most applications you won't need to use the manual wiring method. The smart wiring method can still be used to enter wire segments one by one, simply by selecting start and end points that have an obvious straight line route. The fundamental difference between smart and manual is that smart mode will always route around obstacles such as existing wire terminations or whole symbols. In manual mode the wire will always go exactly where you take it even if it crosses existing connections or passes through existing symbols.

## Smart Wiring Procedure

1. Initiate smart wiring by bringing the mouse cursor close to either an unselected symbol pin or an unselected wire end. As you do this you will notice that the cursor changes shape to depict a pen symbol.
2. Click the left button (press and release), to mark the starting point of the wire connection.
3. Move, the cursor to the destination point. This may be anywhere on the schematic, not just at a wire end or symbol pin.

If there is a viable route from the start point to the destination point, SIMetrix will locate it and draw the wire route.

## Smart Wiring Notes

The smart wiring algorithm use an heuristic algorithm that finds as many routes as possible then chooses the best one based on a number of criteria. The criteria used in the selection include the number of corners, the number of wires crossed, the number of property labels crossed and its overall length. It attempts to find the most aesthetically pleasing route, but as this is somewhat subjective, it may not necessarily find the route you may have chosen manually. However, in our tests, we have found that it usually finds the best route for situations where there are no more than 2 or 3 corners required. In developing the algorithm we paid particular attention to common scenarios found in analog design such as routing the output of an opamp back to its inverting input and you should find that these common scenarios work well.

## Smart Wiring Options

There is two option to control the smart wiring algorithm. Firstly, you can disable the smart wiring algorithm altogether, in which case the smart wiring procedure will place wires in a similar fashion to the manual wiring methods.

Secondly, there is an option that controls whether or not the smart wiring algorithm is allowed to route wires through existing wires that are connected to the start and end points. By default this option is on, i.e. the smart algorithm is allowed to route through connected wires. If the option is off, the algorithm will not allow any wires in the route to connect to any existing wire regardless of what it is connected to. In general, we recommend that the option is left switched.

To change the smart wiring options, select menu **File | Options | General...** The two wiring options are in the section titled **Wiring**.

## Manual Wiring Procedure

If you have a three button mouse you can start a wire by clicking the middle button. Clicking the middle button a second time will complete the wire and start a new one. Click the right button to terminate wiring.

If you have a two button mouse you can start a wire by pressing **F3** or double clicking the left button. Single clicking the left button will complete the wire and start a new one. Click the right button to terminate wiring.

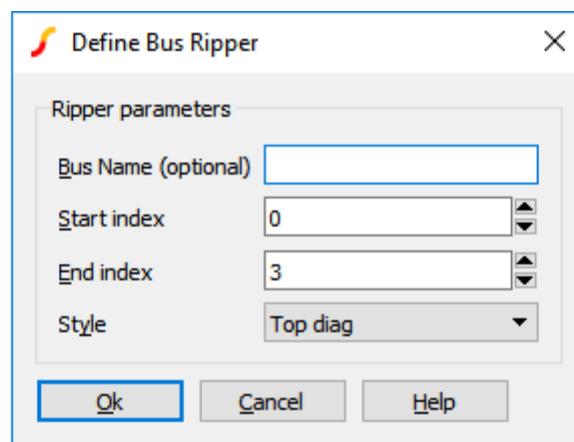
Alternatively, click the **Wire** button on the toolbar. You can start a wire by single clicking the left button, otherwise continue as described above. Press the **Wire** button again to cancel this mode.

### 6.1.5 Bus Connections

SIMetrix provides the Bus Ripper symbol to allow the connection of buses.

#### To Add a Bus Connector

1. Select the menu **Place | Connectors | Bus Ripper...** This will display dialog:



Enter a bus name if you require it.

2. Start index and end index define the wires within the bus that you wish to connect to. Suppose you were connecting to a data bus called DATA and it was 16 bits wide. If you wish to make a connection to the 4 bits from DATA8 to DATA11, you would enter 8 and 11 for the start and end index respectively. The bus ripper doesn't care about the size of the bus to which it is connecting.
3. Choose an appropriate style. This only affects the appearance of the symbol not its functionality.
4. Click Ok then place the symbol on your schematic.

#### To Draw Buses

There is no special method of drawing buses. Simply wire up bus rippers as you would any other part. As soon as you connect to the bus pin of a bus ripper, the colour and thickness of the wire will automatically change to signify that it is a bus.

#### To Increase/Reduce the Connections to a Bus

If you wish to add connections to or delete connections from a bus ripper, select the ripper device and press **F7** or popup menu **Edit Part...** The same dialog as above will be displayed. Adjust the start and end indexes appropriately then close the box.

## Connecting Buses in a Hierarchy

See “[Connecting Buses in a Hierarchy](#)” on page 67.

### 6.1.6 Copying to the Clipboard

To copy schematics to the clipboard, select the entire schematic then choose menu **Edit | Copy**. If you wish the schematic to be copied in black white select **Edit | Copy Monochromatic**. It is recommended that you zoom the schematic to fill the window prior to copying to the clipboard.

After copying to the clipboard, the schematic can be pasted into another application such as a word processor.

### 6.1.7 Annotating a Schematic

You can add a caption or free text to a schematic. The only difference between them is the font style and justification. Captions use a heavy font and are centre justified. Free text use a smaller font and are left justified. To place a caption or free text use the popup or fixed menus:

**Edit | Annotation | Add Caption...**

**Edit | Annotation | Add Free Text...**

### 6.1.8 Assigning Part References

#### Standard Behaviour

As you place parts on a schematic, they are automatically assigned a part reference (R1, Q42, C11 - etc.). These references are assigned in sequence and breaks in the sequence are reused. So if you place resistors on the schematic R1, R2, R3 and R4 then delete R2, the next resistor placed will use the reference R2 that has become available.

#### Setting Start Value

By default, auto assigned references start at 1. You can change this using the AnnoMinSuffix option variable (see the table in “[List of Options](#)” on page 454). For example, type this at the command line:

```
Set AnnoMinSuffix=100
```

Auto assigned part references will now begin with 100.

#### Assigning By Position

You can reassign part references so that they are allocated by their position on the schematic. To do this select menu **Edit | Assign References By Position**.

### 6.1.9 Checking the Schematic

The schematic menu **Simulator | Check** performs a number of checks. First, a netlist of the circuit is created. During this process the following potential errors will be reported.

- Unconnected pins.
- Dangling wires.

- Implicit connections (e.g. two terminal symbols with the same name)
- Name translations. This is for buses with different names connected together. One name has to win.
- Shorted parts. Any parts with two or more pins which have all their pins connected to each other.

Next the netlist is read in by the simulator but the simulation is not started. This will identify any devices for which models have not been found.

## 6.1.10 Schematic Preferences

### Part Toolbar

The default toolbar show a selection of symbols useful. There are however many more buttons available and these can be added as desired. To do this select the schematic menu **View | Configure Toolbar...** . This will display a dialog box allowing full customisation of the part buttons on the schematic toolbar. Note that the toolbar configuration in SIMetrix mode is independent of the configuration in SIMPLIS mode.

Further customisation of all toolbars is possible using script commands. You can also define your own toolbars and buttons. Full details may be found in the *Script Reference Manual*

### Part Placement Options

You can specify whether or not you prefer multiple or single placement of parts. By default, placement of parts from the schematic tool bar is repetitive while placement of parts from the menus is done one at a time. This can be changed. Select the menu **File | Options | General...** In the schematic sheet, the options available are presented in the Placement box.

## 6.1.11 Adding and Removing Worksheets

A number of standard sizes of worksheet are included. See menu **Place | Worksheets**. The worksheet menus automatically *protect* the worksheet after it has been placed. This prevents it from being selected. To delete a worksheet, use the **Place | Worksheet | Delete Worksheet** menu. You should avoid placing a worksheet from the **Place | From Symbol Library** menu as it will not be protected if you do this.

## 6.1.12 Finding and Specifying Net Names

When a simulation is run, a netlist of the schematic is created and this is delivered to the simulator. The netlist generator automatically assigns names to every net (or node) of the circuit. There are some situations where you need to find the name of a net. For example, in noise analysis (see [“Noise Analysis” on page 241](#)) you must specify an output node. In these situations you can either find the name of the net that the netlist generator assigned or alternatively you can specify a name of your choice.

### To Find an Assigned Net Name

Place the mouse cursor on the net of interest. You will see the name appear in the fifth entry of the status box at the base of the schematic window in the form “NET=netname”. Note that the schematic must have been netlisted for this to work. Netlisting occurs when you run a simulation for example, but you can force this at any time by selecting the menu **Simulator | Check**.

## To Specify a User Defined Name

User defined net names can be specified using either the Terminal symbol or the Small Terminal symbol. Select menu **Place | Connectors | Terminal** or **Place | Connectors | Small Terminal**. To specify the net name select the terminal then press **F7** and enter your choice of name.

## 6.2 Hierarchical Schematic Entry

Schematics can be organised into multiple levels in a hierarchy. Typically the top level would contain a number of blocks, each of which represents an underlying child schematic. Each of the child schematics can in turn contain more blocks.

You can create a hierarchical schematic in one of two ways:

The schematic and its symbol are stored within the same file. The combined element is known as a component and is usually given the file extension **.SXCMP**.

All the methods for creating hierarchical schematics described in this section use components.

### 6.2.1 Top-Down Method

#### Creating Part Symbol

Select schematic menu **Hierarchy | Create New Symbol....** This will open the graphical symbol editor. See [“Graphical Symbol Editor” on page 109](#) for details. Note that the symbol you create must be given a REF property typically with the initial value U? and a Model property which must have the value X.

#### Placing Symbol

If the schematic containing the block has never been saved (‘untitled’ in caption bar) you must save it now. This is so that the schematic has a title. This step is only necessary if the schematic has never been saved before.

Select either **Hierarchy | Place Component (Full Path)...** or **Hierarchy | Place Component (Relative Path)...** The first references the component file using a full file system path name while the second uses a path relative to the parent schematic. See [“Placing - Full vs Relative Path” on page 66](#) for more details. Select the **.SXCMP** file you used to save the symbol in the above paragraph. Note that you will see the warning message “Component module ports in underlying schematic do not match symbol pins” displayed in the command shell. This warning may be ignored at this point.

#### Creating Schematic for Block

1. Select the symbol whose schematic you wish to define then select schematic menu **Hierarchy | Descend Into**. Note the symbol must have been saved as a component as described above.
2. A new schematic sheet will be opened with a number of module port symbols already placed. These will be named according to the pin names of the block. You must use these to make connections to the outside world.

## 6.2.2 Bottom-up method

### Creating Schematic

1. Open or draw schematic. It must have at least one Module Port symbol on it. To place a module port, use schematic menu **Hierarchy | Place Module Port**.
2. Save the schematic as a component. Select menu Save As... then select Components from Save as type: list.

### Creating Symbol for Schematic

1. Select **Hierarchy | Open/Create Symbol for Schematic...**
2. A graphical symbol editor window will be opened with a default symbol generated from the number, orientation and names of the module ports on the schematic.
3. The symbol created can be saved straight away or you can edit it to suit your requirements. To save it, in the symbol editor window, select the menu **File | Save...** . You will not usually need to change any of the settings in the dialog. Just press **Ok** to close.

## 6.2.3 Navigating Hierarchical designs

There are a number of means of navigating hierarchical designs. You can go up or down one level or you can jump straight to the top level (or root).

### Descending into a Block

1. Select the block then either press Control-E or select **Hierarchy | Descend Into**.
2. If schematic attached to the block is already open, it will be brought into view. If it isn't it will be opened. Note that the schematic will now be associated with the block that you entered. This is important if you have more than one block attached to the same schematic and you intend to plot curves from it after a simulation. This is explained more fully in the section on simulating hierarchical designs.

### Ascending to Parent Schematic

1. Select **Hierarchy | Ascend**
2. If schematic is open, it will be brought into view. if it isn't, it will be opened.

## 6.2.4 Placing - Full vs Relative Path

Components can be placed using their full path or a relative path.

When placed with a full path, the component file is referenced using its full file system path name (e.g. C:\Projects\Proj123\amplifier.sxcmp). This allows the schematic file that uses the component to be freely moved as long as the component file stays in the same place. However if the component file is moved the schematic will no longer be able to locate it.

When placed with a relative path, the component file is referenced with a file system path name relative to the schematic that uses it. Most likely the component file will be in the same directory (or folder) as the schematic and will therefore be referenced by its file name alone. (E.g. amplifier.sxcmp). This allows the

schematic file and component file to be moved together to any location but they may not be moved individually.

In general we recommend using relative paths wherever possible. The exception is when placing a component that is held in a general library, for example a standard cell used in an integrated circuit design.

### To place a component using its full path

Select schematic menu **Hierarchy | Place Component (Full Path)...** Select a component file then place in the normal way.

### To place a component using its relative path

Select schematic menu **Hierarchy | Place Component (Relative Path)...** Select a component file then place in the normal way.

### Using symbolic constants

SIMetrix has a facility to define path names using symbolic constants. This system allows absolute locations for files to be defined using a single value and thus making it easy to change that location. See [“Symbolic Path Names” on page 440](#) for further details.

### Windows/Linux Inter-operability

Paths are stored on each schematic instance using the UNIX directory separator, that is the forward slash ‘/’. This is provided for compatibility with older versions that run under Linux. (SIMetrix for Linux is no longer available.) This allows schematics created using a Windows version to be used with a legacy Linux version. In most cases Windows accepts a forward slash as a directory separator whereas Linux does not accept a back slash.

## 6.2.5 Connecting Buses in a Hierarchy

### Overview

Bus connections can be passed through a hierarchy in much the same way as normal single wire connections. Bus connections are defined by the underlying schematic. The symbol representing the schematic does not require any special treatment.

### Creating Bus Connections Using the Bottom Up Method

1. Enter the schematic in the usual way.
2. To define a bus connection, place the part Hierarchy | Place Module Bus Port instead of the usual Module Port. Select the device and press **F7** to define the port name and bus size (i.e. the number of wires in the bus).
3. Save schematic as a ‘Component’
4. Select menu **Hierarchy | Open/Create Symbol for Schematic...**
5. Edit symbol if required then save

## Changing the Bus Offset in the Parent Schematic

The bus connection in the parent schematic has a size that is determined by the module port in the child schematic. However, the offset - that is the first wire it connects to in the bus in the parent - can be changed on a per instance basis. To do this, proceed as follows:

1. Select the label next to the bus pin. This will be of the form [A..B] where A is the start wire (default is 0) and B is the final wire. Note that if you edited an existing symbol to add a bus connection, you may not see this label. If so select the component then menu **Hierarchy | Update Bus Connections**.
2. Press **F7** then enter the new offset and OK. You will see the label change accordingly. For example, suppose the bus has 8 wires as defined in the child schematic. To begin with the label will be [0..7] and will therefore connect to bus wire 0 to 7. If you change the offset to, for example, 4, the label will change to [4..11] meaning that the connection will now be made to wires 4 to 11.

## Changing a Non-bus Connection to a Bus Connection

1. In the child schematic change the Module Port to a Module Bus Port and edit as appropriate.
2. In the parent schematic, select the block then menu **Hierarchy | Update Bus Connections**. This will update the schematic to show the bus connection on the hierarchical block.

## Changing the Size of a Bus Connection

1. In the child schematic, select the appropriate Module Bus Port.
2. Press **F7** and enter the new size as required.
3. In the parent schematic, select the block then menu **Hierarchy | Update Bus Connections**.

### 6.2.6 Global Nets

You can access any net at the top level of a hierarchy using a terminal symbol and prefixing the name with '#'.

For example, suppose you have a net at the top level called VCC. You can access this net at any level of the hierarchy without having to pass the connection by connecting a terminal symbol (**Place | Connectors | Terminal**) and then assigning it the name #VCC.

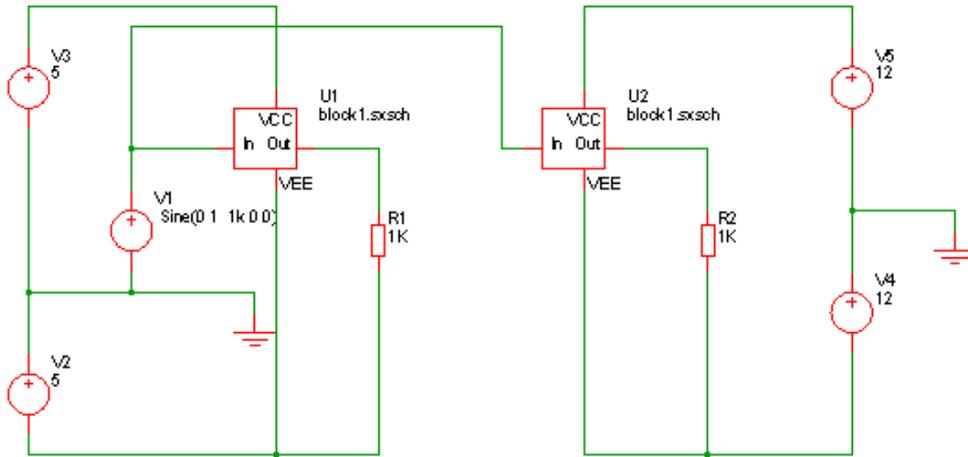
### 6.2.7 Global Pins

Supposing you have two instances of a hierarchical block which you wish to connect to different supply rails. To do this you would need to connect the supplies - say VCC - to pins at the top level with explicit (i.e. non-global) connections at the lower levels. So every child schematic at lower levels would also need VCC pins.

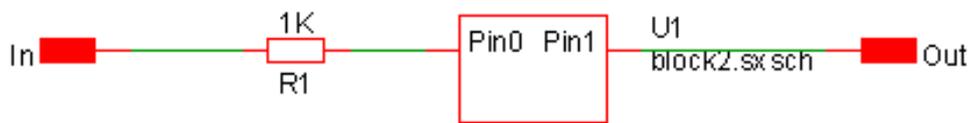
However, it is sometime convenient to hide these connections. When there is only one supply for an entire design, this can be done using global nets. However, in the scenario we described above, there are two versions of VCC so we would not be able to use a global net in this case.

A solution to this is to use a feature of SIMetrix called Global Pins. Global pins are defined during symbol definition. Once a pin is defined as global, a net of the same name will be available in all child schematics at all levels without the need for it to be explicitly passed.

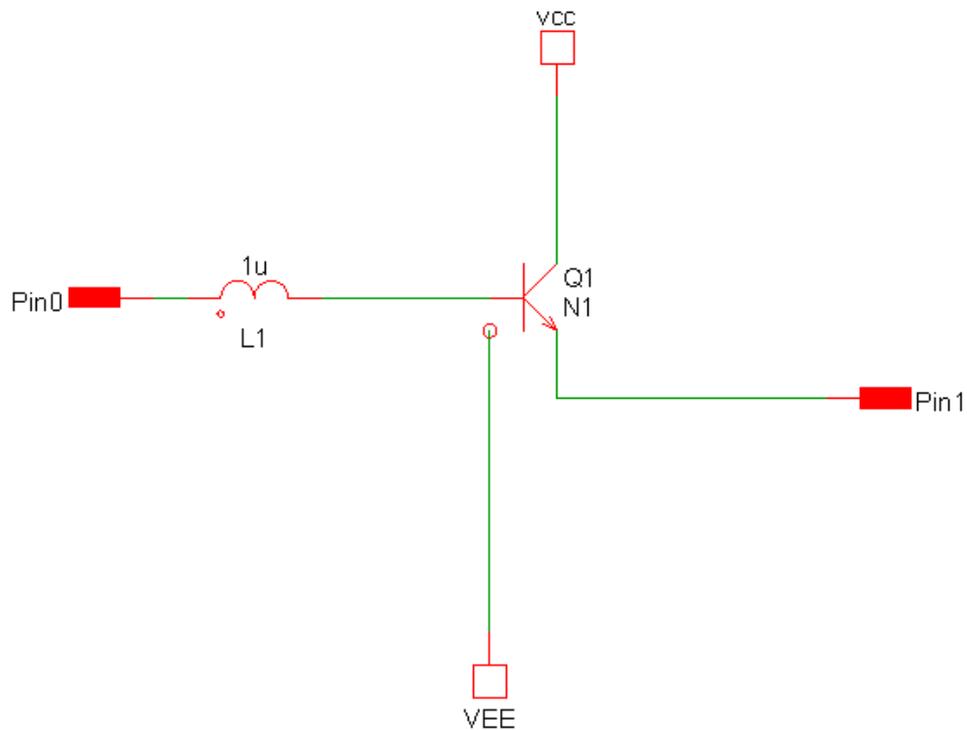
**Example**



Top Level Schematic



Block 1



Block 2

In the above example, VCC and VEE connections have been made in block2 without them having to be passed via the parent block1.

The above trivial example is supplied as an example. See Examples/Hierarchy/Global Pins.

## Creating Global Pins

To define a global pin, select the symbol editor menu **Property/Pin | Global Pins....** Double click on the pin you wish to assign as global and select **Yes**.

### 6.2.8 Passing Parameters Through a Hierarchy

To pass parameters through a hierarchy, assign a PARAMS property then give it a value to assign each parameter you wish to pass (e.g. PARAM1=10 PARAM2=57). See supplied example in folder Examples/Hierarchy/Passing Parameters.

This feature works in both SIMetrix and SIMPLIS runs.

## Adding Parameters to a Symbol

The PARAMS property is most easily added in the symbol editor when the symbol for the hierarchical block is created. This is the procedure:

1. Open the symbol in the symbol editor. If you are editing a hierarchical symbol that has already been created and placed on a schematic, select the symbol then menu **Hierarchy | Open/Create Symbol for Schematic**.
2. In the symbol editor, select menu **Property/Pin Add Property**. This will open the Add Property dialog box.
3. In the **Name** box enter PARAMS.
4. In the **Value** box enter the parameter names and their default values.  
For example: PARAM1=10 PARAM2=57
5. You can leave the remaining settings at their default values or edit as desired.
6. Click **Ok**.
7. Select menu **File | Save...** , then click **Ok** to close. Close symbol editor window if desired.

The above procedure will add the PARAMS property to all new instances of the symbol. It will not add the property to any existing instances already placed on a schematic.

If you have already placed instances of the symbol you can update it so that it acquires the new PARAMS property you have just added to the symbol definition. To do this proceed as follows:

1. Select the instance of the symbol in the schematic editor.
2. Select right click menu **Update Properties...** . Accept the default action - this will add any properties missing from the existing instance but present in the symbol definition.

## Editing Parameters

To edit parameters after they have been added, proceed as follows:

1. Select hierarchical instance.
2. Select right click menu **Edit/Add Properties....**

3. Double click the item with name PARAMS
4. Enter new values as required in the Value box.

## Accessing Parameters in the Child Schematic

You can use any parameter defined on the symbol in an expression to define a component value or model value. You should enclose the expression with curly braces: '{' and '}'.

### 6.2.9 Missing Hierarchical Blocks

When a hierarchical schematic is opened, SIMetrix needs to locate the component files that contain the symbols used for each hierarchical block. If, however, the file for a particular component is missing or is in the wrong location, then SIMetrix will not be able to display that component's symbol. Unlike library symbols, component symbols are *not* stored locally in the schematic file.

In order to make it possible to resolve the problem, SIMetrix instead puts a place holder symbol in place of the missing symbol. The place holder symbol is a diagonal cross.

## Repairing Missing Components

If a component is missing you can either edit the schematic to identify the new location of the component, or you can move files around so that the components are once again in the expected locations.

To edit the schematic, select the place holder symbol then menu **Hierarchy | Replace Component...**

To relocate files, use the system's file handling tools to move the component files, then select menu **Hierarchy | Update Symbols**.

### 6.2.10 Highlighting

The schematic highlighting features will work through a hierarchy. The menus **Edit | Highlight by Net** and **Edit | Other Highlighting | Highlight Net by Name...** will highlight a selected net within the displayed schematic and any connected nets in other schematics in the same hierarchy. But note the following:

1. In very large hierarchies, it is possible that the mechanism that traces through the hierarchy to identify connected nets can noticeably slow down the time taken to descend into a new schematic. Hierarchical highlighting can be disabled if this becomes a problem. See menu **File | Options | General...** then check **Disabled** under **Hierarchy Highlighting**
2. Connectivity information in SIMetrix schematics is normally only generated when a netlist is created. For this reason it is possible for highlighting to be incomplete if a schematic has been edited since a simulation was last run. The highlighting algorithms seeks to minimise this problem by running the netlister at certain times, but for performance reasons does not netlist the whole hierarchy. You can use the menu **Edit | Other Highlighting | Refresh Hierarchical Highlights** to resolve this problem. This will netlist the complete hierarchy and rebuild the highlights from scratch.

### 6.2.11 Copying a Hierarchy

A complete hierarchy may be copied for archival purposes subject to certain conditions as follows:

1. The hierarchy must use relative paths throughout

2. All child components must be either in the same directory as the root or in a directory that is a direct descendant of the root.

If these conditions are met then the hierarchy may be copied using schematic menu **File | Copy Hierarchy**. Before any file copying is started, checks that the above conditions are met is made first. A check will also be made for any existing target files to ensure that no existing file will be over-written. If all checks are successful, you will be presented with a list of all the files that will be copied before the copying operation is started.

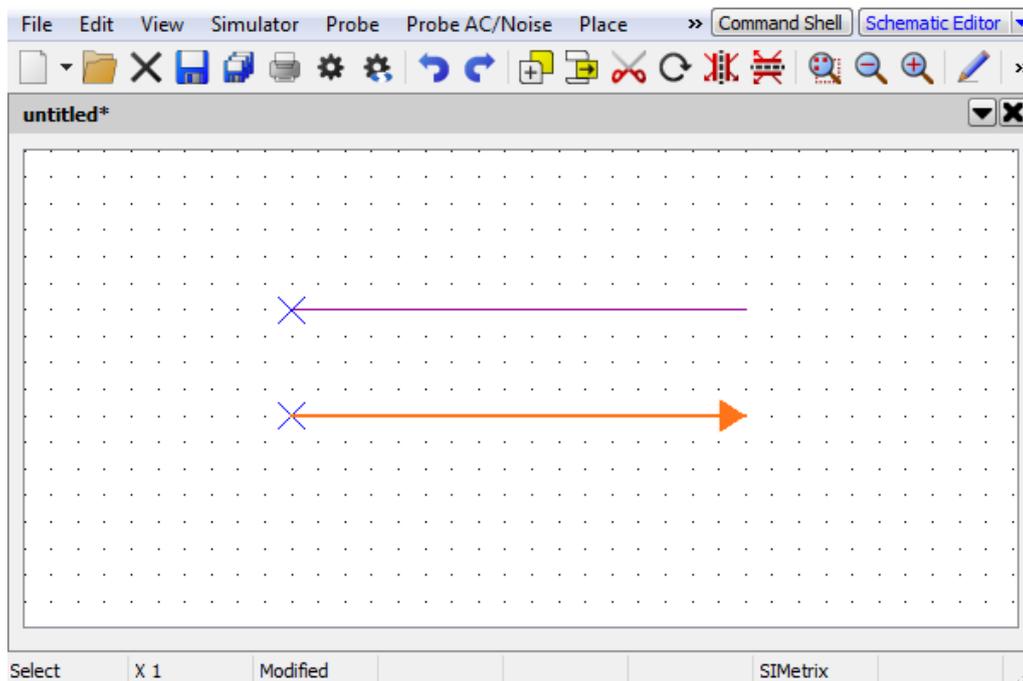
## 6.3 Schematic Annotations

Annotations are elements that can be placed on the schematic that do not alter the behaviour of a circuit, but give a method of adding additional descriptions for someone using the schematic or understanding the design. There are four core annotation types: lines, shapes, images and text. Additionally there is also a title block annotation that allows for marking title and authorship information about a schematic.

### 6.3.1 Line Based Annotations

There are two types of line annotations available: lines and arrows. These can be placed through the **Place | Annotation | New Line** and **New Arrow** menu options.

To place a line based annotation, use the menu link, then click on the schematic where you want the annotation to start and drag to where you want the annotation to end. The annotation can be moved by clicking on the annotation then dragging it. One end of the annotation can be moved by rectangle selecting one end, so that a cross is shown on the annotation like in the figure below, then clicking on the cross and dragging it to where you want that end to now be placed.



A partially selected line and arrow annotation. The cross marker on the left hand end of each shows that only the left part of the annotation is selected.

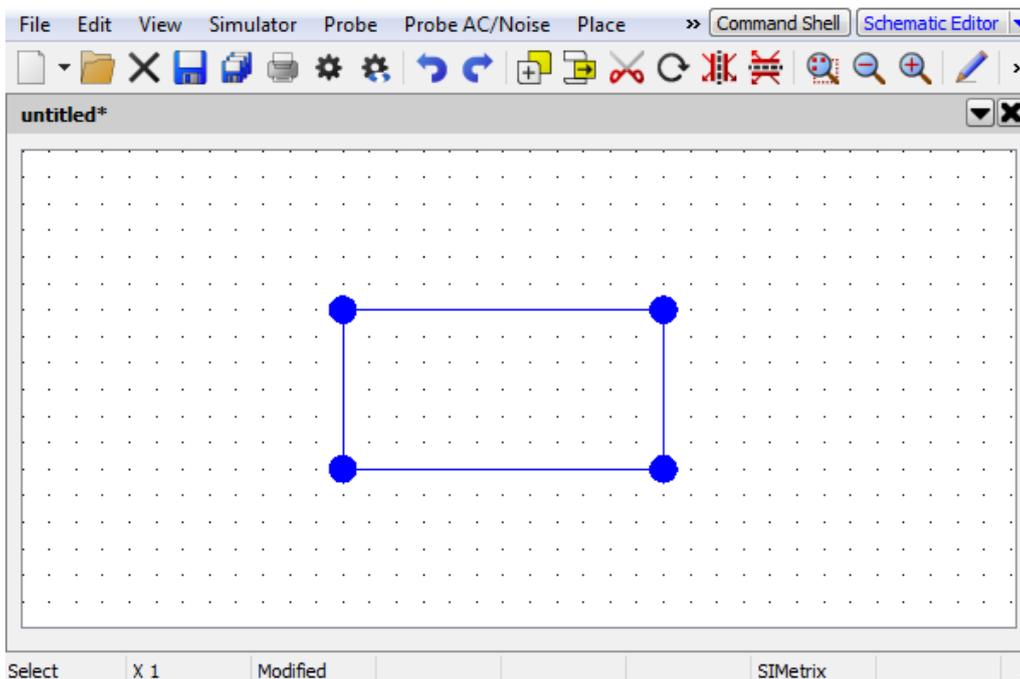
All line annotations can be rotated and mirrored using the appropriate tool buttons.

### 6.3.2 Shape Based Annotations

There are several shape based annotations available, such as rectangles and ellipses.

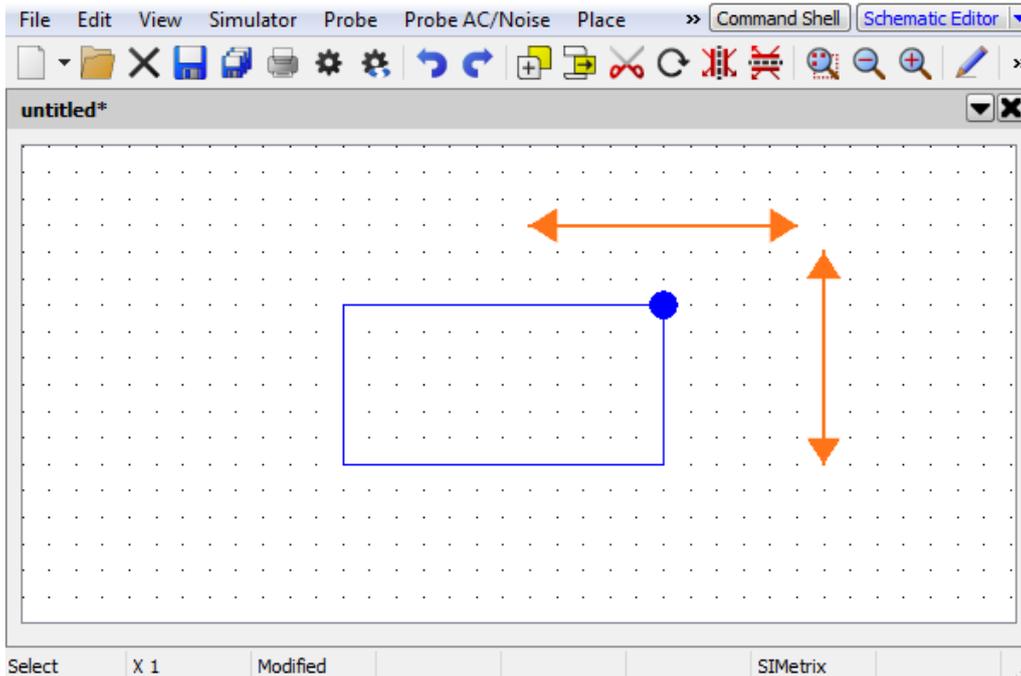
All shape annotations follow the same rules when placing them on the schematic. After selecting the annotation from the **Place | Annotation** menu, click on the schematic where to start drawing the annotation, then drag and release where the annotation should be ended. A preview of the annotation will be shown as you do this. For non-rectangular annotations, the annotations are drawn within a bounding rectangle.

By clicking on the line of a shape you select the whole annotation, as shown in the figure below of a fully selected rectangle annotation. The annotation will display in the selected style, but will also have four dot markers, one on each corner. These dot markers show which corners are selected and will move with the cursor if dragged. If all four markers are shown then it means the whole annotation will move if dragged.

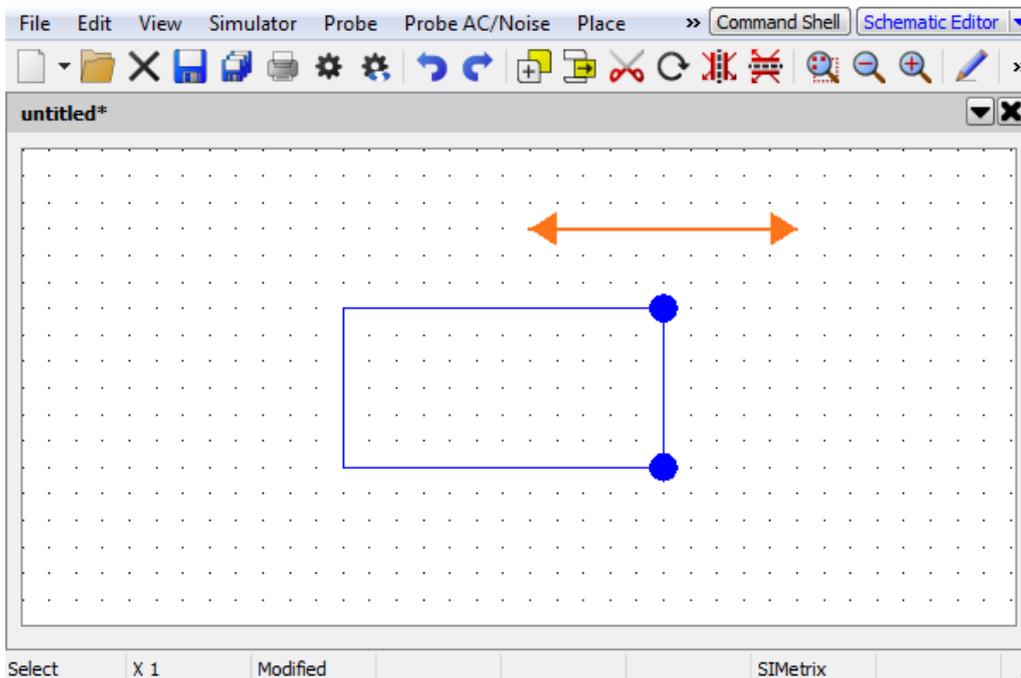


A fully selected rectangle annotation. The dot markers on each corner show that the whole annotation is selected.

Shape annotations can be resized by dragging the corner or side of the shape that you want to resize (in the case of non-rectangular shapes, you resize the bounding rectangle). The directions that the resize can occur on, depend on the corners selected, examples are shown in the following two figures. To select a single corner or two adjacent corners, use the rectangle selection method and ensure the corners you want to select are within the rectangle. Then to resize, click and drag on one of the dot markers.



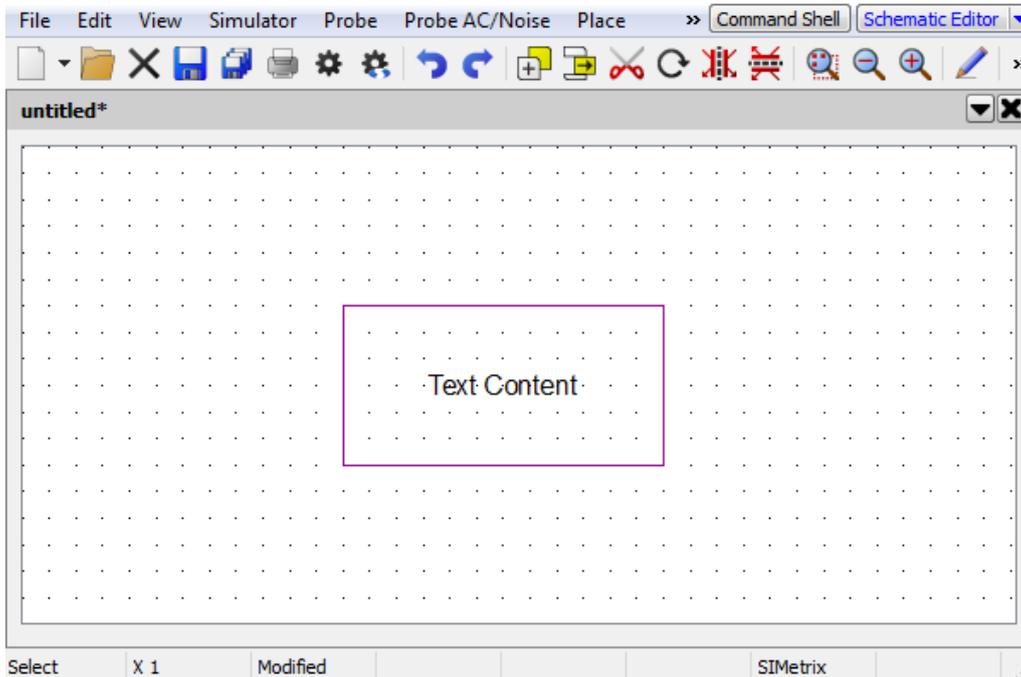
A partially selected rectangle annotation with one corner selected. The dot marker on the top right corner shows that only the top right corner of the annotation is selected. The arrows show the directions of movement that can be made by dragging this marker.



A partially selected rectangle annotation with two corners selected. The dot marker on the top right corner shows that the entire right side of the annotation is selected. The arrows show the directions of movement that can be made by dragging this marker.

Text can be added to shape annotations, as shown in the figure below of a rectangle with text content. To add text to a shape annotation, either double click on the annotation or select the annotation and go to menu **Edit | Annotation | Add Caption to Shape....** This will open a dialog to which you can enter

your text. The text will appear in the annotation with the line formatting given in the text box. If you enter text that is too large to fit the annotation, the annotation will resize to fit the text in.

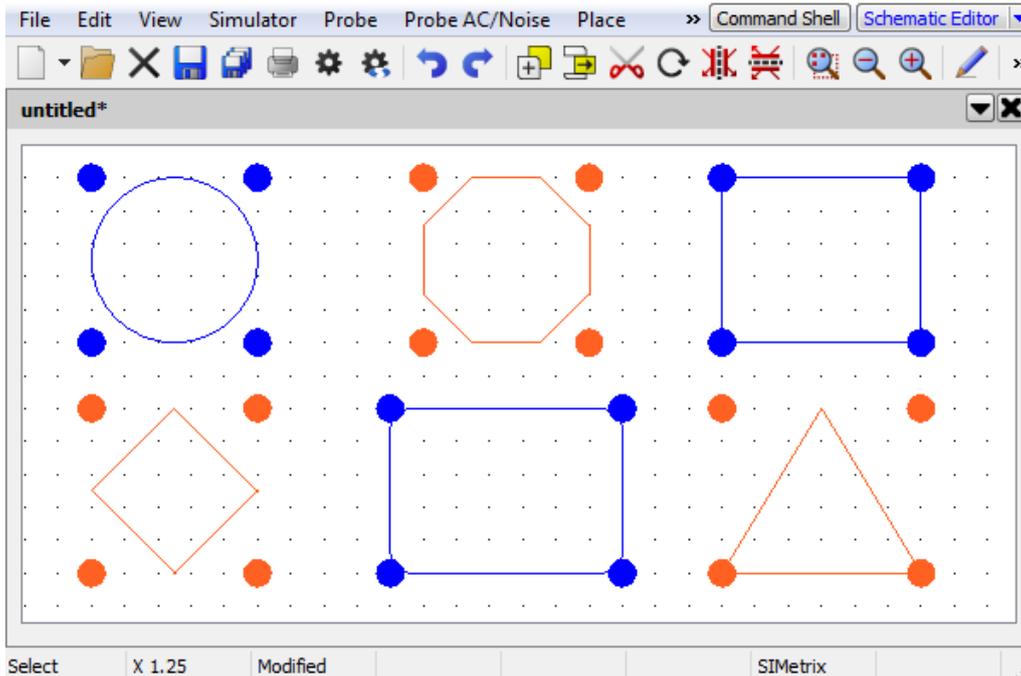


A rectangle annotation with text content.

When text is added to an annotation, clicking anywhere within the annotations shape will cause the annotation to be selected. Without text, only the edge of the shape is selectable.

With multiple lines of text, the justification of the text can be changed. The available justifications are: Left-Align, Centered and Right-Align. These can be changed by selecting the annotation, then going to the menu **Edit | Annotation | Text Alignment** and selecting the alignment required.

The available shapes are: Ellipse, Octagon, Rectangle, Rhombus, Rounded Rectangle and Triangle. An example of each of these, all shown fully selected, is given in the figure below showing all available shape annotations.



Available shape annotations, all fully selected. The dot markers for non-rectangular shapes extend outside of the shapes bounds, to an imaginary bounding rectangle. Different colours are used to easily identify the ownership of dot markers to annotations.

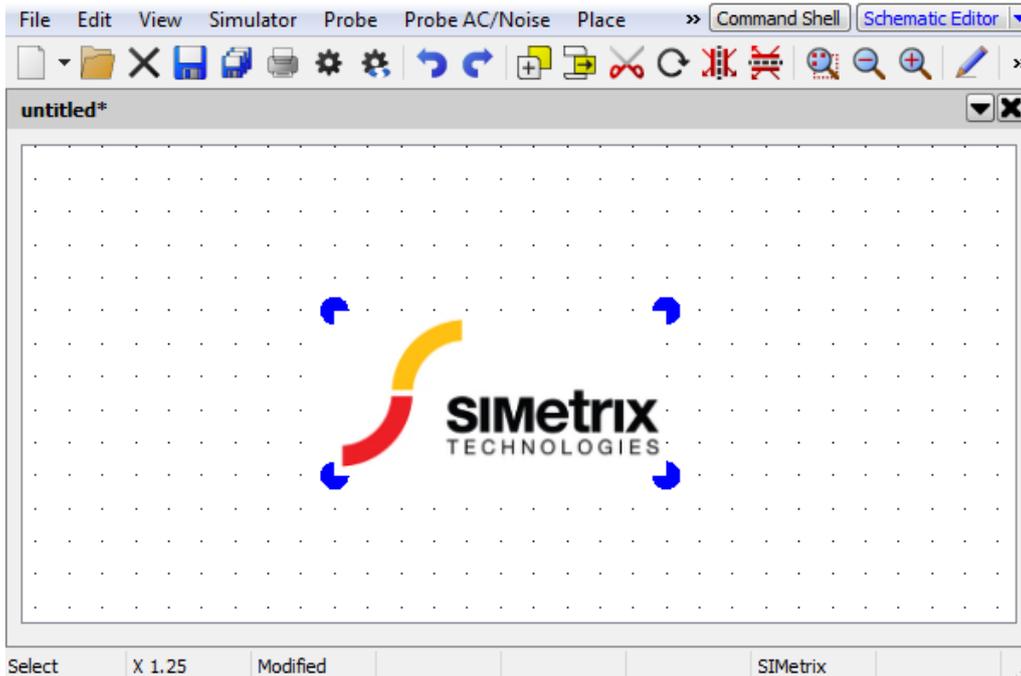
All shape annotations can be rotated and mirrored using the appropriate tool buttons.

### 6.3.3 Image Annotations

Images can be placed within schematics by either dragging the image into the program from an operating system folder, pasting an image in the clipboard or using the menu **Place | Annotation | Add Image...** and choosing a file.

Once in a schematic, the image will be initially resized to a size that is proportional to schematic components and attached to the mouse. You place the image by clicking on the schematic where you want the image to be.

The image can later be moved by clicking and dragging on the image and resized using the same dot marker system used in [“Shape Annotations” on page 73](#).



An image annotation, which has been selected with dot markers visible.

### Warning About Images

Whilst images are scaled visually within the schematic, they will not be compressed in terms of their data content. When saved, images are converted to a text based representation, which will often be larger in size than their native format. Using too large an image therefore may make schematic files excessively large. Images added to a schematic should therefore be scaled to a suitable size *before* placing them into SIMetrix.

Additionally, as images are scaled using a fast scaling process when drawing them to the screen that do minimal anti-aliasing (removal of jagged lines), some images may produce poor looking results when scaled even when shrinking a large initial image. If the image looks poor when placed in a schematic, try resizing the image in an external image editor then add it back into the schematic.

### 6.3.4 Text Annotations

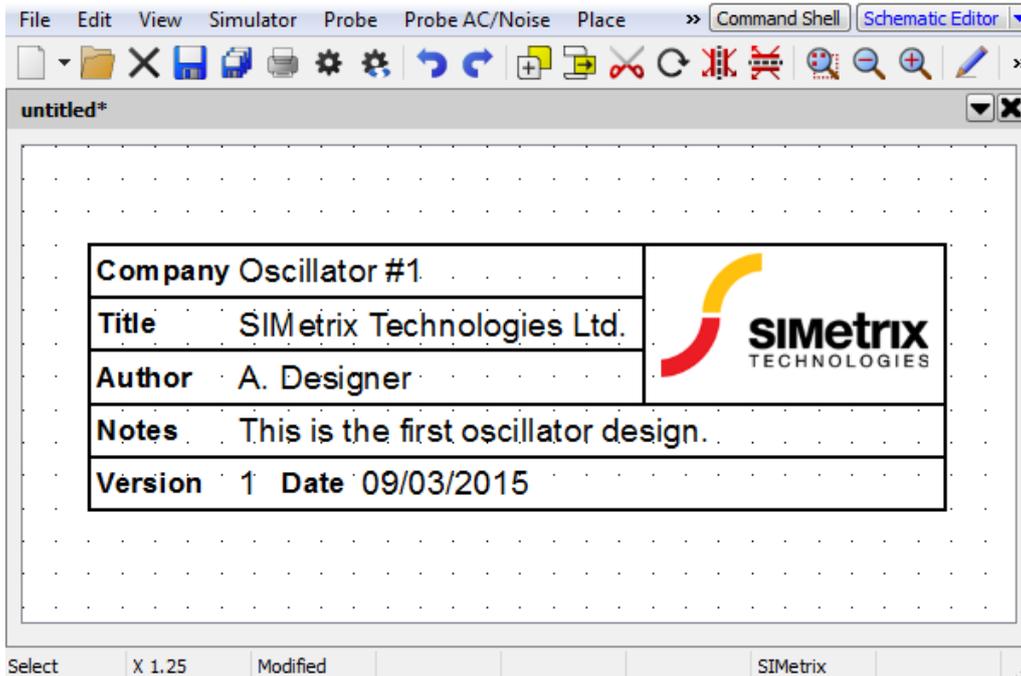
Text annotations can be added separately using the menus **Edit | Annotation | Add Free Text...** or **Edit | Annotation | Add Caption...**. These two menu options produce text using two different styles, both of which can be changed through editing the styles.

Alternatively, text can be copied from elsewhere and pasted directly onto the schematic.

### 6.3.5 Title Block

Title block annotations present information about a schematic such as title, authorship, version and notes, as shown in the figure below. The annotation takes the provided information for these and automatically resizes itself to fit the content correctly.

To create a title block, use the menu **Place | Annotation | Add Title Block...**, which will open a window asking for schematic information. The information provided here will be placed into the title block annotation in the same line format as given, so you may need to add line breaks if necessary.



A title block annotation in the horizontal layout.

The *Company Name*, *Author* and *Logo* fields can have their values set as default, so that for all title blocks they will appear automatically. The *Version* and *Date* values can be set manually or have an auto value assigned. If auto values are selected, they will be set when the schematic is saved.

There are two layouts for the title block: horizontal and vertical. The horizontal layout provides a more detailed title block, with title and author information, notes, version information and a logo picture. The vertical layout is a less detailed layout and only shows the title, authorship and version information.

If an image is used, care should be taken to ensure that the image is not too large. More information can be found in the “[Image Annotations](#)” on [page 76](#) section.

## 6.4 Schematic Styles

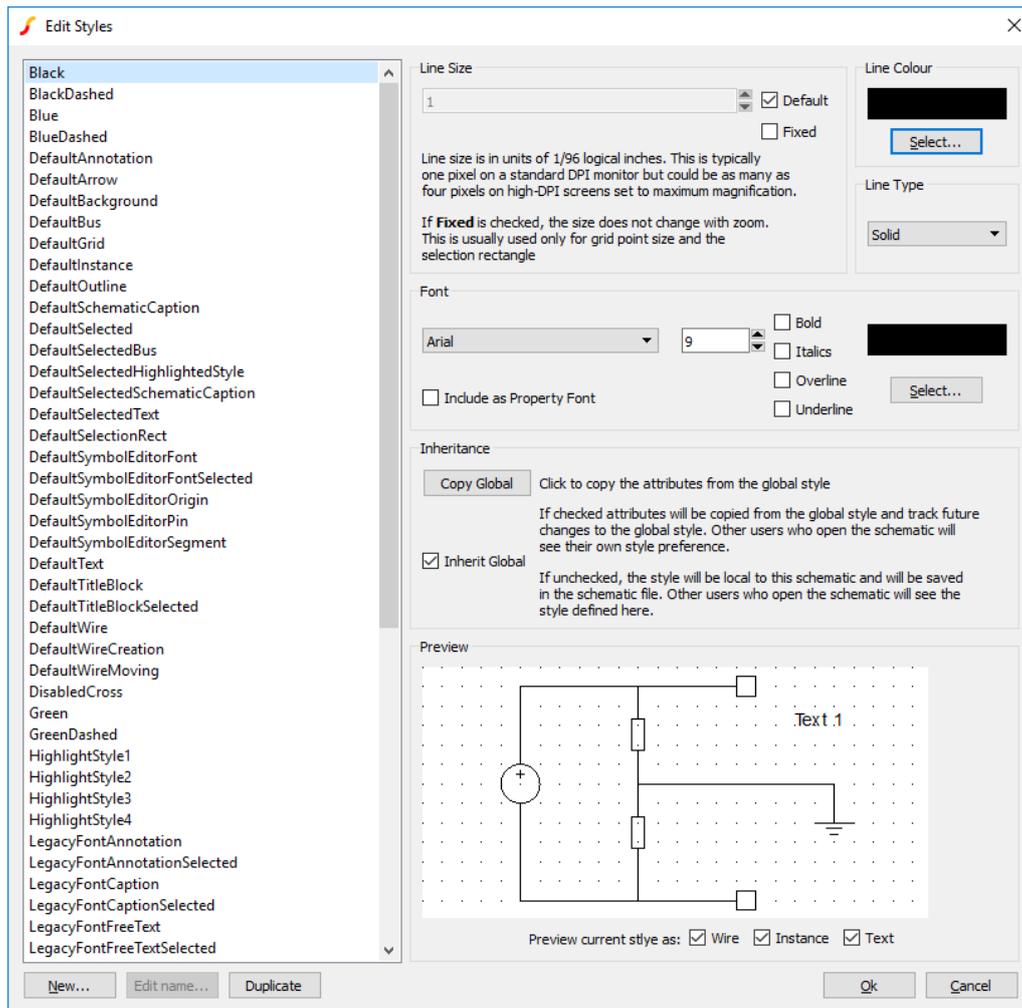
Styles manage the look and feel of the schematic and symbol editors. They control the colour of backgrounds, wires, components and annotations, along with the line type and thickness used.

Styles can exist either globally within your SIMetrix environment, or locally within a single schematic. If local, the styles will transfer with the schematic file so that they appear on other user’s machines with those styles. The other user can override the local styles by selecting menu **Edit | Style | Use Global Styles**

Elements on a schematic have at least two styles associated with them, one for how the element looks normally and one for how the element looks if it is selected. In some advanced cases there may be additional styles associated with a single element.

### 6.4.1 Style Editor

The Style Editor is made of four key components as shown in the figure below of the Edit Styles dialog. These components are: the style names, the line settings, the font settings and a preview window.



The Edit Styles dialog when opened to edit a local style. When opened to edit a global style, the Inheritance group will not be displayed. Each style listed in the left hand panel of names has settings for the lines and fonts used.

To edit a style, first open the style editor by selecting menu item **Edit | Style | Add/Edit All Local Styles...** in the schematic editor. This will allow you to edit the styles used on the schematic currently open. To change the styles globally across the program for all schematics, use the menu **Edit | Style | Add/Edit All Global Styles....**

Next select the style to change from the left hand list of style names. When selected, the right hand panel will show the line and font settings used by this style. The preview window may also update, this is covered more in Preview Window.

The line and font settings can now be changed for this style. In some cases, such as wires, line annotations and text annotations, you may only be interested in editing either the line or font settings. For components, you may wish to change both.

To create a new style, use the **New...** button at the bottom of the style names list. Alternately you can duplicate an existing style by selecting the style to duplicate and pressing the **Duplicate** button. You can then change its name using the **Edit name...** button.

For local styles, you can inherit the global style in which all the characteristics of that style will be copied from the global style. Other users opening the schematic will see their preference as defined by their own global styles. To inherit a global style, check the **Inherit Global** box.

If you edit any characteristic of a local style, the **Inherit Global** box will automatically become unchecked

and that style's characteristics will be saved in the schematic file. Another user opening that schematic will see the style defined here.

## Line Settings

There are settings for line type, line size and line colour.

The line type has the options of: Solid; Dash; Dot; Dash-dot and Dash-dot-dot.

The line size is a numeric value. A size of 1 is one pixel wide and is commonly used throughout the program. Typically most lines would not need to have a line size above 3, with the majority being size 1.

The line colour selection box allows for different ways of inputting the required colour.

## Font Settings

There are settings for font name, font size, font decoration and font colour.

The font name drop down box provides a list of available fonts.

The font size is a numeric value. The preview window will demonstrate the relative size of the font to the schematic.

The check boxes of bold, italics, overline and underline allow additional font styling and decorations to be applied.

The colour selection box allows for different ways of inputting the required colour.

The Include as Property Font check box allows the styles font values to be added to possible style overrides for symbol instance properties. This is covered more in [“Advanced Style Settings” on page 82](#).

## Preview Window

By default, the preview window will display an example schematic using the currently selected style. In the figure at the top for example, the line settings have been used to draw both the wires and symbol instances, whilst the font settings have been used for the text. The background and grid marker colours come from the styles `DefaultBackground` and `DefaultGrid`.

The preview window can be set to allow the current style to only edit some of the preview output. To do this, deselect the tick boxes below the preview window. If deselected, the wires will be drawn using `DefaultWire`, the symbol instances using `DefaultInstance` and the text using the font information from `DefaultInstance`.

## Default Styles

The styles that are automatically used within the schematic editor by default are editable and are named with the prefix `Default`. Below is a table of where the default styles are used.

Style Name	Where used	Uses
<code>DefaultAnnotation</code>	All annotations except arrows and standalone captions and free text.	All settings
<code>DefaultArrow</code>	The arrow annotation.	Line settings
<code>DefaultBackground</code>	The schematic and symbol editor background colour	Line colour

Style Name	Where used	Uses
DefaultBus	Schematic buses.	Line settings
DefaultGrid	Grid used in schematic and symbol editor.	Line colour
DefaultInstance	Symbols in the schematic editor and their properties.	All settings
DefaultOutline	Used on all elements during a drag event to show where they were originally placed.	All settings
DefaultSchematicCaption	Caption annotation font.	Font settings
DefaultSelected	Used on all elements when they are selected, except for overridden exceptions.	All settings
DefaultSelectedBus	Selected override for buses.	Line settings
DefaultSelectedHighlightedStyle	Selected override for highlighted elements.	All settings
DefaultSelectedSchematicCaption	Selected override for schematic captions.	Font settings
DefaultSelectedText	Selected override for schematic free text.	Font settings
DefaultSelectionRect	Click and drag selection rectangle.	Line settings
DefaultSymbolEditorFont	Fonts used in the symbol editor.	Font settings
DefaultSymbolEditorFontSelected	Selected override for symbol editor fonts.	Font settings
DefaultSymbolEditorOrigin	Origin marker in the symbol editor.	Line settings
DefaultSymbolEditorPin	Pin markers in the symbol editor.	Line settings
DefaultSymbolEditorSegment	Symbol line segments used in the symbol editor.	Line settings
DefaultText	Free text annotation font.	Font settings
DefaultTitleBlock	Title block annotation.	All settings
DefaultTitleBlockSelected	Selected override for title block.	All settings
DefaultWire	Schematic wires.	Line settings
DefaultWireCreation	Schematic wires being created by the user.	Line settings
DefaultWireMoving	Schematic wires being moved around the schematic.	Line settings

Additionally, there are legacy font settings used when importing schematics from earlier version of SIMetrix (pre-version 8), which are prefixed with *Legacy*.

## 6.4.2 Simplified Style Editing and Creation

Below are simplified methods for creating or editing styles.

### Creating a Style for a Subset of Elements on a Schematic

To create a style for a selection of elements within a schematic:

1. On the schematic, select the elements to create a new style for.
2. Go to menu **Edit | Style | Create New Style (selected)**.
3. Enter names to be used for the normal style and selected styles that are to be created and press *OK*.

4. The *Edit Styles* dialog will now display, with only the two styles that were created on the previous step showing.
5. Set the styles as required for both the normal and selected styles.
6. Finish by pressing *OK*.
7. The selected elements will now use the new styles.

Note: Styles applied to wires will only remain consistent when changed, if all connected wires share the same style.

## Editing the Styles used by an Element or Selection of Elements

To change the style properties used by a particular element or group of elements:

1. On the schematic, select the element or elements to change the style properties for.
2. Go to menu **Edit | Style | Edit Selected Styles**.
3. The *Edit Styles* dialog will now display, with only the styles used by the selected elements showing in the styles list.
4. Make the changes to the styles as required.
5. Finish by pressing *OK*.
6. All elements using those styles will now be updated on the schematic with those changes. Changes are made to the local styles.

## Changing the Styles used by an Element or Selection of Elements

To change the style used by a particular element or group of elements:

1. On the schematic, select the element or elements to change the style properties for.
2. Go to menu **Edit | Style | Set Normal Style (selected)** for changing the normal style, or **Edit | Style | Set Selected Style (selected)** for changing the selected style.
3. A list of available styles will appear.
4. Choose the style to use and press *OK*.
5. The selected elements will now change to use the requested style.

### 6.4.3 Advanced Style Settings

Below are advanced topics for using styles.

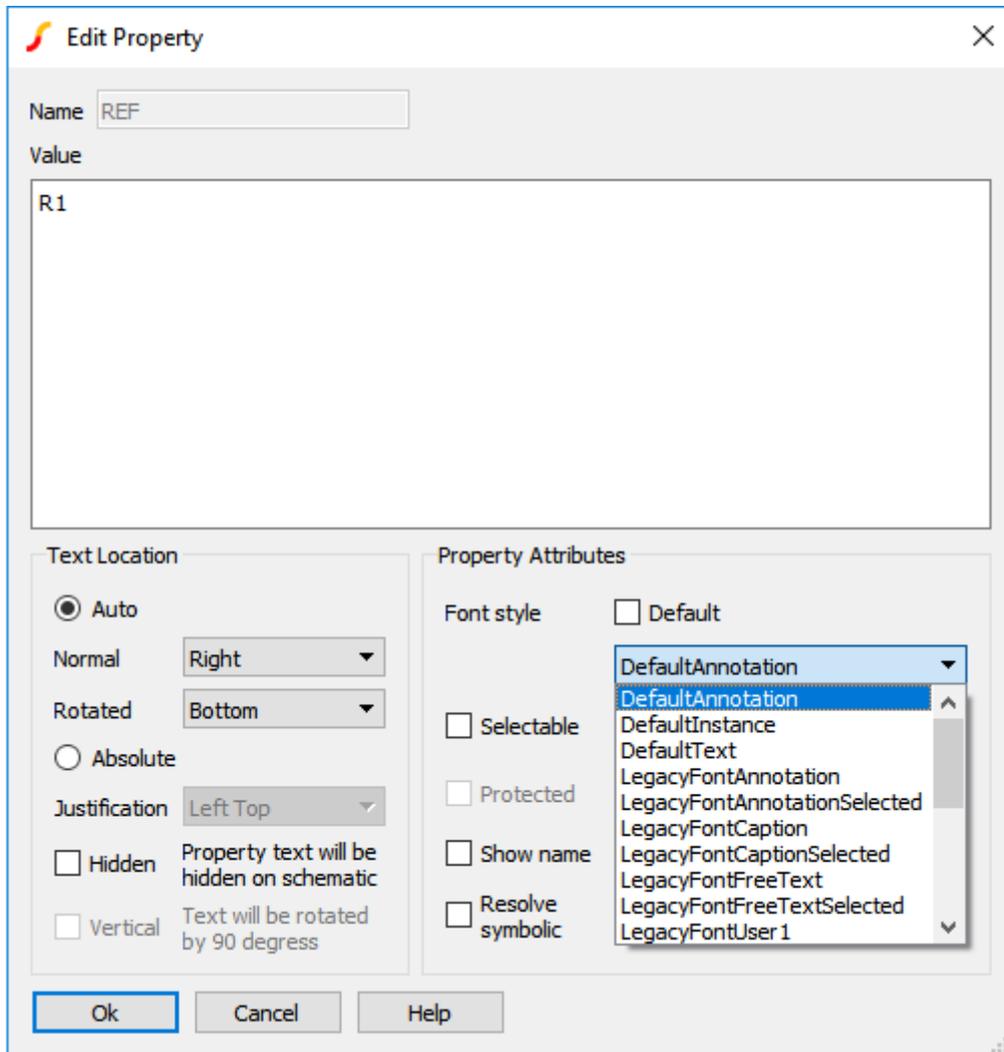
#### Property Fonts

Property fonts are overrides to the standard font used for properties within a symbol instance. Any style can be added to the available property fonts by selecting the **Include as Property Font** checkbox. Checking this on a local style will make the style available to only properties within that schematic, whilst checking it on a global style will make it available to all schematics.

The fonts of a property are then changed by:

1. Right clicking on symbol instance in the schematic
2. Click **Edit/Add Properties...**

3. The Edit Properties dialog will appear, double click on the property to change.
4. The Edit Property dialog will appear. Under Property Attributes, ensure the Font Style checkbox is selected.
5. Default is deselected, then choose a font style, as shown in the figure below.
6. Click OK on all open dialogs to close them and save the changes.



The Edit Property dialog, showing drop down list of override font styles. If default is selected, the font information in the symbol instance's style will be used, otherwise the selected override style will be used for the property text.

Alternatively, the globally available property styles can be altered by using the dialog available in the Schematic Editor menu **Edit | Style | Set Property Styles**.

## Manually Changing Styles Used

The styles used by an element can be manually changed by changing the style properties for that element. The properties to change are *StyleNormal* for the style used when displaying the element normally on the schematic. The style *StyleSelected* is used when the element has been selected by the user. The value of the property should correspond to a style name.

## Symbol Fill Colour

If a symbol has been created with a filled region, the colour used for the filled region can be specified separately from the main style used by setting the *StyleNormalInstanceFill* property and setting the value to a style name. This can also be added in the symbol editor at time of creation.

There is also a dialog that can be used to change the fill colour, in the Schematic Editor menu **Edit | Style | Set Symbol Fill Style**.

## 6.5 Schematic Grouping

Grouping allows the user to interact with a set of elements as if they were interacting with a single element. Any selection of elements can be grouped together and doing so allows the user to subsequently only have to select one of the elements to cause them all to be selected.

To group together a set of elements:

1. Select the elements to group together.
2. Right click and select menu item **Group Selected**, alternately **Edit | Group Selected**.

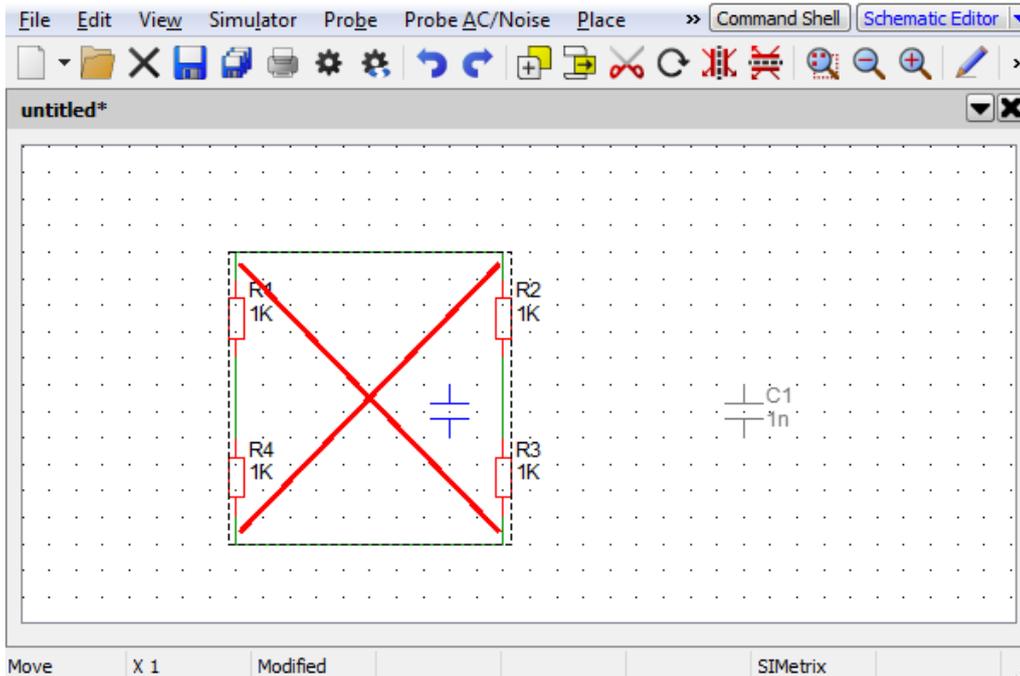
To ungroup a set of elements:

1. Select the group.
2. Right click and select menu item **Ungroup Selected**, alternately **Edit | Ungroup Selected**.

There can be multiple levels of grouping. For example two sets of elements can be grouped individually into Group A and Group B. Later, the two groups can be grouped together to form Group C. To ungroup, either the grouping levels can be ungrouped individually, so for example, Group C ungrouped to leave Group A and Group B, or the entire grouping can be removed by using menu **Edit | Group | Ungroup All Selected**.

### 6.5.1 Group Breaking

If an element is moved onto a group, or a group is moved onto another element, the grouping will break. As shown in the figure below, a visual indicator will be shown whilst elements are being dragged warning if a group will be destroyed if the drag ends in that state.



Moving an element into a group with warning of group destruction. The wires and resistors have been grouped together. The capacitor is being dragged over the grouping. A warning box with a red cross appears around the group to indicate that placing the capacitor here would break the grouping of the elements.

## 6.6 Schematic Printing

### 6.6.1 Printing a Single Schematic Sheet

1. Select menu **File | Print...**
2. If there is a graph window currently open (See [“Graphs, Probes and Data Analysis”](#) on page 287) you can choose to plot the schematic alongside the graph on the same sheet. Select your choice in the Layout section.
3. In the Schematic box select an appropriate scale. **Fit area** will fit the schematic to a particular area relative to the size of paper. If multiple sheets are chosen, a small overlap will be included. **Fixed grid** means that the schematic's grid will be mapped to a fixed physical size on the paper. The sizes are in inches (1 inch=25.4mm). So 0.3 means that 1 grid square on the displayed schematic will be 0.3 inches or 7.5mm on the printed sheet.

### 6.6.2 Printing a Hierarchical Schematic

1. Select menu **File | Print Hierarchy...**
2. You will be presented with a complete list of schematics used in the current hierarchy. Select the schematics you wish print and Ok.
3. Select options as appropriate then Ok.

## 6.7 File Operations

### 6.7.1 Saving

For normal save operations use the **File | Save** or **File | Save As...** menus.

To save all the sheets currently open use **File | Save All**.

### 6.7.2 Exporting Schematic Graphics

You may export schematic graphics to other applications such as word processors or drawing programs. You can do this via the clipboard (see “[Copying to the Clipboard](#)” on page 63) or by writing out to a file. To export schematic graphics to a file, select the schematic menu **File | Save Picture...** then select the format of your choice using the **Save as type:** drop down box. The choices are:

1. **Windows Meta File (.EMF and .WMF)**. Nearly all windows applications that support graphics import will accept this format. Note that this is a scalable format and therefore suitable for high resolution printing.
2. **Scalable Vector Graphics (.svg)**. This is a relatively new format and is not supported by many applications. However, unlike WMF and EMF, it is supported by operating systems other than Windows and may be useful for users wanting to export scalable graphics to other applications that run on other operating systems.
3. **Bitmap - default image size (.png, .jpg, .bmp)** These are available on all platforms, are widely supported by graphics applications but these are not scalable formats and so do not offer good quality when printed using high resolution printers. PNG is the default format if you do not choose a file extension. PNG tends to be the most efficient format to use for images such as schematics and it is also lossless meaning that it uses a compression technique which does not lose information. To choose JPG (JPEG format) or BMP (windows bitmap format) you must explicitly enter .jpg or .bmp file extensions respectively. With this option the image size will match the image size currently displayed on screen. If you wish to specify a different image size, use next option.
4. **Bitmap - specify image size (.png, .jpg, .bmp)**. As 3 above but you must explicitly define the image resolution in pixels. You will be prompted for this when you close the file selection dialog box. Note that schematics always maintain their aspect ratio so the final image size may differ from what you specify. The actual image will always fit within the X and Y values you give.

### 6.7.3 Exporting to Earlier Versions of SIMetrix

From version 8.0, all schematics are stored in an ASCII format. This format is recognised by all SIMetrix versions from version 5.0. So all schematics can be read by all versions from version 5.0 or later. Note the following when reading a version 8 schematic with an earlier version:

- Features introduced in version 8 will not be useable in older versions. In particular, annotations and styles were not available in earlier versions. A version 8 schematic with annotations will open in older versions but the annotations will not be displayed.
- Although the schematic will open, it won't necessarily simulate. This will depend on the devices used in the schematic

The ASCII format used is fully documented allowing the development of translators to other formats. Also there are some editing operations that are easier performed on an ASCII file than with the graphical editor. For example, changing a symbol name is very difficult with the schematic editor as you have to delete and replace all instances. But this is a simple task with a text editor operating on the ASCII file.

### 6.7.4 Binary format

SIMetrix version 7.2 and earlier saved schematics in a binary format by default, although the ASCII format could be used as an option from version 5.0.

#### Opening binary Schematics

No special procedure is needed. Just open the schematic in the usual way. SIMetrix will detect that it is in the binary format automatically. Note that if saved again, the ASCII format will always be used.

#### Saving in binary Format

It is no longer possible to save schematics in the binary format.

#### File Format

Documentation for the format be found on the install CD (see [“Install CD” on page 474](#)) may be found at Docs/File-Formats/schematic-ascii-format-revnnn.pdf. (nnn is the format revision number).

### 6.7.5 Autosave

When enabled, SIMetrix will automatically save all open schematics at regular intervals. This system does not write to the schematic’s normal file but to a backup location. If SIMetrix closes unexpectedly due perhaps to a power failure, you will be asked whether you would like to recover the auto-saved schematics when you restart SIMetrix.

To enable auto saving and to set the auto-save interval, select menu **File | Options | General....** See the Auto-save interval section in the Schematic sheet.

## 6.8 Properties

### 6.8.1 Overview

Properties are one of the schematic editor’s most important concepts. They are actually used for a number of purposes but the most importantly they are used to determine how a schematic device behaves during simulation. A property tells the simulator what type of device it is (resistor, BJT, sub-circuit etc.), another property specifies a device’s value or model name and, for a hierarchical block, a property specifies the file location of the underlying schematic.

For many applications, you only need to understand the meaning of *ref*, *value* and *model* properties. These are explained below but also in [“Adding Standard Properties” on page 116](#). It is also useful, but not essential, to understand the SCHEMATIC\_PATH property used in hierarchical blocks.

### 6.8.2 What is a Property?

A Property is an item of text that is attached to a schematic part to specify some circuit parameter such as a part reference (e.g. R23), value (e.g. 2.2K) or model name (e.g. BC547).

All properties have a name, a value and a number of attributes. A property’s value may be displayed on the schematic. Most attributes determine how the value is displayed; an exception is the protected attribute which determines whether a property is allowed to be modified.

A property can have any name (as long as it does not have spaces in it) and any value. However, certain property names have a special meaning and impart a particular functionality on the part that owns it. These special properties are described in the following table. Note, however, that this is not an exhaustive list as many properties are used for special parts and the behaviour they impart is defined in the script that is used to edit those parts.

Property name	Function
REF	Part reference. (E.g. R23) All circuit devices must have this property and its value must be unique.
VALUE	Part value or model name. (E.g. BC547). All circuit devices must have this property. (This may be confusing. What is described here is a property of name VALUE, not the property's value.)
MODEL	<p>Single letter to signify type of device. For list of signifying letters for each device supported by simulator see <a href="#">“Summary of Simulator Devices” on page 105</a>. If absent the first letter of the part reference will be used instead (as for SPICE) For example, a device with a MODEL property of value Q will always be a BJT regardless of its part reference. MODEL properties of X, H and F have a special significance as follows:</p> <ul style="list-style-type: none"> <li>X Subcircuit instance. PINNAMES specifier will be added to inform simulator of the devices pin names. The simulator will then choose names for device current vectors which will allow cross-probing of currents from the schematic.</li> <li>F Current controlled current source. The standard SPICE CCCS is a two terminal device which uses a separate voltage source for the controlling current. SIMetrix provides the facility to use a single four terminal device with pins 3 and 4 for the controlling current and pins 1 and 2 for the output. Any symbol with four terminals and a MODEL property of F will be treated as a such a device. An additional voltage source will be created by the netlist generator and connected to pins 3 and 4 to be used as the controlling current.</li> <li>H Current controlled voltage source. As F above but has a voltage output.</li> </ul> <p>For a list of valid device types and their signifying letters see <a href="#">“Summary of Simulator Devices” on page 105</a>.</p> <p>(In some respects, the special behaviour of MODEL property values X, F and H is legacy from the past. The recommended method of customising netlist output is to use the TEMPLATE property, but this was not supported in very early versions of SIMetrix.)</p>
NETNAME	If this property is present on a symbol, all nets connected to any of its pins will be named according to the VALUE property. The NETNAME property is used by the Terminal part from menus <b>Place   Connectors   Terminal</b> and <b>Place   Connectors   Small Terminal</b> . The Terminal parts force the net to which it is attached to have a user specified name. (The value of the NETNAME property will be used in the absence of a VALUE property).
SCTERM	Identifies the part as a <i>Module Port</i> . These identify connections in hierarchical blocks.
TOL, LOT, MATCH	These are used for Monte Carlo analysis to specify tolerances. See page <a href="#">“Monte Carlo Analysis” on page 411</a> .

Property name	Function
SCHEMATIC_PATH	Path of schematic in hierarchical designs
MAPPING	Rearranges pin order. This is a sequence of numbers each representing a symbol pin order. The order of the numbers in the mapping is the order in which the schematic symbol pins placed on the netlist.  For example the LMC6762B comparator in the library is assigned a mapping of 1,2,5,3,4. The output on the comparator symbol is pin 5 but the model requires this to be the third node in the netlist entry.
PARAMS	Additional parameters for device appended to value. If MODEL property is X, the keyword PARAMS: prefixes the PARAMS property value.
TEMPLATE	Specifies a customised netlist entry for the device. See <a href="#">“Template Property” on page 89</a> for full details.
VALUESCRIP	Specifies a script to be called when <b>F7</b> or equivalent menu is selected.
INCSCRIP	Script to be called when the shift-up key is pressed. This is to increment a part’s value. Currently used for potentiometers and some passive devices.
DECSCRIP	As incscript but for shift-down to decrement a device.
HANDLE	This property is automatically allocated to <i>every</i> instance and always has a unique value. Because it is automatically added, it is the only property that every schematic part is guaranteed to possess. This property is protected and therefore cannot be edited
SIMULATOR	Determines simulator compatibility. See <a href="#">“Adding Standard Properties” on page 116</a> .

### 6.8.3 Template Property

This is the subject of its own section. See below.

### 6.8.4 Editing Properties in a Schematic

Unprotected properties of a symbol placed on a schematic may be edited using the popup menu **Edit Properties...** This first opens a dialog listing all properties owned by the device. After selecting the property to edit a dialog box similar to the box described in [“Defining Properties” on page 114](#). If the property you select is protected, the dialog box will still open but you will not be able to change any of the settings.

### 6.8.5 Restoring Properties

This is a method of restoring an instance’s properties to the values and attributes of the original symbol. This is especially useful in situations where a symbol has been edited to, for example, add a new property and you wish that new property to be included on existing instances of that symbol.

To restore instances properties follow the instructions below.

1. Select the instances whose properties you wish to restore.
2. Select popup menu **Restore Properties...**

3. There are two options:

**New Properties Only** will only add new properties to the selected instances. That is any property that is present on the symbol definition but not on the schematic instance of it will be added. All other properties will remain intact.

**All Properties** will restore all properties to that of the symbol definition. This includes deleting any instance properties that are not in the symbol definition. In effect this will restore the symbol as if it had just been placed using the **Place | From Symbol Library** menu. Note that REF properties will be automatically annotated to make them unique. *This option must be used with care.* Don't use it unless you are very clear about what it will do.

This function will restore properties according to the local symbol definition stored in the schematic. This won't necessarily be the same as the global definition in the symbol library. For more information see [“How Symbols are Stored” on page 105.](#)

## 6.9 Template Property

### 6.9.1 Overview

The TEMPLATE property provides a method of customising the netlist entry for a schematic part. Normally a single line is created in the netlist for each schematic part (except 'F' and 'H' devices which are defined by two lines). The line is created according to the values of various properties most importantly the *ref*, *model*, and *value* properties. If, however, a TEMPLATE property is specified this system is bypassed and the netlist entry is defined by the value of this property.

The TEMPLATE property value can contain a number of special keywords and characters that will be substituted during the netlist creation. Values that can be substituted include node names, pin names and the value of any property.

There are three template keywords that define multiple lines to implement series or/and parallel combinations, ladder networks or arrays of devices.

### 6.9.2 Template Property Format

The netlist entry for a device with a TEMPLATE property will be the literal text of the TEMPLATE property's value with certain special characters and words substituted. Text enclosed by '<' and '>' are keywords and have a special meaning. Text enclosed with '%' is substituted with the value of the property whose name is enclosed by the '%' character. Finally text enclosed by curly braces, '{' and '}' will be treated as an expression and will be evaluated. Each of these is described in more detail in the following sections.

#### Property Substitution

Any text enclosed with '%' is substituted with the value of the property whose name is enclosed by the '%' character. So %REF% would be substituted with the value of the *ref* property.

#### Expressions

Text enclosed by curly braces, '{' and '}' will be treated as an expression and will be evaluated. Note that property substitutions are performed before expressions are evaluated, so the result of an expression can depend on any combination of property values.

If the attempt to evaluate the expression fails the result will be empty. No error message will be reported.

## Keywords

Any text enclosed by ‘<’ and ‘>’ represents a keyword. The keyword along with the ‘<’ and ‘>’ will be substituted according to the keyword as defined in the following table. There are two types of keyword: simple and compound. Simple keywords are just a single word whereas compound keywords consist of sequence of names and values separated by colons (‘:’). Compound keywords are used to generate multiple netlist lines for applications such as creating series and parallel combinations.

## How Template Properties are Evaluated

Template properties are processed in two passes. In the first pass the property names enclosed by ‘%’ are substituted, while keywords and expressions pass through untouched. In the second pass, keywords and expressions are processed and the ‘%’ character is treated literally.

This makes it possible to use property values in expressions and the control values for the multi line keywords. For example:

```
%MODEL%$%REF% <nodelist> %VALUE% L=%L% W=%W% AD={%W%*0.5u}
```

if L=1u, W=0.5u, MODEL=M, REF=Q23 and VALUE = N1, this would resolve to the following *after the first pass*:

```
M$Q23 <nodelist> N1 L=1u W=0.5u AD={0.5u*0.5u}
```

The second pass would then be able to evaluate the expression and resolve <nodelist> (see below). The value of AD will be calculated according to whatever W is set to. This is an alternative method of setting MOSFET area and perimeter values. (The method used with the standard symbols is different in order to remain compatible with earlier versions).

Note that if the property value contains any of the special characters (‘<’, ‘>’, ‘,’, ‘,’, ‘%’), these will be treated literally. So if for example a property value was tailres, no attempt would be made to evaluate tailres in the second pass.

## Keyword Summary

The keywords available are summarised in the following table and explained in detail below.

Keyword	Description
BUS	Returns name of bus connected to the specified pin
FOREACHPIN	Repeats for each pin
IFD	Conditional on whether a property is defined
IF	Conditional on the result of an expression
INODE	Generates an internal node.
JOIN, JOIN_PIN, JOIN_NUM, JOIN_REF	Returns information about a connected device. Used for current probes.
MAPPEDNODE[n]	As ‘node’ but order defined by <i>mapping</i> property if present
MAPPEDPINNAMES	As pinnames but order is altered according to <i>mapping</i> property if present
NODELIST, NODELIST_H	Substituted with full list of nodes for device.

Keyword	Description
NODENAME	This is not replaced by any text but signifies that the item following is a node name. The netlist generator must be able to identify any text that is a node so that it can correctly substitute the name when required.
NODE[n]	Substituted for individual node
NUMPINS	Returns the number of pins on a symbol
PARALLEL	Start of compound keyword to create a parallel combination.
PARAMSVALUE	Returns passed parameters
PINLIST	Substituted with full list of pin names for symbol
PINNAMES	Equivalent to 'pinnames: <pinlist>' except that no substitution takes place if the /nopinnames switch is specified for the Netlist command.
PROBE	Similar to node but resolves mapped nodes in SIMPLIS mode
REF	SPICE compatible part reference
REPEAT	Start of compound keyword to create a general purpose repeating sequence
ROOTDIR	Root schematic directory in hierarchical design
ROOTPATH	Root schematic full path in hierarchical design
SCHEMDIR	This schematic directory
SCHEMPATH	This schematic full path
SEP	Returns separator character. (Usually '\$')
SERIES	Start of compound keyword to create a series combination
STEP	Used by series and parallel to return sequence number.
T	Substitutes a property value treating it as a template
VALUE	Returns the resolved value

In the following descriptions the square bracket character is used to denote an optional item. Square brackets in bold ('[', ']') mean the literal square bracket symbol.

## BUS

`<BUS [n]>`

Returns the name of the bus connected to the *n*'th pin.

## FOREACHPIN

`<FOREACHPIN:var:<body>>`

Repeats *body* for each pin on the symbol. *var* is a variable that will be assigned with the pin number being processed and may be used inside an expression.

*body* may also use keywords `<node>` and `<pin>` to access connected nodes and pin names respectively.

## IFD

```
<IFD:propname:action1[:action2]>
```

If *propname* is present and not blank, *action1* will be substituted otherwise *action2* will be substituted.

Example

```
ifd:value:<%value%>:1>
```

In the above, if the property value is present it will be substituted otherwise the value '1' will be substituted.

## IF

```
<IF:test:action1[:action2]>
```

If *test* resolves to a non-zero value *action1* will be substituted otherwise *action2* will be substituted. Typically *test* would be an expression enclosed in curly braces. ('{' and '}').

For example, the following implements in a somewhat complex manner a series connection of resistors. (This should actually all be on one line)

```
<REPEAT:line:%SERIES%:<%REF%$R{line} <if:{line==
1}><<NODE[1]>>:$%REF%${line}> <if:{line==
%SERIES%}><<NODE[2]>>:$%REF%${line+1}> %VALUE%;>>
```

Note that usually each action should be enclosed with '<' and '>'. They can be omitted if the action does not contain any keywords. If in doubt, put them in.

## INODE

```
<INODE:name>
```

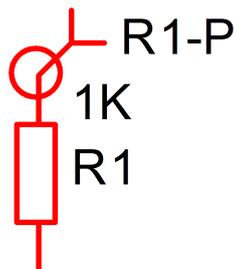
Resolves to a unique node number that is guaranteed not be used anywhere else. The value *name* may be used to identify the number for repeated use within the same TEMPLATE. For example <inode:a> will always resolve to the same node number if used more than once within the same TEMPLATE definition.

INODE is intended to be used to create devices that need multiple netlist lines with connected nodes. This can also be done using a subcircuit but for simple cases INODE may be more convenient.

## JOIN

```
<JOIN:prop_name[:index]>
```

This can only be used with instances of symbols with one and only one pin. Returns the value of *prop\_name* on an instance attached directly to the single pin of the device. For example in the following:



<JOIN:REF> on the probe (R1-P) would return R1 as this is the value of the REF property of the resistor. In situations where more than one instance is connected to the pin, *index* may be used to specify which one. *index* may have a value between 0 and 1 less than the number of devices connected. Use <join\_num> to determine how many devices are connected.

Note that the pin of the device must be directly connected i.e. with pins superimposed and not by wires.

<JOIN> is intended to be used for current probes.

## JOIN\_REF

<JOIN\_REF>

Similar to <JOIN:REF> except that instead of the literal REF property, it returns how the connected instance is identified in the netlist. This takes account of any TEMPLATE property the connected instance possesses or the MODEL property prefix if it does not have a TEMPLATE property.

## JOIN\_NUM

<JOIN\_NUM>

Only valid for single pin instances. Returns number of joined devices. See <JOIN> for details.

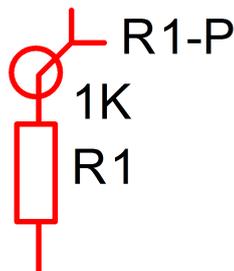
## JOIN\_PIN

<JOIN\_PIN[:index]>

Only valid for single pin instances. Returns connected pin name for another device connected to this device's only pin. This can be used in conjunction with <JOIN> to return the current vector for a part. E.g.

<JOIN:REF>#<JOIN\_PIN>

for the probe device in:



would return R1#p.

In situations where more than one instance is connected to the pin, *index* may be used to specify which one. *index* may have a value between 0 and 1 less than the number of devices connected. Use <join\_num> to determine how many devices are connected.

## MAPPEDNODE

<MAPPEDNODE [n]>

Same as NODE except that the mapping property is applied. The mapping property is used to rearrange nodes in a different order than defined on the symbol. It is used by the model-symbol association system to allow a single symbol to be associated with multiple models that may not necessarily have the same terminal order.

## MAPPEDPINNAMES

<MAPPEDPINNAMES>

As PINNAMES but with the mapping property applied. See MAPPEDNODE above.

## NODELIST, NODELIST\_H

<NODELIST[:map[|nox]]>  
<NODELIST\_H[:map[|nox]]>

Replaced by the nodes connected to the device's pins. NODELIST\_H includes hidden global pins (see [“Global Pins” on page 68](#)) used in hierarchical schematics whereas NODELIST omits these. Has two options:

map	If present will order the nodes according to the MAPPING property.
nox	If present, will disable XSpice pin attributes. See <a href="#">“Adding XSpice Pin Attributes” on page 113</a> for details.

## NODENAME

<NODENAME>

This is not replaced by any text but signifies that the item following is a node name. The netlist generator must be able to identify any text that is a node so that it can correctly substitute the name when required. For example, the following is the template definition of the N-channel MOSFET with bulk connected to VSS:

```
%model%$%ref% <nodelist> <nodename>vss %value%
```

If VSS were actually connected to ground, the netlist generator would replace all nodes called VSS with 0 (meaning ground). If the <nodename> keyword were not present in the above the netlist generator would not be able to determine that VSS is a node and the substitution would not take place.

## NODE

<NODE[n]>

Replaced by the individual node identified by n starting at 1. So node[1] is node name connected to the first pin on the symbol.

## NUMPINS

<NUMPINS>

Returns the number of pins on the symbol owning the TEMPLATE property.

## PARALLEL

```
<PARALLEL:num:<line>>
```

Creates a parallel combination of the device described in *line*. For example:

```
<parallel:%parallel%:<<ref> <nodelist> %VALUE%>>
```

creates a parallel combination of parts. The number in parallel is determined by the property PARALLEL. Note that the REF keyword returns the part reference appropriately modified by the MODEL property *and* appended with the sequence number. If PARALLEL=5, REF=R1, VALUE=1k, MODEL=R and the device is connected to external nodes R1\_P and R1\_N, this is the end result.

```
R1$1 R1_P R1_N 1k
R1$2 R1_P R1_N 1k
R1$3 R1_P R1_N 1k
R1$4 R1_P R1_N 1k
R1$5 R1_P R1_N 1k
```

If the *num* element is empty - e.g. in above example if PARALLEL property were empty or missing - then no output will be made at all.

The example above can be used for any two terminal part. There must however be a PARALLEL property present on the symbol.

## PARAMSVALUE

```
<PARAMSVALUE>
```

Returns the instance's parameters. This is defined by the PARAMS property and will be prefixed with the parameter separator for subcircuit devices. This is params: for SIMetrix mode and vars: for SIMPLIS mode. This keyword will also include tolerance parameters defined by the properties LOT, TOL and MATCH if present.

## PINLIST

```
<PINLIST>
```

Replaced by the symbol's pin names.

## PINNAMES

```
<PINNAMES>
```

Equivalent to 'PINNAMES: <PINLIST>' except that no substitution takes place if the /nopinnames switch is specified for the Netlist command.

## PROBE

```
<PROBE [n]>
```

In SIMetrix mode, behaves identically to NODE. In SIMPLIS mode, will return the mapped node name if relevant. This will happen if the node has a name defined by a terminal symbol. Instead of the assigned node number this keyword will return the node name prefixed by a '#'. As the name implies, this is intended for use with probe symbols.

## REF

```
<REF>
```

Returns the part reference of the device using the same rules that are used when the TEMPLATE property is not present. The rules are:

IF MODEL property is blank OR MODEL is a single character AND first letter of REF property equals MODEL property

```
<REF> = %REF%
```

otherwise

```
<REF> = %MODEL%<SEP>%REF%
```

Where <SEP> is the separator character. This is usually '\$' but can be changed at the netlist command line. See Netlist command syntax in the *Script Reference Manual/Command Reference/Netlist*.

If <REF> is used for a series or parallel repeat sequence, it will be appended with:

```
<SEP><STEP>
```

where <STEP> is the sequence number for the series/parallel function. See below.

## REPEAT

Repeats *line*, *num* times. *var\_name* is incremented on each step. *var\_name* may be used in an expression to define device or node names.

The following example creates a subcircuit that define an RC ladder circuit with a variable number of sections defined by the property NUM. The resistance of each section is defined by the property RES and the capacitance by the property CAP. Note that, as explained above, templates are resolved in two passes. In the first pass the property names enclosed by '%' are substituted with their values while expressions and keywords are left untouched. In the second pass the keywords and expressions are processed.

```
.subckt ladder 1 {%NUM%+1} gnd
    <repeat:idx:%NUM%:<X{idx} {idx} {idx+1} gnd section;>>
    .subckt section in out gnd
    R1 in out %RES%
    C1 out gnd %CAP%
    .ends
    .ends
```

*var\_name* in the above is set to *idx*. If NUM were set to ten, the line:

```
X{idx} {idx} {idx+1} gnd section;
```

would be repeated 10 times with *idx* incrementing by one each time. Note the semi-colon at the end of the line. This signifies that a new line must be created and is essential. The end result of the above with NUM=10, RES=1k and CAP=1n is:

```
.subckt ladder 1 11 gnd
X1 1 2 gnd section
X2 2 3 gnd section
X3 3 4 gnd section
X4 4 5 gnd section
X5 5 6 gnd section
X6 6 7 gnd section
X7 7 8 gnd section
X8 8 9 gnd section
```

```

X9 9 10 gnd section
X10 10 11 gnd section

.subckt section in out gnd
R1 in out 1k
C1 out gnd 1n
.ends

.ends

```

Although it is legal to nest REPEAT keywords, we recommend avoiding doing so as it can lead to unexpected results. You can always use subcircuit definitions to each multi-dimensional repeats and these are usually easier to understand.

The above example has multiple lines. These can be entered using the Edit Properties dialog box. The best way to define multiple line templates is to first enter them in a text editor and then copy and paste to the Edit Properties dialog.

## ROOTDIR

Returns the root directory in a hierarchical design. If the design is not hierarchical, returns the same value as <SCHEMDIR>.

## ROOTPATH

Returns the full path of the root schematic in a hierarchical design. If the design is not hierarchical, returns the same value as <SCHEMPATH>.

## SCHEMDIR

Returns the directory where the current schematic is located.

## SCHEMPATH

Returns the full path of the current schematic.

## SEP

```
<SEP>
```

Returns separator character used to separate the device letter and part reference. This defaults to '\$' but can be changed at the Netlist command line. See Netlist command syntax in the *Script Reference Manual/Command Reference*.

## SERIES

```
<SERIES:num:<line>>
```

Creates a series combination of the device described in *line*. For example:

```
<series:%series%:<<ref> <nodelist> %VALUE%>>
```

Creates a series combination of parts. The number in series is determined by the property `SERIES`. Note that the `REF` keyword returns the part reference appropriately modified by the `MODEL` property *and* appended with the sequence number. If `SERIES=5`, `REF=R1`, `VALUE=1k` and `MODEL=R` and the device is connected to external nodes `R1_P` and `R1_N`, this is the end result.

```
R1$1 R1_P 1 1k
R1$2 1 2 1k
R1$3 2 3 1k
R1$4 3 4 1k
R1$5 4 R1_N 1k
```

If the *num* element is empty - e.g. in above example if `SERIES` property were empty or missing - then no output will be made at all.

The example above can be used for any two terminal part. There must however be a `SERIES` property present on the symbol.

## STEP

```
<STEP>
```

Used with `SERIES` and `PARALLEL` keywords. Returns sequence number.

## T

```
<T:prop_name>
```

Does the same as `%prop_name%` except that the properties value is evaluated as if it were a template itself. With `%prop_name%` the literal value of the property is always used. Note that recursive properties will simply be substituted with nothing. E.g. `<T:TEMPLATE>` will return empty if used in a template property called `TEMPLATE`.

## VALUE

```
<VALUE>
```

Returns the `VALUE` property value unless the instance is a hierarchical block in which case it returns the name of the referenced subcircuit definition.

## Further Information

To put a new line in the netlist entry you can use a `;`. Literal new lines are also accepted.

To put a literal `<>`; or `%` character in the text, use two of them. So `'<<'` will become `'<'`.

### 6.9.3 Template Scripts

It is also possible to write a script to compile the line generated by the netlist generator. Such scripts are called 'Template Scripts'. With this approach, you enjoy the full power of the scripting language and very complex devices can be created in this manner. For full details of this approach, please refer to the *Script Reference Manual/Applications/Schematic Template Scripts*.

## 6.10 Load Component Values

The Load Component Values feature allows you to quickly change component values in your design without having to edit each component individually. Multiple designs that use the same schematic can be

configured easily by saving a configuration file containing a set of component values. Each design can then be configured by loading the appropriate configuration file. Refer to the following topics:

- “Overview” on page 100
- “Menus” on page 100
- “Configuration File Format” on page 100
- “Creating a Configuration File from the Schematic” on page 101
- “Loading a Configuration File” on page 101
- “Automating Load Component Values with Command line Switches” on page 102
- “Automating Load Component Values with a Script” on page 102

### 6.10.1 Overview

The **Load Component Values** feature can change any schematic symbol property, however; the schematic configuration, including symbol placement and wiring, are unchanged. The **Load Component Values** feature works for both the SIMetrix and the SIMPLIS simulator modes.

**CAUTION:** Be sure to backup your schematic before loading a set of component values. When you load a configuration file, the schematic values change and this action cannot be undone.

The Load Component Values feature does the following:

- Load component property values from a text file
- Change each component in the design according to the values in text file
- Produces a report In the SIMetrix/SIMPLIS command shell, detailing the properties that were changed.

### 6.10.2 Menus

There are six menu items under the main menu **Tools | Load Component Values**. These are:

<b>Overview</b>	Opens this help topic
<b>Create Config File from Schematic...</b>	Creates a component-value configuration file from the currently selected schematic
<b>From Default File</b>	Loads component values from the default text file. The default text filename is <schematic_name>.compvalues.txt
<b>Select a File to Load...</b>	Prompts for a configuration file to load.
<b>Edit Configuration File...</b>	Prompts for a configuration file to edit.
<b>View Log File...</b>	Views values in last component-value log file.

### 6.10.3 Configuration File Format

The configuration file is a plain ASCII text file with two columns separated by a tab character. Following are the rules for the configuration file:

- Blank lines and comment lines (starting with \*) are ignored.
- Each non-blank, non-comment line is assumed to be a load-component-value directive.
- Load-component-value directives are in the following form:

ADDRESS	VALUE
L1.DCR	5m

- The ADDRESS information should point to an explicit designator.property combination. For example, a resistance value is specified as R1.VALUE, not merely R1. Note: The correct ADDRESS for any given parameter depends heavily on the type of symbol that is being used, not merely the type of component. The program supports many standard symbols, but custom or seldom-used symbols may require some tweaking in order to get the desired behavior. ADDRESS and VALUE must be separated by tabs.
- The VALUE should appear exactly as you want it to be set in the component. For values containing white-space characters, enclose the value in double quotes ("").

### 6.10.4 Creating a Configuration File from the Schematic

To create a configuration file from the schematic, follow these steps:

1. Select menu **Tools | Load Component Values | Create Config File from Schematic...** The system reads all schematic symbols on the currently selected schematic and creates a component-value configuration file.
2. At the file selection dialog, type a filename and select a location to save the configuration file. The configuration file contains symbol properties only for component values that can be successfully loaded. Property values with spaces are automatically double quoted per the required configuration file format. Property values that are protected or that contain the double quote character are excluded from the generated configuration file.
3. Edit the configuration file using the menu **Tools | Load Component Values | Edit Configuration File...** to remove any load component value directives which you don't need.

### 6.10.5 Loading a Configuration File

**CAUTION:** Be sure to backup your schematic before loading a set of component values. When you load a configuration file, the schematic values change and this action cannot be undone.

An example including the schematic and a configuration file can be downloaded here : [simplis\\_003\\_load\\_comp\\_values\\_MP8708.zip](#).

To load component values for this design, follow these steps:

1. Unzip the archive to a location on your computer.
2. To open the schematic, double click the .sxsch file or drag that file into the SIMetrix/SIMPLIS Command Shell.
3. Execute one of the menu options from the Load Component Values menu:
  - **Tools | Load Component Values | From Default File**
  - **Tools | Load Component Values | Select a File to Load...**

The schematic reflects the new values, and the following report appears in the command shell window.

Located default component value input file  
(<path>/simplis\_003\_load\_comp\_values\_MP8708.compvalues.txt).

Action:	refdes	old	new
_____	_____	_____	_____

No Change	Vin.VC	12	12
No Change	Vin.ESR	1m	1m
Change	C5.CC	22u	4.7u
No Change	C6.CC	1	1u
No Change	C10.CC	100n	100n
No Change	C11.CC	1u	1u
Change	L1.LI	1.8u	4.2u
No Change	C7.CC	22u	22u
Change	C7.QUANTITY	2x	2
No Change	C9.CC	1p	1p
Change	R1.VALUE	40.2k	59k
Change	R2.VALUE	12.7k	19.1k

Load component value – complete @ XX:XX:XX

You may also view the log file with the **Tools | Load Component Values | View Log File...** menu.

### 6.10.6 Automating Load Component Values with Command line Switches

A special switch /l ( the letter l, not the numeral 1) has been added to SIMetrix/SIMPLIS to automatically open a schematic and load a set of component values when the program starts. The syntax for the command line switch is:

```
<full_path_to_simetrix.exe> <full_path_to_schematic_to_open> /l <full_path_to_comp_values_file>
```

Note each of the three command line arguments are full paths.

### 6.10.7 Automating Load Component Values with a Script

The load component values feature is provided by the script load\_comp\_values.sxscr. The script has one optional argument – a filename. Without the optional argument, the script attempts to load the default file, which is <schematic\_name>.compvalues.txt. If that file cannot be found or cannot be opened, a file selection dialog opens to allow you to select a configuration file.

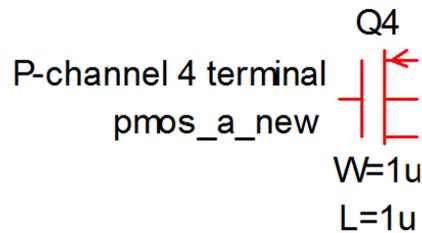
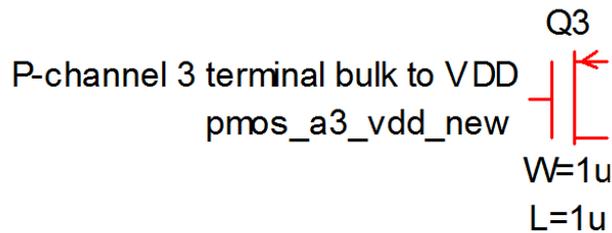
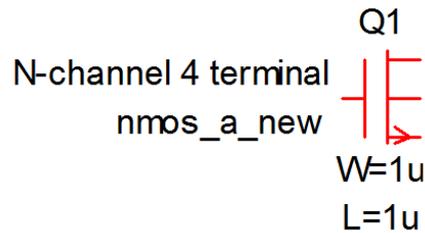
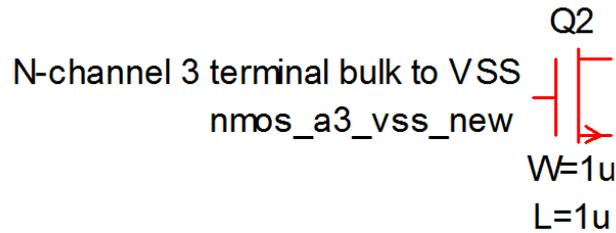
- To load the default file, type load\_comp\_values at the SIMetrix/SIMPLIS command line.
- To load a different filename, type load\_comp\_values '<config\_filename>' at the SIMetrix/SIMPLIS command line. Note the filename is enclosed in single quotes.

The load component values script can be called from any SIMetrix Script. The scripting language is available only in the full version of SIMetrix/SIMPLIS, although the script feature can be temporarily unlocked in SIMetrix/SIMPLIS Elements by using the Unlock feature.

## 6.11 Using Schematic Editor for CMOS IC Design

### 6.11.1 MOSFET Symbols

Four MOSFET symbols are supplied for use in CMOS IC design. These are:



These symbols have the model name N1 for the N-channel types and P1 for the P-channel types. These names can of course be changed after being placed on the schematic, but this would be time consuming to have to do each time. To avoid having to do this, you can do one of the following:

- Modify your SPICE model files so the devices are always called N1 and P1
- Modify the standard symbol so the model name corresponds to your SPICE models.
- Create a new set of symbols for each process you use.

The best course of action is probably to create a new symbol for each process. Once you have created the new symbols, you can modify the toolbar buttons so that they call up your new symbols instead of the standard ones. You must do this using the DefButton command which redefines toolbar buttons. To make permanent changes the DefButton command should be put in the startup script. Here is the procedure:

1. Select menu **File | Options | Edit Startup Script**. This will open the startup script in a text editor window. The startup script is automatically run each time the program starts.
2. Enter a DefButton command for each toolbar button you wish to redefine. For the MOS symbols the

commands will be one or more combinations of the following:

```
DefButton NMOS4 "inst /ne your_nmos4_symbol"
DefButton PMOS4 "inst /ne your_pmos4_symbol"
DefButton NMOS3IC "inst /ne your_nmos3_symbol"
DefButton PMOS3IC "inst /ne your_pmos3_symbol"
```

Replace your\_nmos4\_symbol, your\_pmos4\_symbol etc. with the internal names of your new symbols.

### 6.11.2 Edit Length and Width

A popup menu **Edit MOS Length/Width...** will edit L and W properties. The menu is initially hidden but can be enabled by setting the option variable EnableLWEditMenu. You can do this by typing this command at the command line:

```
Set EnableLWEditMenu
```

You will need to restart SIMetrix to bring this into effect.

### 6.11.3 Automatic Area and Perimeter Calculation

The optional popup menu **Edit MOS Length/Width...** (See [“Edit Length and Width” on page 104](#)) edit device length, width and scale factor. These menus alter the symbol properties W, L and M for width, length and multiplier.

The symbols described above have been designed in a manner such that additional parameters such as AS, AD, PS, PD, NRS and NRD may be automatically calculated from width and length. To use this facility append the VALUE property with the parameter definitions defined as expressions. E.g.:

```
N1 AD={2*%W%+0.8u}
```

The above device will have an AD parameter calculated from “2\*width+0.8u”. Note that the formula is enclosed in curly braces (‘{’, ‘}’) and width and length expressed as %W% and %L% respectively. You can use similar expressions for any other parameter.

As an alternative, you can define AS, AD etc. as a parameter expression in a sub-circuit. See [“Subcircuits” on page 181](#) for more details.

### 6.11.4 Editing the MOS Symbols

You may wish to create your own MOS symbols for each process you use. We suggest that you always make a copy of the standard symbols and save them with a new name in your own symbol library.

Once you have your copied version, you can edit it to suit your IC process. In most applications, you will probably only need to edit the VALUE property. See next paragraph.

#### Editing VALUE property

The VALUE property defines the model name and all the device’s parameters except length, width and the multiplier M. The standard VALUE property defines just the model name and this defaults to N1 for NMOS devices and P1 for PMOS devices. You should edit these to match the model name used in your process.

In addition (as described in [“Automatic Area and Perimeter Calculation” on page 104](#)) you can append the VALUE property with other parameters such as AD, AS etc. and define these as expressions relating width (using %W%) or/and length (using %L%).

## To Edit the Default Values of L and W

Edit the L and W properties as appropriate.

## To Edit the Hidden Node for 3 Terminal Devices

The hidden bulk node for three terminal devices is defined by the BULKNODE property. This defaults to VSS for NMOS devices and VDD for PMOS devices.

## 6.12 Further Information

### 6.12.1 How Symbols are Stored

When a symbol is placed on a schematic, a copy of that symbol definition is stored locally. This makes it possible to open the schematic even if some of the symbols it uses are not available in the symbol library. However, if you edit a symbol definition for a schematic that is saved, when you open that schematic, it has a choice between its local copy of the symbol or the copy in the library. Which it chooses depends on an option chosen when the symbol is saved. When saving the symbol with the graphical editor, you will see the check box **All references to symbols automatically updated**. If this is checked then the schematic editor will always use the library symbol if present. If not, it will use its local copy.

If a schematic is using a local copy and you wish to update it to the current library version, select the symbol or symbols then select the popup menu **Update Symbol**. Note that *all* instances of the symbol will be updated. It is not possible to have two versions of a symbol on the same schematic.

### Important Note

Note, that only the symbol geometry, pin definitions and protected properties of a schematic instance will be changed when its symbol definition is edited.

Unprotected properties will remain as they are. For example, the standard NPN bipolar transistor symbol has an initial VALUE property of Q2N2222 so when you place one of these on the schematic from the **Place** menu or tool bar, this is the value first displayed. This can of course be subsequently changed. The initial value of Q2N2222 is defined in the NPN symbol. However, if you edit the symbol definition and change the initial value to something else - say - BC547, the value of the VALUE property for any instances of that symbol that are *already placed* will *not* change.

You can use the popup menu **Restore Properties...** to restore properties to their symbol defined values. For more information, see [“Restoring Properties” on page 89](#).

If you wish a property value to always follow the definition in the symbol, then you must protect it. See [“Defining Properties” on page 114](#) for details.

### 6.12.2 Summary of Simulator Devices

The following information is needed to define schematic symbols for the various devices supported by the simulator.

In order to be able to cross-probe pin currents, the pin names for the schematic symbol must match up with those used by the simulator. So for a BJT (bipolar junction transistor) the simulator refers to the four pins as 'b', 'c', 'e' and 's' for base, collector, emitter and substrate. The same letters must also be used for the pin names for any schematic BJT symbol. The simulator device pin names are listed below.

The MODEL property is the schematic symbol property which describes what type of device the symbol refers to. SPICE uses the first letter of the part reference to identify the type of device. The SIMetrix netlist generator prefixes the MODEL property (and a '\$' symbol) to the part reference to comply with this. This makes it possible to use any part reference on the schematic.

Device	Model property	Pin no.	Pin names	Pin function
XSPICE device	A			
Arbitrary Sources	B	1	p	
		2	n	
Bipolar junction transistors	Q	1	c	Collector
		2	b	Base
		3	e	Emitter
		4	s	Substrate
Capacitor	C	1	p	
		2	n	
Current Controlled Current Source (2 terminal)	F	1	p	
		2	n	
Current Controlled Current Source (4 terminal)	F	1	p	+ output
		2	n	- output
		3	any	+ control
		4	any	- control
Current Controlled Voltage Source (2 terminal)	H	1	p	
		2	n	
Current Controlled Voltage Source (4 terminal)	H	1	p	+ output
		2	n	- output
		3	any	+ control
		4	any	- control
Current Source	I	1	p	+
		2	n	-

Device	Model property	Pin no.	Pin names	Pin function
Diode	D	1	p	Anode
		2	n	Cathode
GaAs FETs	Z	1	d	Drain
		2	g	Gate
		3	s	Source
Inductor	L	1	p	
		2	n	
Junction FET	J	1	d	Drain
		2	g	Gate
		3	s	Source
MOSFET	M	1	d	Drain
		2	g	Gate
		3	s	Source
		4	s	Bulk
Resistors	R	1	p	
Transmission Line	T (lossless)	1	p1	Port 1 Term 1
		2	n1	Port 1 Term 2
	O (lossy)	3	p2	Port 2 Term 1
		4	n2	Port 2 Term 2
Voltage Controlled Current Source	G	1	p	+ output
		2	n	- output
		3	cp	+ control
		4	cn	- control
Voltage Controlled Switch	S	1	p	Switch term 1
		2	n	Switch term 2
		3	cp	+ control
		4	cn	- control
Voltage Controlled Voltage Source	E	1	p	+ output
		2	n	- output
		3	cp	+ control
		4	cn	- control
Voltage Source	V	1	p	+ output
		2	n	- output

<b>Device</b>	<b>Model property</b>	<b>Pin no.</b>	<b>Pin names</b>	<b>Pin function</b>
Subcircuits	X			Pins can be given any name. Numbering must be in the order that pins appear in the .subckt control which defines the subcircuit. SIMetrix uses a special extension of the netlist format to tell the simulator what the pin names are.
Verilog-A device	U (recommended)			Pin count, names and order must match ports in Verilog module statement. See Verilog-A User Manual for details.
VSXA (Verilog-HDL device)	U			Pin count, names and order must match ports in Verilog module statement. See VSXA device in the Simulator Reference Manual.
AC Table Lookup	U			Pin count = 2 x number of ports. This device does not currently provide current readback. Pin names can thus be assigned arbitrarily.

# Chapter 7

## Symbol Editor and Library

### 7.1 Creating Schematic Symbols - Overview

A large variety of schematic symbols are supplied with SIMetrix which should cover many uses. However, there will be occasions when you wish either to define your own new symbol - perhaps to implement a hierarchical block or subcircuit - or to modify one of the standard symbols. This chapter describes how this can be done.

Schematic symbols are organised in a library handled by the library manager. For details refer to [“Symbol Library Manager” on page 119](#)

There are two different methods to create symbols:

1. Use the graphical symbol editor (see [“Graphical Symbol Editor” on page 109](#)).
2. Create manually with a script

For most applications, using the graphical symbol editor is the most appropriate method.

Creating a symbol from a script is appropriate for automated symbol creation. Details are provided in *Script Reference Manual/Applications/Schematic Symbol Script Definition*.

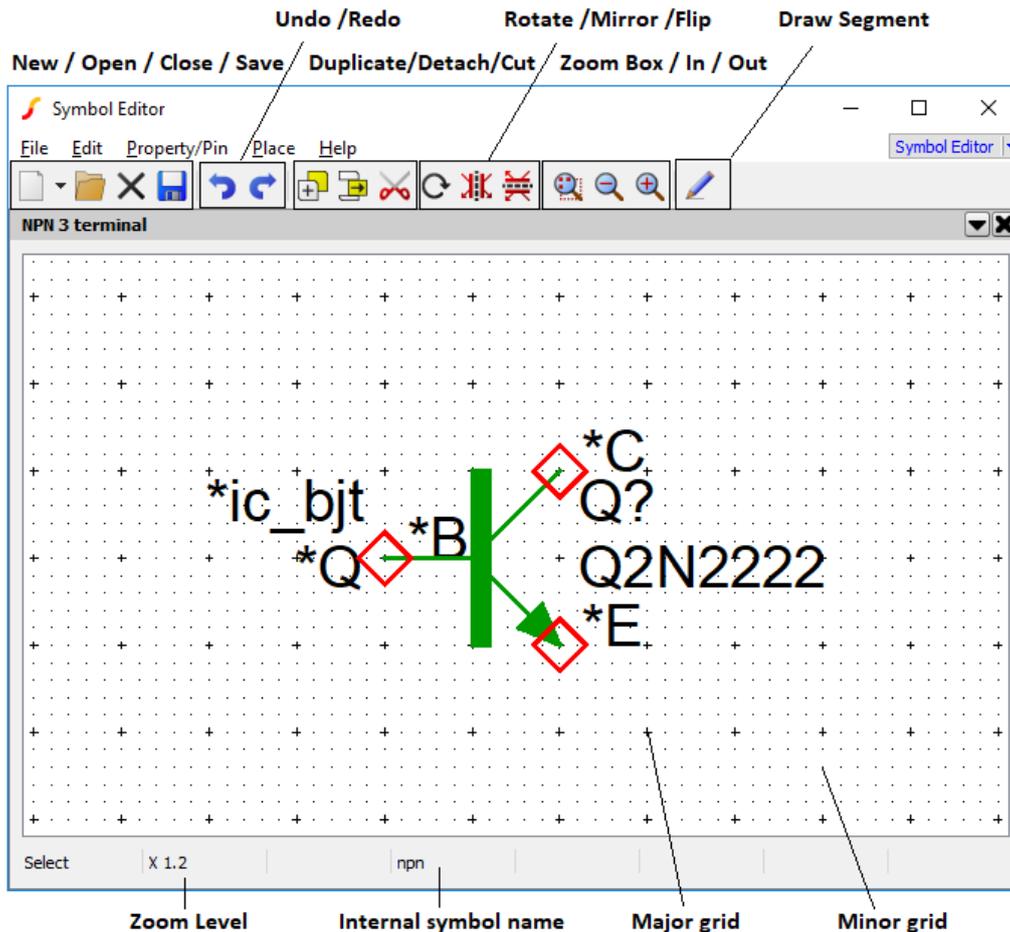
### 7.2 Graphical Symbol Editor

#### 7.2.1 Notes

The graphical symbol editor shares much in its operation and layout with the schematic editor. For this reason, it is recommended that, before learning how to use the symbol editor, you become competent in the operation of the schematic editor. In some parts of the following sections the explanations assume that you are already familiar with the schematic editor.

#### 7.2.2 Symbol Editor Window

The following diagram shows the main elements of the symbol editor



### 7.2.3 The Elements of a Symbol

Schematic symbols are composed of a combination of the following elements. All symbols that represent an electrical device would comprise *all* of these elements. Symbols used purely for annotation would not need pins and may not need one or other of the remaining elements either. The schematic caption, for example, is a symbol that consists purely of properties.

- *Segments*. These make up the visible body of the symbol. They include straight line segments and *arc segments*.
- *Pins*. These define electrical connections to the device.
- *Properties*. Properties have a name and a value and are used to define the behaviour of the device represented by the symbol. They can also be used for annotation, for example, a label or a caption.

### 7.2.4 Creating a New Symbol

Select the menu **File | New | Symbol**. This will open a symbol editor window as shown above. Now create the elements of the symbol as described above. Details are provided in the following sections.

### 7.2.5 Editing an Existing Symbol

Select the menu **File | Symbol Manager...** and select the symbol you wish to edit. See “[Symbol Library Manager](#)” on page 119 for details.

If you wish to edit a symbol that is placed on an open schematic, select the symbol on the schematic then choose popup menu **Edit Symbol....**

### 7.2.6 Drawing Straight Line Segments

Drawing straight line segments in the symbol editor is very similar to drawing wires in the schematic editor. You can do one of the following:

1. Select Draw Segment Mode by clicking the **Draw segment tool** button. You can now draw segments using the left and right mouse buttons. Press the button again to revert to normal mode.
2. If you have a three button mouse, the middle button will start a new segment. The left button will complete a segment and terminate the operation, while the right button will terminate without completing the current segment.
3. Enter Draw Segment Mode temporarily by pressing **F3**.
4. Double click the left button to start a new segment.

### 7.2.7 Drawing Arcs, Circles and Ellipses

The basic method of drawing each of the curved elements is the same for each case. Before drawing starts, you must define the start-finish angle and, for ellipses, the ratio of height to width. The drawing operation itself defines the start and finish points. For full circles and ellipses the start and finish points are on opposite sides.

Dedicated menus are supplied for starting a full circle, half circle and quarter circle. For everything else use **Arcs | Ellipse/Arc...** .

When you have initiated the operation, the cursor will change to a shape showing a pencil with a small circle. You can now draw the curved segment by *dragging* the mouse with the left key. When you release the mouse button the operation will be complete and the mouse mode will revert to normal select mode.

It is easier to demonstrate than explain. You may wish to experiment with arc/circle/ellipse drawing to gain a feel of how the system operates.

You will note that full circles are displayed with a small filled square on opposite sides. These are the select points. You can pick either one and drag it to resize the circle.

### 7.2.8 Placing and Defining Pins

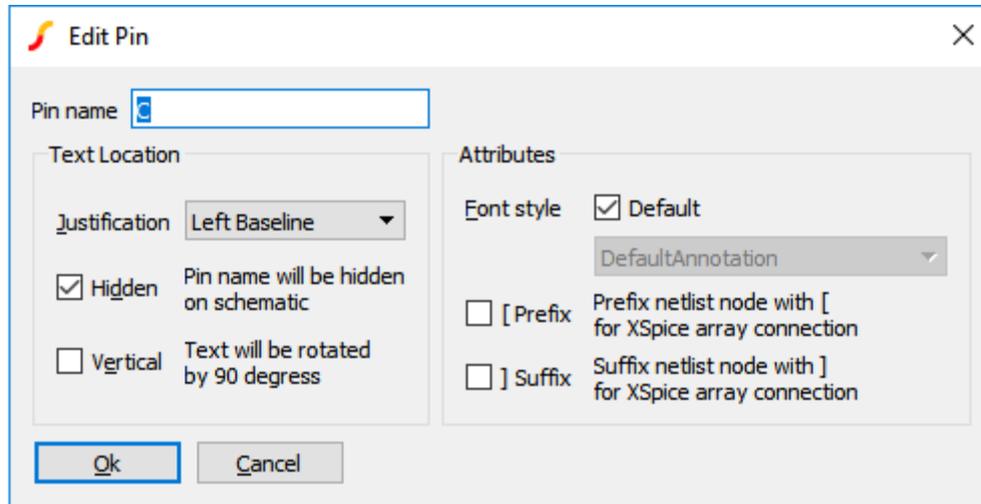
#### Placing a Single Pin

To place a single pin, select **Property/Pin | Place Pin...** . Place this on the sheet by left clicking the mouse at your desired location. Note that pins always snap to major grid points. (See diagram in section “[Symbol Editor Window](#)” on page 109).

The first pin you place on the sheet will be called ‘\*pin’. The \* signifies that the pin name will not be visible when the symbol is placed on a schematic.

## Editing Pin Attributes

To edit the attributes of the pin, (e.g. to change its name or visibility) select either the pin or its label with the left mouse key then press **F7** or select popup menu **Edit Property/Pin/Arc...** . This will display the following dialog:



### Pin name

Must be unique within the symbol and may not contain spaces.

If the symbol is to be used as a hierarchical block, the pin name must match the names of the module ports on the schematic which it represents.

If the symbol is to be used for an existing sub-circuit from, for example, a model library, the pin names are not important and you can choose any suitable name. The pin names *do not* need to match the node names in the sub-circuit definition.

### Text Location

Justification	If the pin name is visible this specifies its alignment.
Hidden	Check this box if you do not wish the pin name to be visible on the schematic.
Vertical	The pin's label will be displayed vertically if this is checked.

### Attributes

Font style	Select font style to use for a visible pin name. There is a choice of 8 styles. Schematic fonts are explained in “ <a href="#">Fonts</a> ” on page 472.
[ Prefix	Instruct netlist generator to prefix node with a '['. This is used to mark the start of a vector connection of an XSpice device

] Suffix	Instruct netlist generator to suffix node with a ']'. This is used to mark the end of a vector connection of an XSpice device
----------	---

## Placing Multiple Pins

To place more than one pin select menu **Property/Pin | Place Pin (repeated)...** You will be prompted to supply a *Base* pin name which will be used to compose the actual pin name. SIMetrix will append a number to this name to make the pin name unique. The first number used will be '0' unless you append the Base name with a number in which case your appended number will be used as the starting point. For example, if you supplied a Base name of DATA, the first pin placed will be called DATA0, the second DATA1 etc. assuming there aren't already pins of that name on the sheet. If you supplied a base name of DATA2, the first pin you place will be called DATA2, the second DATA3 etc.

## Editing Multiple Pins

You can only edit the names of pins one at a time, but you can edit the attributes of a group of pins in a single operation. First select all of the pins you wish to edit. Selecting is done in the same manner as for the schematic except note that you can select the pins themselves or the pin names; either will do. Now press **F7** or select popup menu **Edit Property/Pin/Arc...** You can change any of the pins attributes except its name and the change will be applied to all selected pins.

## Moving Pins or Pin Names

Moving any item in the symbol editor is done the same way as in the schematic. Note, however that pin names are attached to the pins. If you move a pin, its name moves with it. You can move the name on its own by making sure that only the name is selected and not the pin.

## Defining Pin Order

The symbol's pin order is important if you are defining a symbol for use with a sub-circuit or primitive simulator device. If it is a sub-circuit, the symbol's pin order must match the order in which the corresponding nodes are defined in the .SUBCKT statement. If the symbol is a primitive device, then it must follow the order defined in section "[Summary of Simulator Devices](#)" on page 105.

If you are creating a symbol for a hierarchical block, you do not need to define the pin order. The connection between the symbol and the underlying child schematic is made by name.

To define the symbol's pin order select menu **Property/Pin | Edit Pin Order...** Use the up and down buttons to reorder the pins as appropriate.

## Adding XSpice Pin Attributes

Some XSPICE devices support vector connections and/or variable type connections. These are designated in the netlist with the characters '[', ']' and '%' and are explained in *Simulator Reference Manual/Simulator Devices/Using XSPICE Devices*. You can add these to a symbol by prefixing the appropriate pin name with the same characters as required in the netlist. E.g. to start a vector connection at a pin named IN1 enter the pin name [IN1. To close a vector connection at pin IN3 use pin name IN3].

Similarly to change a connection whose default type is 'v' (i.e. a single-ended voltage) to a differential current (type%id), prefix the first pin name with%id and a space. E.g. pin name 'VIN' would become '%id VIN'.

Examples of the use of vector connections in symbols can be found with any of the digital gate symbols.

## 7.2.9 Defining Properties

Properties define the behaviour of the symbol. For full documentation on the use of properties, see section “[Properties](#)” on page 87. In this section, the methods of adding and editing properties in the symbol editor are described.

### Adding a Single Property

To add a property to a symbol, select **Property/Pin | Add Property...** You will see the following dialog box:

The dialog box is titled "Add Property" and contains the following fields and options:

- Name:** A text input field.
- Value:** A large text area for entering the property value.
- Text Location:**
  - Auto**
  - Absolute**
  - Normal:** Left (dropdown)
  - Rotated:** Left (dropdown)
  - Justification:** Left Top (dropdown)
  - Hidden** (Property text will be hidden on schematic)
  - Vertical** (Text will be rotated by 90 degrees)
- Property Attributes:**
  - Font style:** Default (checkbox checked)
  - DefaultAnnotation:** DefaultAnnotation (dropdown)
  - Selectable** (Allows symbol property to be selected by clicking on property text)
  - Protected** (Inhibits editing of property values on schematic)
  - Show name** (Display name as well as value)
  - Resolve symbolic** (Enable resolution of expression and special symbols)
- Buttons:** Ok, Cancel, Help

This box allows you to define the name, value and attributes of the property. Note that if the property is not protected, the value and attributes can be changed after the symbol has been placed on a schematic using the schematic popup **Edit Properties...**

### Name

Name of property. This would usually be one of the special properties documented in “[Properties](#)” on page 87. You can, however, add any property name you wish to display as text or to provide a special function that you define in a custom script. The only restriction is that the name must not contain spaces.

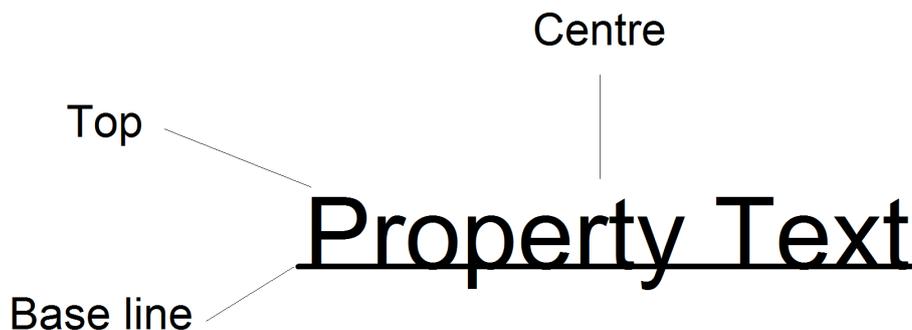
## Value

The property's value. (Don't confuse this with the VALUE property). You can insert a new line by pressing the **ENTER** key. But be careful that if you press the ENTER key accidentally intending to close the dialog that you must delete the erroneously entered new line.

## Text location

Define the position of the property's value text on the schematic.

Auto/Absolute	When auto is selected, the property's value text is positioned automatically outside the symbol's border according to the options specified in Normal and Rotated. When absolute is selected, the property is placed at a fixed position relative to the symbol body. You can define the location interactively with the mouse. When auto is selected, the text is always horizontal, when absolute is selected, the text is vertical when the symbol is at a 90 degree rotation.
Normal	When auto is selected, this specifies which side of the symbol the text is located when the symbol is in normal orientation.
Rotated	When auto is selected, this specifies which side of the symbol the text is located when the symbol is rotated 90 degrees.
Justification	Defines the reference point on the text when absolute is specified. See diagram below for meaning of options. The reference point is always at a fixed location with respect to the symbol body. The position of the remainder of the text may vary with zoom level or font size.
Hidden	The property's value text will not be displayed in the schematic.
Vertical	If checked, the property will be displayed vertically. This option is only available if absolute location is selected.



## Property Attributes

Font style	Select one of eight font styles. The actual font definition is defined by the Font dialog box. See <a href="#">“Fonts” on page 472</a> for details.
Selectable	If checked, the instance of the symbol owning the property can be selected by clicking in the property text. It is recommended that this option is off unless the symbol has no body e.g. a pure text item.
Protected	If checked, it will not be possible to edit or delete the property on a schematic instance of the symbol.

Linear	When this is <i>not</i> checked, the size of the font is adjusted for best readability and does not necessarily scale exactly with the zoom magnification. When the box is checked, the font size follows the magnification in a linear fashion.
Show	Name If selected the name of the property as well as its value will be displayed.
Resolve symbolic	<p>The value may contain expressions enclosed by ‘{’ and ‘}’, keywords enclosed by ‘&lt;’ and ‘&gt;’ and property names enclosed by ‘%’. These items will each be substituted with their resolved value to obtain the property text that is actually displayed.</p> <p>Expressions may contain the usual arithmetic operators and may also use functions as defined in <i>Script Reference Manual/Function Reference</i>. Property names enclosed with ‘%’ are substituted with that property’s value. Keywords may be &lt;date&gt;, &lt;time&gt;, &lt;version&gt;, &lt;if&gt;, &lt;ifd&gt; and &lt;t&gt;. &lt;date&gt;, &lt;time&gt; resolve to date and time in local format and &lt;version&gt; resolves to an integer value which is incremented each time a schematic is saved. The keywords &lt;if&gt;, &lt;ifd&gt; and &lt;t&gt; behave in the same manner as the TEMPLATE property keywords of the same name. See <a href="#">“Template Property” on page 89</a> for details.</p> <p>Note that, like template properties, the resolution is performed in two passes with the property values being substituted first.</p>

## Adding Standard Properties

Select menu **Property/Pin | Add Standard Properties....** This prompts you for values for the *ref*, *value* and *model* properties. These properties are usually specified for all symbols, with the exception of hierarchical blocks which do not require a VALUE property. If you are using the SIMetrix/SIMPLIS product, you will also be prompted to supply a value for the SIMULATOR property.

The following table describes the four standard properties.

Property name	Function
REF property	This is the part reference e.g. U1, R3 etc. This would conventionally be a letter or letters followed by a question mark (?). When you place the part on the schematic, the question mark will be replaced automatically by a number that makes the reference unique. If you don’t specify a MODEL property (see below), the first letter of the reference is important as it defines the type of device. This is explained in more detail below.
VALUE property	This is the device’s value or model name including any additional parameters. For a resistor this might be 12k for an op-amp, LM324, or for a bipolar transistor, Q2N2222. The VALUE property must be present on a symbol at the definition stage but its initial value is not important as it would usually be changed after the symbol is placed on a schematic.

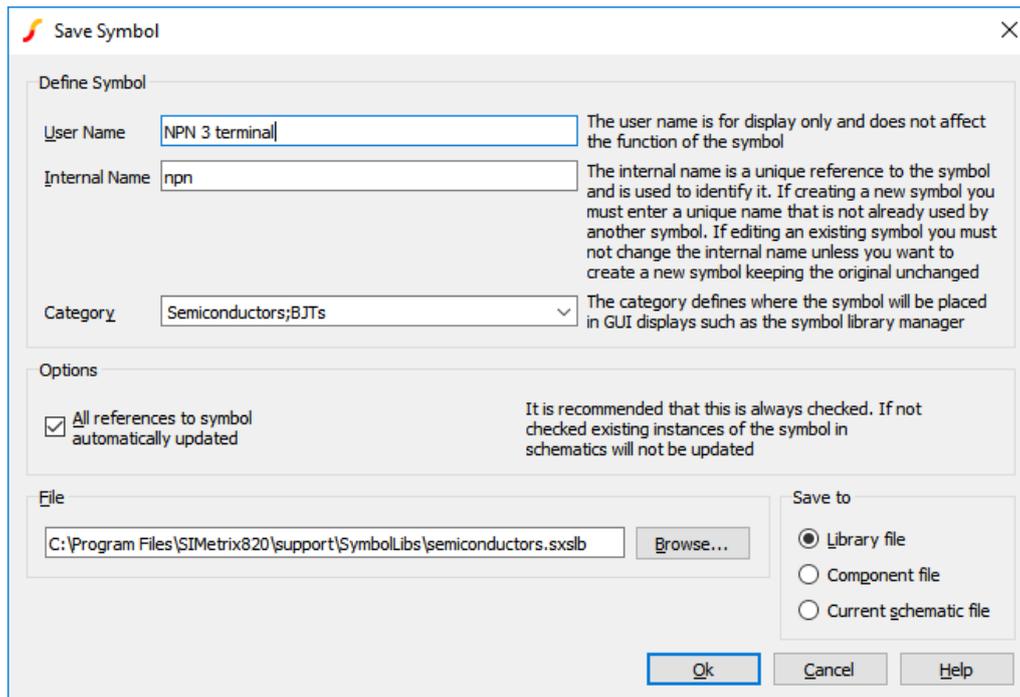
Property name	Function						
MODEL property	<p>This property usually has a value that is a single letter that specifies the type of device. For sub-circuits and hierarchical blocks this letter must be 'X'. For other types of device refer to the table in <a href="#">“Summary of Simulator Devices” on page 105</a>. If this property is not present, the first letter of the REF property will be used to identify the device.</p> <p>For information only: The value of this property and a '\$' symbol are prefixed to the <i>ref</i> property to obtain the first word of the device line in the netlist hence complying with SPICE syntax. (This won't be done if the first letter of the REF property is already the same as the value of the MODEL property as this would be unnecessary.)</p>						
SIMULATOR property	<p>This is only required for the SIMetrix/SIMPLIS product. It declares which simulator the symbol is compatible with. This is only for the purpose of advising the user if a part may not work with a particular simulator. It does not affect the functionality of that part for any simulator.</p> <p>This property can have one of three values:</p> <table> <tbody> <tr> <td>SIMetrix</td> <td>Symbol compatible with SIMetrix simulator only.</td> </tr> <tr> <td>SIMPLIS</td> <td>Symbol compatible with the SIMPLIS simulator only.</td> </tr> <tr> <td>Dual</td> <td>Symbol compatible with both SIMetrix and SIMPLIS</td> </tr> </tbody> </table>	SIMetrix	Symbol compatible with SIMetrix simulator only.	SIMPLIS	Symbol compatible with the SIMPLIS simulator only.	Dual	Symbol compatible with both SIMetrix and SIMPLIS
SIMetrix	Symbol compatible with SIMetrix simulator only.						
SIMPLIS	Symbol compatible with the SIMPLIS simulator only.						
Dual	Symbol compatible with both SIMetrix and SIMPLIS						

## Editing a Property

To edit a property, select it then press **F7** or select popup menu **Edit Property/Pin/Arc....** This will open a dialog box very similar to the one described above but without the option to enter a property name. Make the appropriate changes then Click **Ok**.

### 7.2.10 Saving Symbols

To save the current symbol, select menu **File | Save....** The following dialog will be displayed:



### Define Symbol - User Name

Enter the name as you wish it to be displayed in the dialog box opened with the schematic menu **Parts | All Symbols....**

### Define Symbol - Internal Name

For a new symbol, an internal name will automatically be entered when you type the User Name. In most cases you can leave it at that. However, the internal name must be unique across the whole model library so there may be situations where you will need to change it. If you are unsure whether the name used is unique, prefix it with something that is very unlikely to be used anywhere such as your initials. The resulting name does not need to be meaningful to anyone else; it is an identifying code not a descriptive name.

### Define Symbol - Category

Enter a category to determine how the symbol will be listed in the dialog box opened with the schematic menu **Place | From Symbol Library....** Sub-categories are separated using a semi-colon. Note that you can easily move symbols to different categories using the symbol library manager. So if you are unsure at this stage what category to use, you can place it in a temporary category and move it later.

### Define Symbol - All references to symbol automatically updated

If this is checked, any changes you make to a symbol will automatically be applied to any instance of it in existing schematics whether they are open or not.

If it is not checked, instances of the symbol will not be updated until the **Update Symbols** menu is selected in the schematic. Copies of all symbols used by a schematic are stored locally within the schematic and that local version will only be updated if this box is checked.

See “[How Symbols are Stored](#)” on page 105 for further details.

#### Save to - Library File

Saves the symbol to the library file specified in the File box. This would usually have a .SXSLB extension.

#### Save to - Component File

Saves the symbol as a component to the file specified in the File box. This would usually have the extension .SXCMP. Component files are used for hierarchical schematics and contain a schematic and a symbol representing it in the same file. When you save a symbol to a component file, only the symbol portion of it will be over-written. If it contains an embedded schematic, that schematic will remain unchanged. See Hierarchical Schematic Entry .

#### Save to - Current Schematic Only

The symbol will be saved to the currently selected schematic only and will not be available to other schematics.

#### File

Library or component file name - see above. Press **Browse** to select a new file.

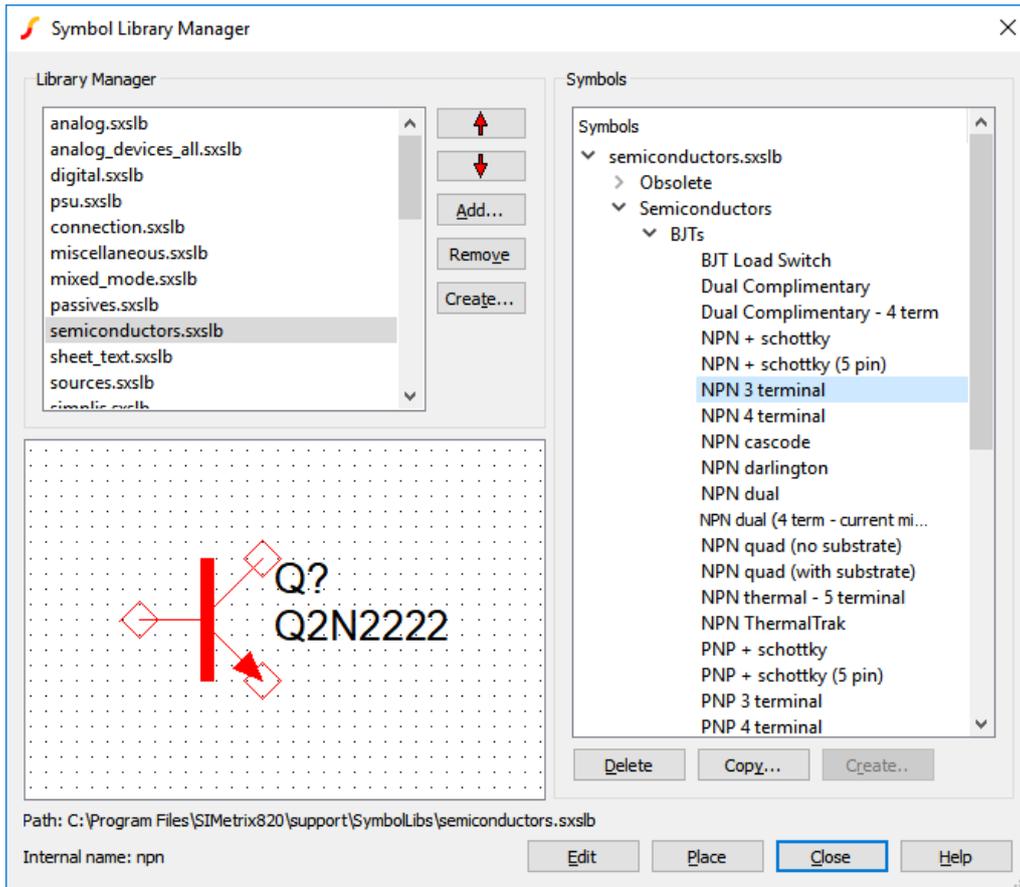
## 7.3 Creating a Symbol from a Script

Creating a symbol from a script is a method of creating symbols programmatically and is useful for creating symbols for devices that have some variable characteristic. An example is a transformer that can have a varying number of windings and indeed this technique is employed to create symbols for the built-in transformer models.

For full documentation on how to create a symbol from a script, refer to the *Script Reference Manual/Applications/Schematic Symbol Script Definition*.

## 7.4 Symbol Library Manager

The symbol manager is a comprehensive system for managing symbols and the libraries that store them. To open the symbol library manager, select menu **File | Symbol Manager...** The dialog shown below will be displayed.



Note that the box can be resized in the usual way.

The symbols available to the schematic editor are stored in library files which conventionally have the extension .SXSLB . Only symbols in installed libraries are available for placing a new part. Note, however, that once a symbol is placed on a schematic a copy is stored locally, so you can still view a schematic that uses symbols that are not installed.

The symbols in each file are grouped into categories using a tree structure as shown above in the Symbols box.

The manager allows you to install or uninstall library files, to move symbols between files or categories, to delete symbols, to copy symbols and to create new categories. You can also create new empty symbols ready for editing with the symbol editor.

## 7.4.1 Operations

### Installing Libraries

Select the Add... button and select a library file to install.

Clicking the Add... button will open a file select dialog box allowing you to add a new symbol library file.

## Uninstalling Libraries

Select the library file you wish to uninstall from the *Library Files* box then click the **Remove** button. Note that this does not delete the file.

## Changing Search Order

When searching for a particular symbol, the library files are searched in the order in which they are listed in the *Library Files* box. To change the order, use the **Up** and **Down** buttons.

## Moving Symbols

To move an individual symbol to a new category, simply pick it up with the mouse and drop it onto the new category. You can move a symbol to a new library by dropping the symbol onto a library file in the *Library Files* box.

You can move more than one symbol at a time by picking up a complete category.

## Copying Symbols

To copy a symbol within a library, select the symbol in the *Symbols* box then click the **Copy...** button or use the right click menu **Copy Symbol...**. Enter a new user name for the symbol. It isn't usually necessary to change the internal name.

To copy symbols to a new library, use the same drag and drop procedure as for moving but hold the **control** key down while doing so. You can do this for a single symbol or for an entire category. Note that when copying to a new library, the symbol retains its user name and internal name. There will therefore be duplicates installed unless they are renamed.

## Deleting Symbols

To delete a symbol, select it then press **Delete** or the right click pop-up menu of the same name. You can also delete an entire category in the same way.

## Renaming Symbols

Select a symbol then press F2 or the right click popup menu **Rename**. You can also rename a category in the same way.

Note that this only renames the user name of the symbol. There is no method of changing the internal name other than making a copy with a new name, then deleting the original.

## Creating a New Category

To create a new category, select the parent category where you wish it to be placed, then click **Create...** or the popup menu of the same name. In the dialog that opens, select the **Category** button and enter the new name.

## Creating a New Symbol

Select the category where you wish the symbol to be placed, then click **Create...** or the popup menu of the same name. Enter the desired user name. An internal name will be automatically entered as you type in the user name. This can usually be left alone.

The symbol created will be empty. Use the symbol editor to define it. You can call this directly by clicking the **Edit** button. Note that this will close the library manager dialog box.

## Placing Symbol

If a schematic sheet is open, you can place a symbol on it directly from the library manager by clicking the **Place** button. Note that this will close the dialog box.

### 7.4.2 Editing System Symbol Libraries

The system symbol libraries are listed in the file `SystemLib.sxlst` located in the symbol libraries folder. The libraries are treated specially when written to - e.g. when editing any symbol in the library.

System symbol libraries are protected from being edited directly. You can still edit the system symbols, but the changes are stored separately in an ASCII file located in a directory in the application data area. This scheme protects such changes from being lost when the system symbol libraries are updated during a service update.

The system symbol libraries are stored in a directory defined by the `SymbolsDir` option variable (see [“List of Options” on page 454](#)). This is typically located at `C:\Program Files\SIMetrixXX\support\symbollibs`. The directory where system library edits are stored is defined by the `UserSystemSymbolDir` option variable (see [“List of Options” on page 454](#)).

# Chapter 8

## Parts

In this chapter we describe the parts available at the schematic level. Broadly speaking parts fall into two categories namely numbered and generic. Numbered parts, also referred to as model library parts are devices that have a manufacturer's part number and are described by a model either supplied with SIMetrix or by the manufacturer itself. Generic parts are devices that are defined by one or more parameters that are entered by the user after the part has been placed on the schematic.

A transistor like a 2N2222 or BC547 is an example of a numbered part and a resistor is probably the simplest example of a generic part.

There are some parts that have characteristics of both types. CMOS IC designers would use MOSFETs defined by a model but will then customise it with length and width parameters. Saturable inductors have an underlying model to describe the core's characteristics but a number of user defined parameters to define the geometry and air gap.

Numbered parts need a model which is usually stored in the model library. Refer to [“Device Library and Parts Management” on page 209](#) for details.

This chapter is concerned only with devices at the schematic level. Many of these devices are implemented directly by the simulator. For example the simulator has a bipolar transistor model built in and such devices can be defined with a set of simulator parameters. However, not all devices are implemented directly by the simulator. It does not, for example have an operational amplifier device built in. These parts are constructed from a number of other parts into a subcircuit.

The devices built in to the simulator are described in the “Simulator Devices” chapter of the *Simulator Reference Manual*.

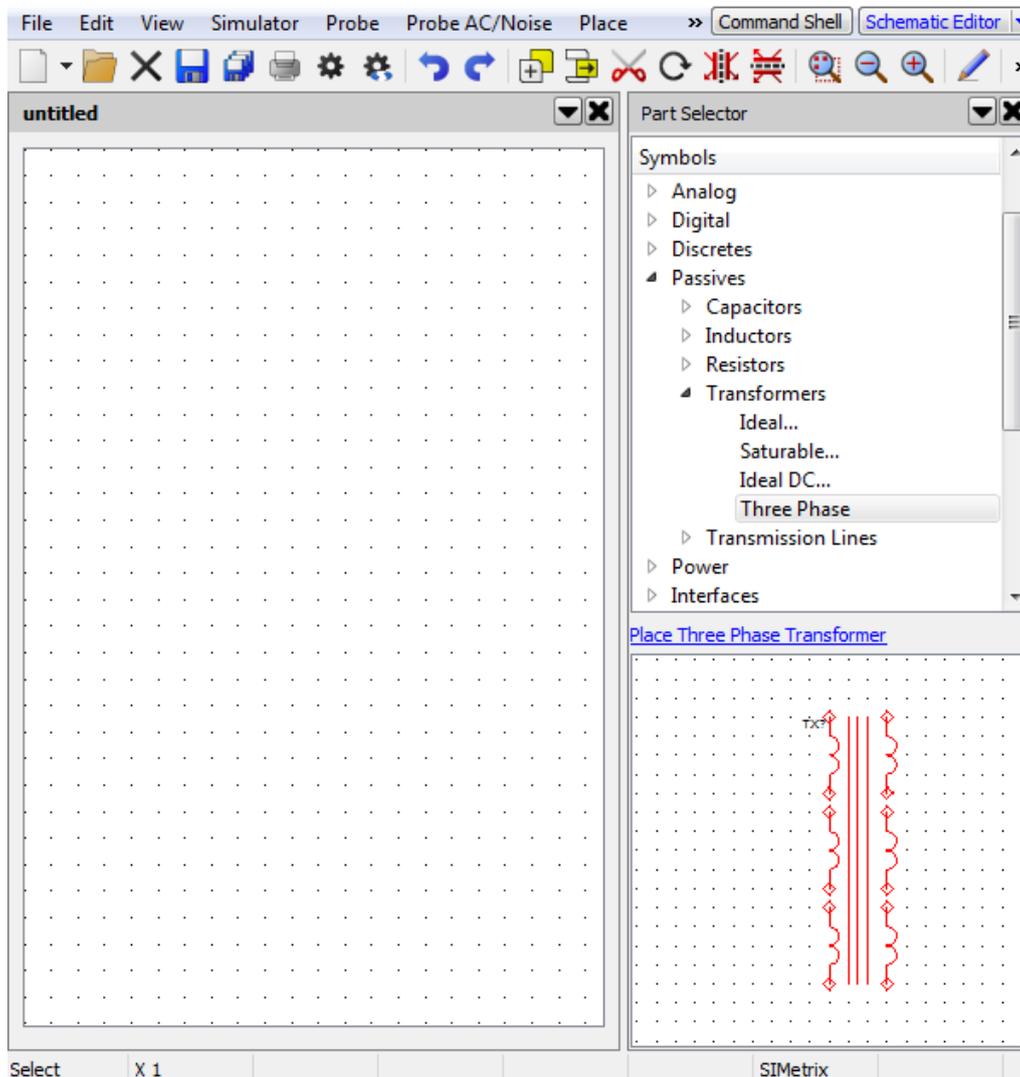
### 8.1 How to Find and Place Parts

SIMetrix has a number of different ways of finding and placing parts. Many parts can be placed in 2 or 3 different ways and which you choose will depend on personal preference. The methods are:

1. [“Part Selector” on page 124](#). The part selector is a hierarchical list of parts that may be displayed within the main application window.
2. [“Part Search” on page 125](#). Searches parts available in the part selector.
3. Place menu. This includes the [“Model Library Browser” on page 125](#) and the search by specification (see [“Selecting a Model by Specification” on page 126](#)) system. Other parts may also be placed from the place menu but many are easier to locate using the part selector.
4. Toolbar. Only a few parts available but quick and easy.

## 8.2 Part Selector

The part selector is hierarchical list of parts used with the schematic editor. As a *System View* the part selector can be placed anywhere around the outside of the window, it's default position is the right hand side of the window. See below:



The part selector is arranged in hierarchical manner to ease browsing for the part you wish to place.

To use the part selector, locate the part to place then either:

1. Right click and select a menu as appropriate. The vast majority of parts offer a single placement or a repeated placement. Parts from the model library also provide a **View Model** menu to preview the actual electrical model.

OR

2. Click on the hyperlink that shows below the part selector tree and above the symbol window. OR
3. Double click the part in the selector tree.

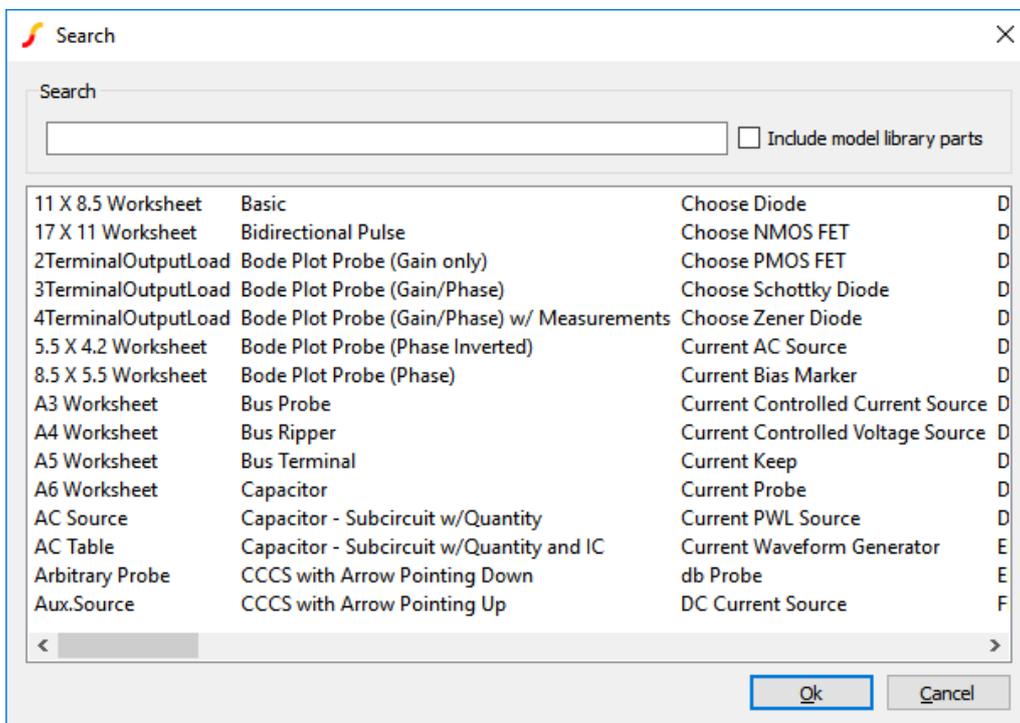
Virtually all available parts may be selected using the part selector. There are differences between SIMetrix and SIMPLIS modes in how parts located in the model library are placed; in SIMetrix mode the part appears directly in the part selector whereas in SIMPLIS mode, the part selector directs you to the model library browser. There are options to change this behaviour - see *PartSelShowSimplisModels* in “List of Options” on page 454.

The part selector may be customised in a number of ways; you can reorganise the hierarchical structure using a simple GUI and you can also add your own special parts or remove some of the standard items. For more information see “Parts Management - Configuring the Part Selector” on page 213.

## 8.3 Part Search

The part search feature locates a part by its name or using keywords attached to the part.

Click on the binoculars button on the schematic editor tool bar to open the search tool:

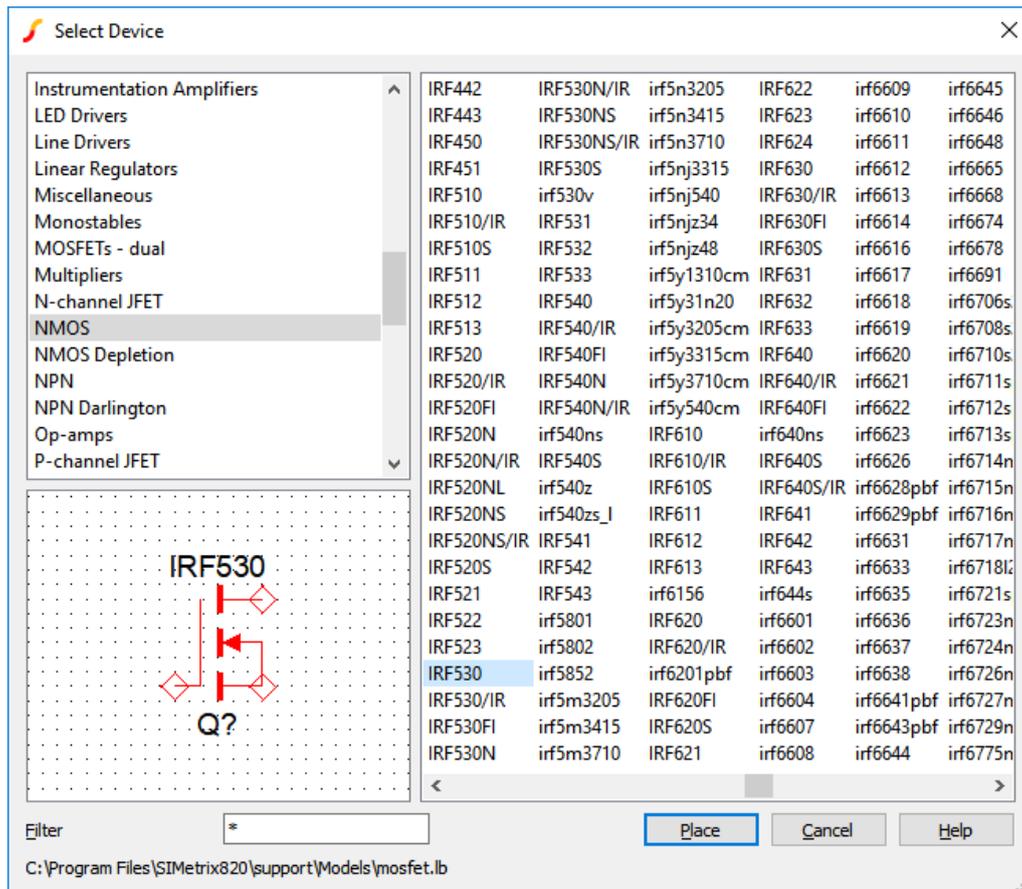


The part selector maintains a list of keywords associated with each part and this list will also be searched. The results will be displayed as you type. You can select whether or not you wish to include model library parts in the search, just check the **Include model library parts** box to enable

When you have located the part you desire, select it from the list then click **Ok**. This will place the part on the schematic in the usual way.

## 8.4 Model Library Browser

Numbered parts - that is parts that are located in the model library - may be accessed via the Model Library Browser. Select menu **Place | From Model Library** to open it.



Select the appropriate category on the left then the part number on the right. The picture above shows what you would see if selecting an IRF530 from the NMOS category. If you are not sure what category the part is in, select the '\* All Devices \*' category which you will find at the bottom of the category list.

If you are looking for a part that you installed (as opposed to a part supplied with SIMetrix) then you will find it in the '\* All User Models \*' category as well as the '\* All Devices \*' category. If installed within the last 30 days, you will also find it under '\* Recently Added Models \*'.

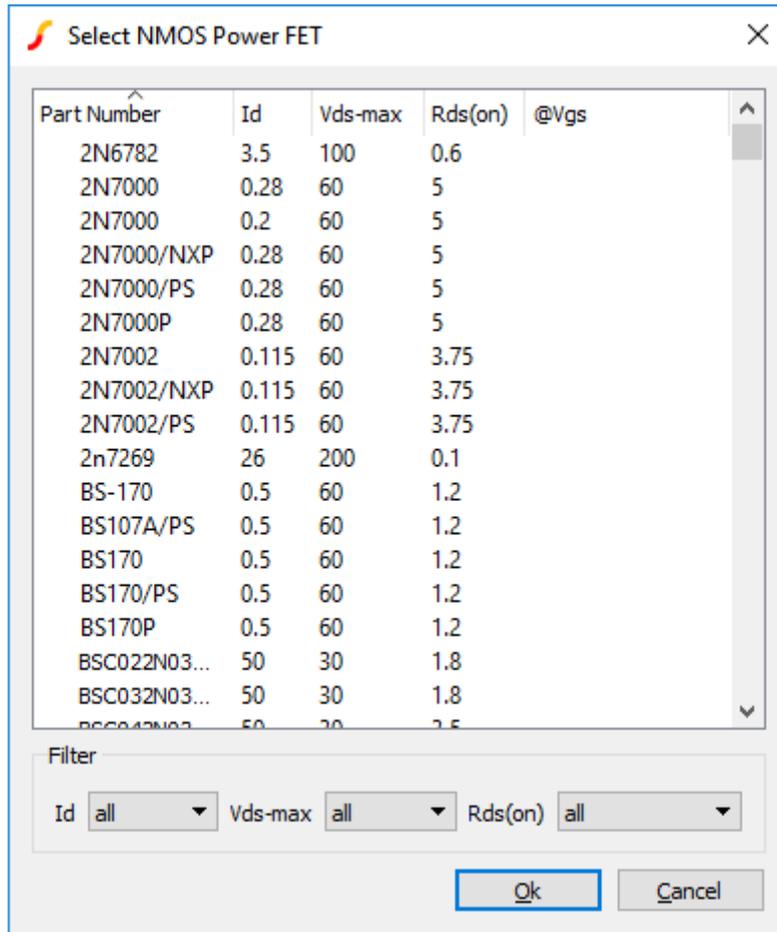
To reduce the number of devices displayed to a manageable level, you can specify a filter. You can use the wild-cards '\*' and '?' here. '\*' will match 1 or more of any character while '?' will match any single character. So, '\*' on its own will match any string and so all devices will be displayed. But 'IRF\*' will display any device that starts with the three letters 'IRF'. 'IRF???' will display any device beginning with IRF and followed by three and only three characters.

### 8.4.1 Selecting a Model by Specification

For some classes of part, you can select a model based on a desired specification. For example, if you are looking for an n-channel power FET with a 200V max  $V_{ds}$  and  $R_{ds(on)}$  less than 0.5Ohms, you can enter this specification and obtain a list of parts that meet it. Currently, devices supported for this scheme include power FETs, diodes and Zener diodes. This feature is available in both SIMetrix and SIMPLIS modes.

To use this method of selecting a device, proceed as follows:

1. Select one of the menus listed under **Place | Select by Specification**. You can also access this feature using the Part Selector. The picture below show what you would see if selecting NMOS Power FET



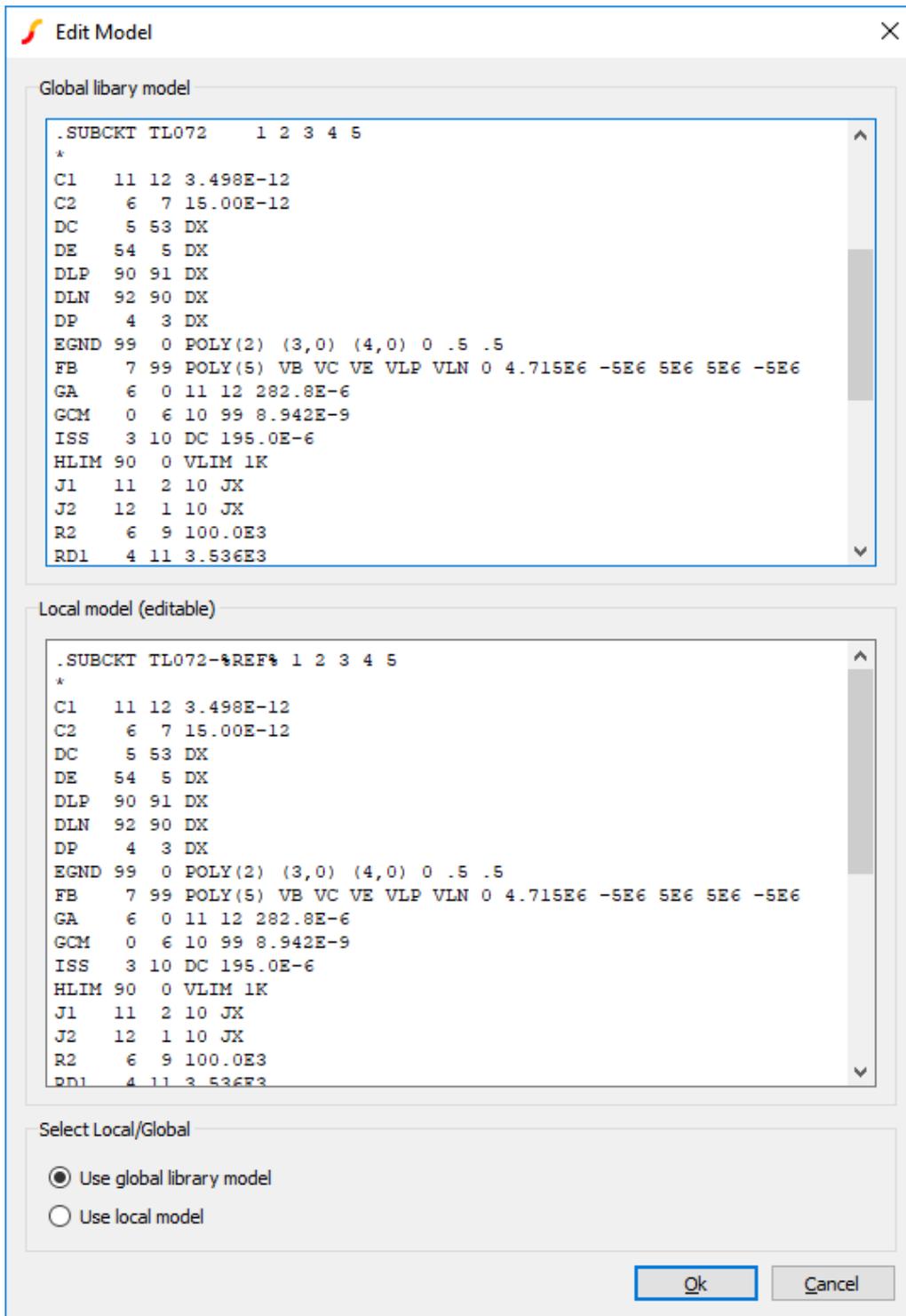
- Note that this is not the entire list of available parts of the desired type; only parts for which a specification is available will be listed. However, there should be a representative range so that it should be possible to find a model suitable for initial development work.

To filter listed parts to include your desired spec, use the drop-down boxes at the bottom of the box in the Filter group. Note that the upper end of each range is exclusive. So for example, '100-500' means everything from and including 100 up to but *not* including 500.

- Once you have selected your model, click Ok to place in the usual way.

### 8.4.2 Viewing and Editing Models

All parts selected using the model library browser as described above are defined using a model located in the global model library. You can view this model using the right click popup menu **View/Edit Model...** . This will display something like the following dialog box:



The top half of the above box shows the definition of the model in the library. The bottom half shows an editable local copy of the model. To begin with, this will be exactly the same as the library model shown in the top half. But you may edit this as desired (but note you should not edit the top line starting `.subckt` or bottom line `.ends`). This model will then be used instead of the global library model if you select the check box at the bottom: **Use local model**.

You may subsequently swap between the global and local models at any time. The local model is stored in

the schematic instance as a property and will continue to be available even if you select the global model at some time. This allows you to freely swap between the library model and your own modified version.

Note that only models defined in the global library may be viewed and edited in this way. Models defined locally in the **F11** window or models defined using .lib or .inc may not currently be viewed or edited.

## 8.5 Numbered Parts in SIMPLIS

This section applies only to the SIMetrix/SIMPLIS product.

SIMPLIS works in a quite different way to SIMetrix (SPICE) and as a result its models for semiconductor devices are completely different. For that reason the selection of devices available from **Place | From Model Library** when in SIMPLIS mode will not be the same as in SIMetrix mode.

However, SIMetrix is able to convert some SPICE models for use with SIMPLIS. When a supported device is placed on a SIMPLIS schematic, a model parameter extraction routine is invoked to automatically convert the SPICE model to a SIMPLIS model. During the model parameter extraction process, SIMetrix/SIMPLIS automatically runs several SPICE simulations on the SPICE model and extracts the SIMPLIS model parameters. After the Piecewise Linear (PWL) model parameters have been extracted, the resulting device model will run in SIMPLIS.

The conversion of SPICE models to SIMPLIS models is covered in the Help system. Navigate to SIMPLIS Documentation -> SIMPLIS Parts -> Semiconductors.

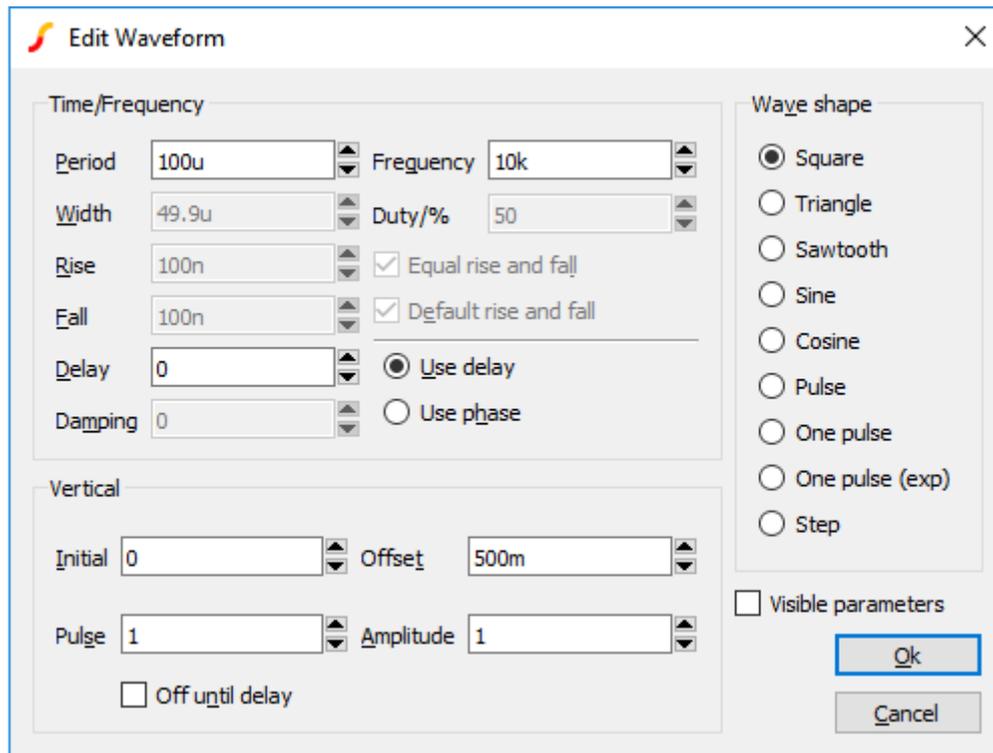
## 8.6 Circuit Stimulus

Most circuits require some form of input signal such as a pulse or sine wave. Such signals - or stimuli - are specified using a voltage or current source which is placed on the schematic in the usual way. A number of different types of source are available. These are described in the following sections.

### 8.6.1 Waveform Generator

This is used to create a time domain signal for transient analysis. This generator will work in both SIMetrix and SIMPLIS modes of operation. To place one of these devices, select menu **Place | Voltage Sources | Waveform Generator** for a voltage source or **Place | Current Sources | Waveform Generator** for a current source.

To specify the signal for the source, select then choose the popup menu **Edit Part...** or press **F7**. This will bring up the dialog box shown below. (In SIMPLIS simulation mode it will have an additional check box - see notes below)



Select the wave shape on the right hand side then enter the parameters as appropriate. The following notes provide details on some of the controls.

- **Damping** describes an exponential decay factor for sinusoidal wave-shapes. The decay is governed by the expression:

$$e^{-\text{damping} \times t}$$

- **Off until delay** if checked specifies that the signal will be at the **Initial** value until the delay period has elapsed.
- Note that some parameters can be specified in more than one way. For example both frequency and period edit controls are supplied. Changing one will cause the other to be updated appropriately. The same applies to duty and width and the vertical controls in the lower half.
- A Cosine wave-shape combined with a positive delay and with **Off until delay** checked, will only function correctly in SIMPLIS mode.
- If in SIMPLIS simulation mode, you will also see a check box titled **Source Idle during POP and AC analyses**. If checked, the source will be disabled in POP and AC analysis modes.

### 8.6.2 PWL Source

This device can be used to describe a piece wise linear source. A PWL source can describe any arbitrary wave shape in terms of time-voltage or time-current pairs. To place a PWL source select menu **Place | Voltage Sources | PWL Source** or **Place | Current Sources | PWL Source**.

To edit the device, select it and press **F7** or **Edit Part...** popup menu. This will open the Edit PWL Dialog which allows you to enter time and voltage/current values.

As well as entering values individually, you can also paste them from the Windows clipboard by pressing the **Paste** button or **ctrl-V**. The values can be copied to the clipboard using a text editor. The values may be separated by spaces, tabs, commas or new lines.

PWL sources may be used in both SIMetrix and SIMPLIS modes, but note that in SIMPLIS, PWL sources are limited to 256 points. Larger definitions are allowed in SIMetrix mode, but for very large PWL definitions, a PWLFILE source will be more efficient. See the “Analog Device Reference” in the *Simulator Reference Manual* for more information.

In SIMPLIS mode, a PWL source is always held inactive in the POP and AC analyses and is held constant to the value that is defined for the source at  $t=0$ .

### 8.6.3 Power Supply/Fixed Current Source

Select menu **Place | Voltage Sources | Power Supply** or **Place | Current Sources | DC Source** to place a fixed voltage or current source. These devices work in both SIMetrix and SIMPLIS modes.

### 8.6.4 AC Source

The small signal analysis modes, AC sweep and Transfer Function, require AC sources for their input stimulus.

To place an AC source select menu **Place | Voltage Sources | AC Source (for AC analysis)** or **Place | Current Sources | AC Source**.

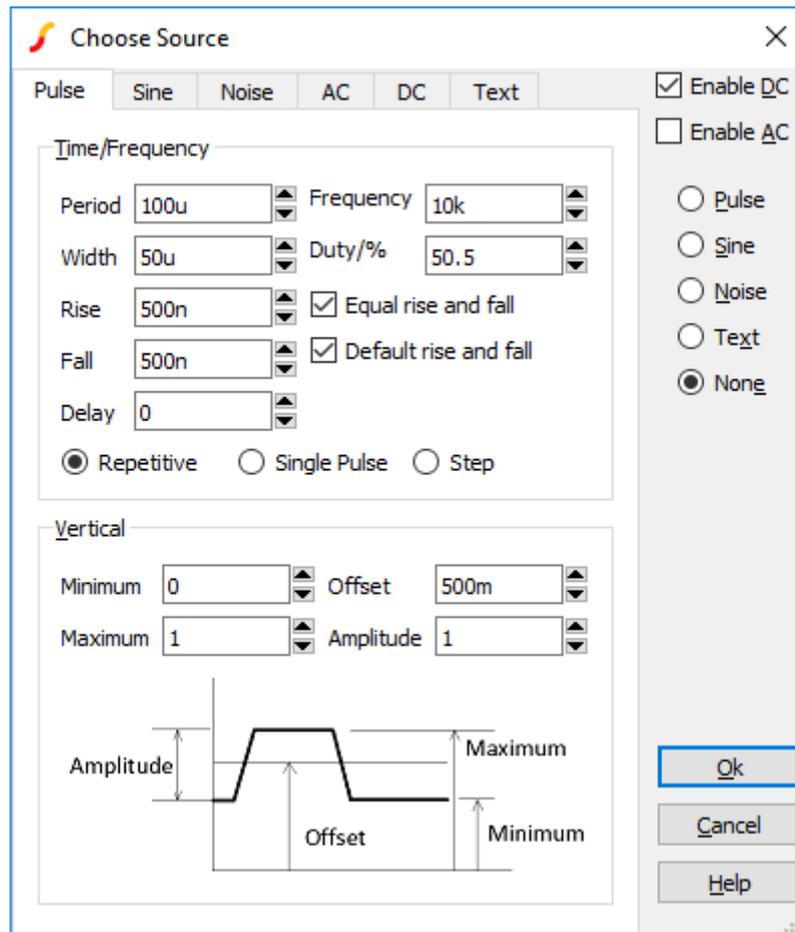
You can also use a Universal Source (see below) for the AC source for SIMetrix (i.e. not SIMPLIS) simulations. Be sure to check the **Enable AC** check box in the Universal Source.

### 8.6.5 Universal Source

All of the sources described above can be used in both SIMetrix and SIMPLIS modes of operation. In SIMetrix mode there is also a Universal source which provides the function of transient, AC and DC sources all in one device. In addition, the Universal source may be used to create a random noise source.

To place a universal source, select menu **Place | Voltage Sources | Universal Source** or **Place | Current Sources | Universal Source**.

To edit a universal source, select the device and press **F7** or popup menu **Edit Part....** This will display the following dialog box:



Refer to the following for details of each tabbed sheet:

[“Pulse sheet” on page 133](#)

[“Sine sheet” on page 135](#)

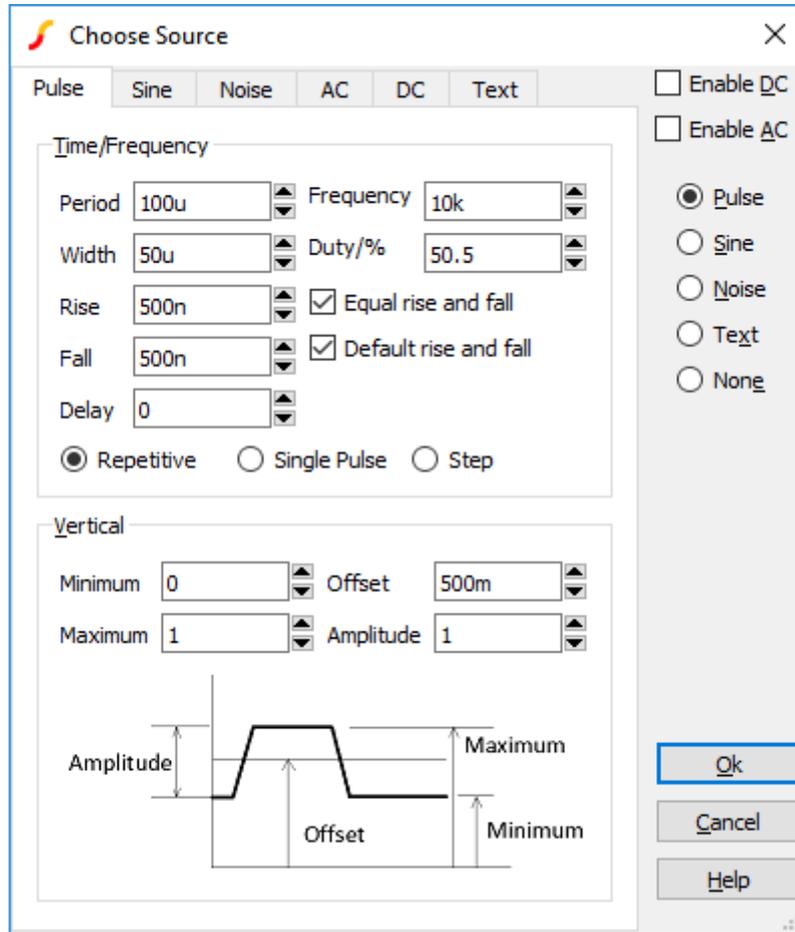
[“Noise sheet” on page 136](#)

[“AC sheet” on page 137](#)

[“DC sheet” on page 137](#)

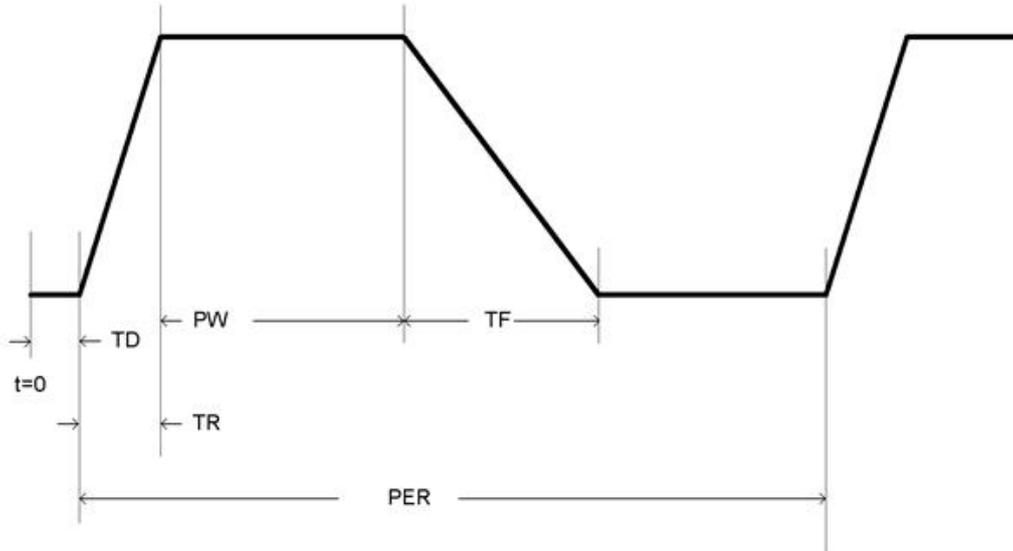
[“Text sheet” on page 139](#)

## Pulse Sheet



This selects the values for pulse waveform. For the values to be used the Pulse radio button at the right of the box must be selected.

The diagram below defines the pulse waveform timings.



PER = Period

PW = Width

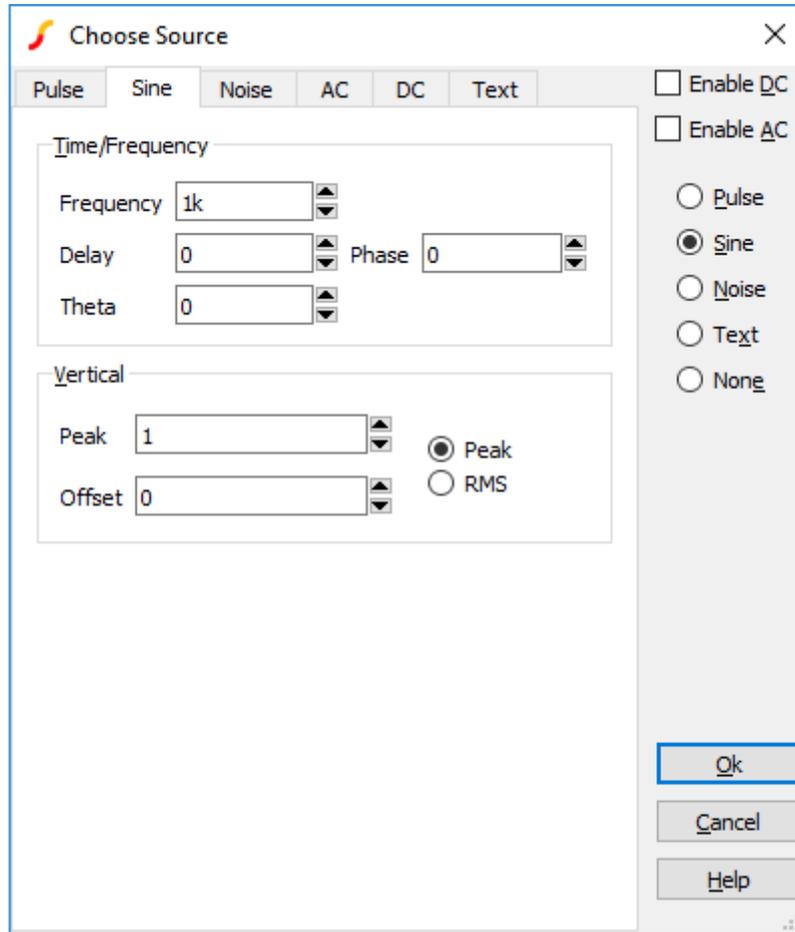
TR = Rise

TF = Fall

TD = Delay

Period/Width and Frequency/Duty boxes are two different ways of specifying the same thing. If you modify one pair the others change accordingly.

## Sine Sheet



This selects values for a sine wave generator. For the values to be used the Sine radio button at the right of the box must be selected.

The shape of the waveform is described by:

0 to delay:  $voffset$

delay to Stop time  $voffset + vpeak \cdot e^{-(t-delay) \cdot theta} \cdot \sin(2 \cdot \pi \cdot freq \cdot (t + delay))$

Where:

$t$  : time

$voffset$  : value in Offset edit box

$vpeak$  : value in Peak edit box. If RMS is selected,  $vpeak = v_{rms} \cdot \sqrt{2}$

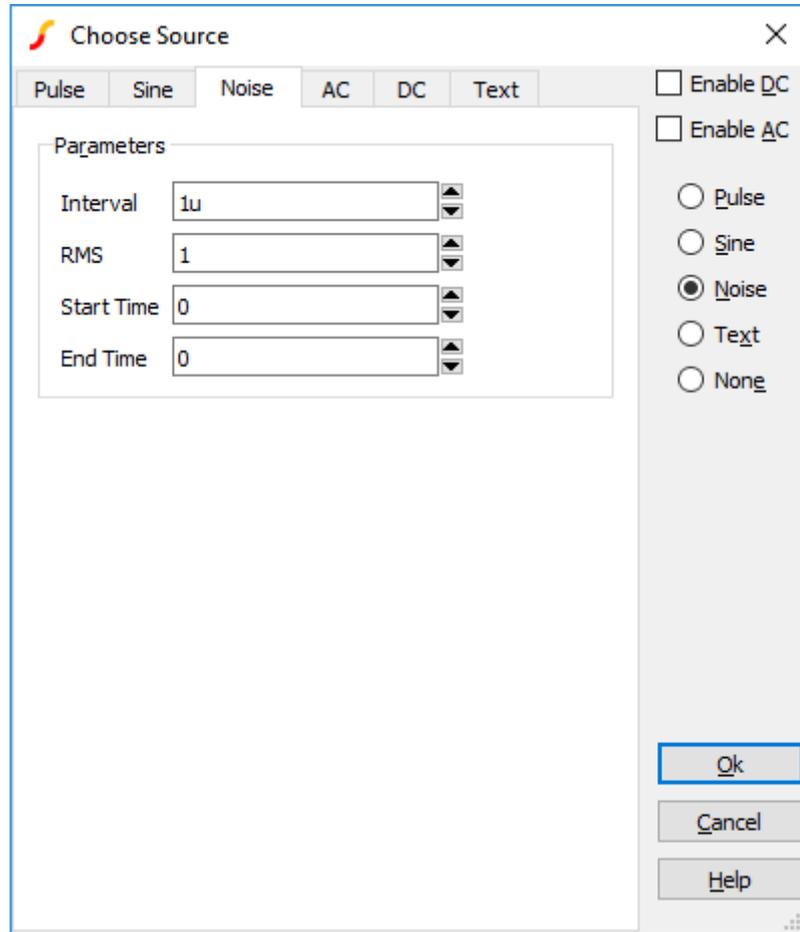
$delay$  : value in Delay edit box

$theta$  : value in Theta edit box

$freq$  : value in Frequency edit box

Phase is implemented using a negative delay value. For example, a phase of 30 degrees is entered, the delay will be  $30/360.1/freq$

## Noise Sheet

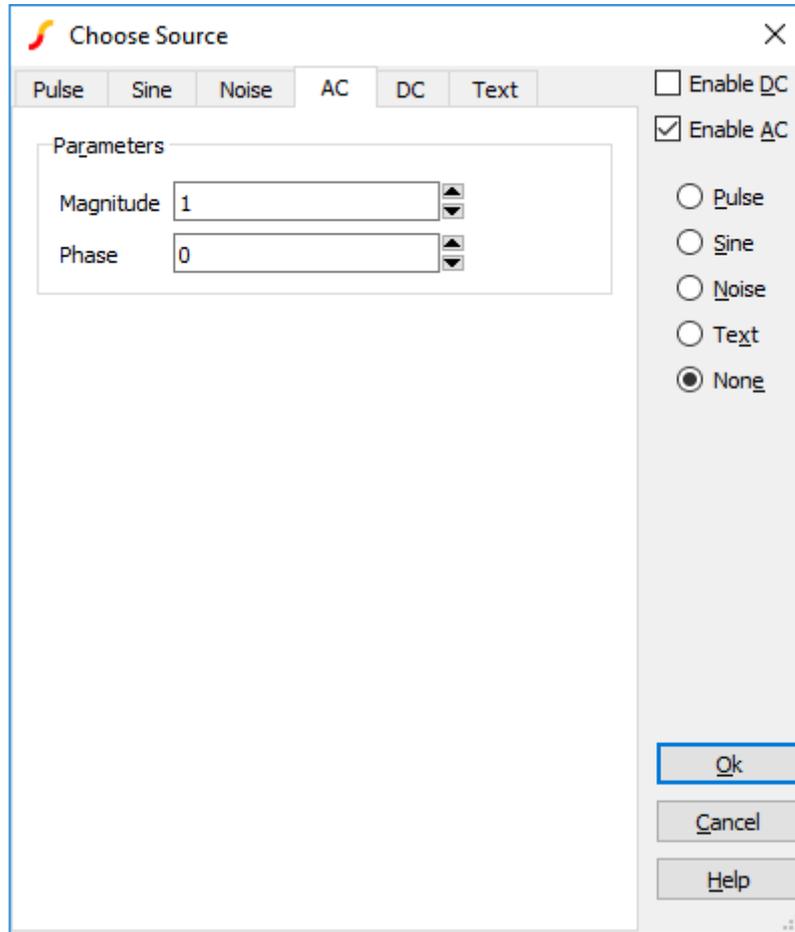


To enable noise, select the **Noise** radio button at the right of the dialog box.

Source generates a random value at `guiInterval` with distribution such that spectrum of signal generated is approximately flat up to frequency equal to  $1/(2*\text{interval})$ . Amplitude of noise is **RMS** volts.

Start time and End time values may be entered to define the interval over which the noise signal is active. If End time is zero, the noise will continue to the end of the simulation.

## AC Sheet



The entries in the AC sheet are only used for AC analysis.

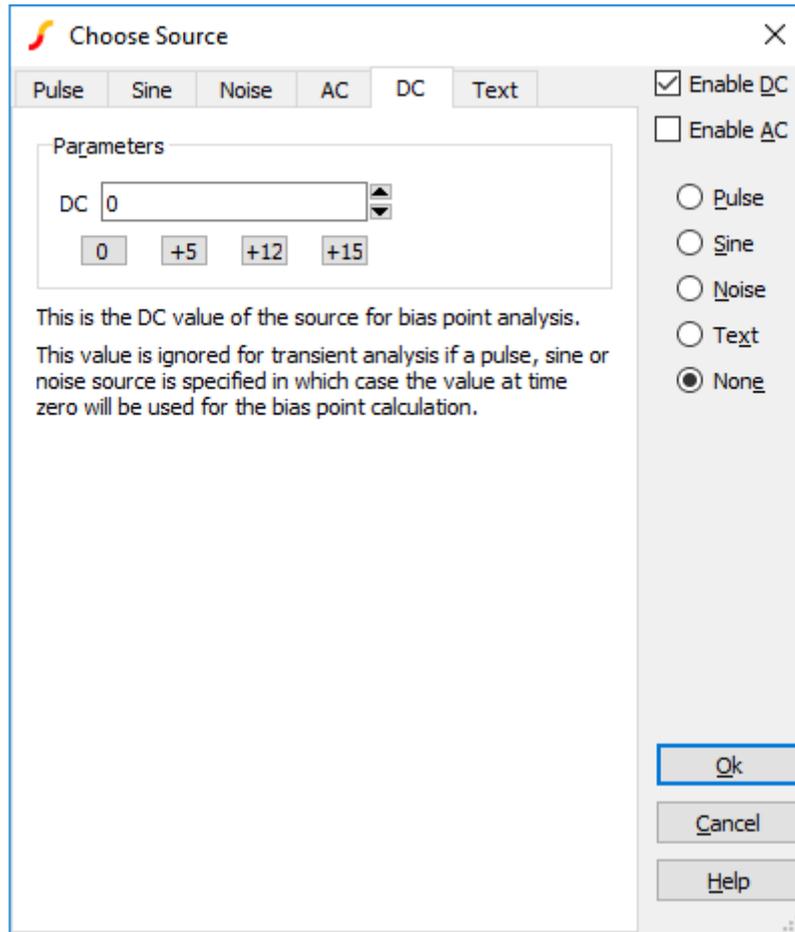
Sets the magnitude and phase of the source for AC analysis. To take effect the Enable AC check box must be checked.

This specification has no effect in other analysis modes.

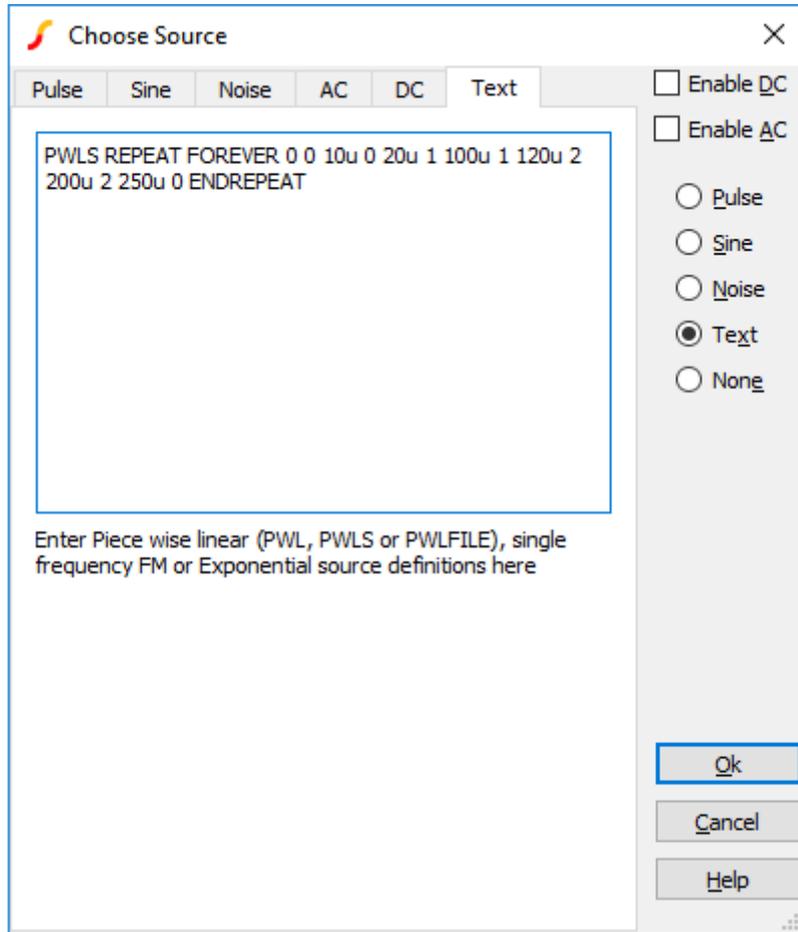
SIMetrix does not require an AC specification for the input referred source in Noise analysis.

## DC Sheet

Specifies DC value for bias point calculation. If one of sine, pulse, noise, piece wise linear, exponential, or single frequency fm sources is also specified, this value will be ignored and the time=0 value used instead.



## Text Sheet



The text sheet may be used to define one of the following

“PWL Definition” on page 139

“PWLS Definition” on page 140

“PWLFILE Definition” on page 145

The text sheet may also be used to define a single frequency FM or exponential source using traditional SPICE syntax. Refer to *Simulator Reference Manual/Analog Device Reference/Voltage Source/Exponential Source* and *Single Frequency FM* for details.

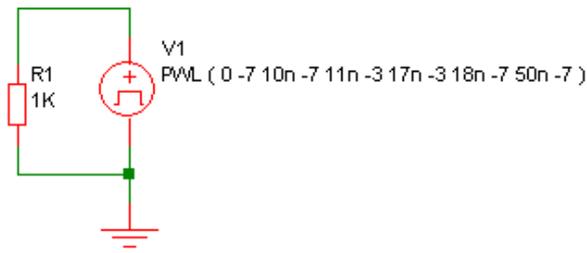
## PWL Definition

An piecewise linear source may be entered in the Text Sheet of the Universal source using the syntax described below.

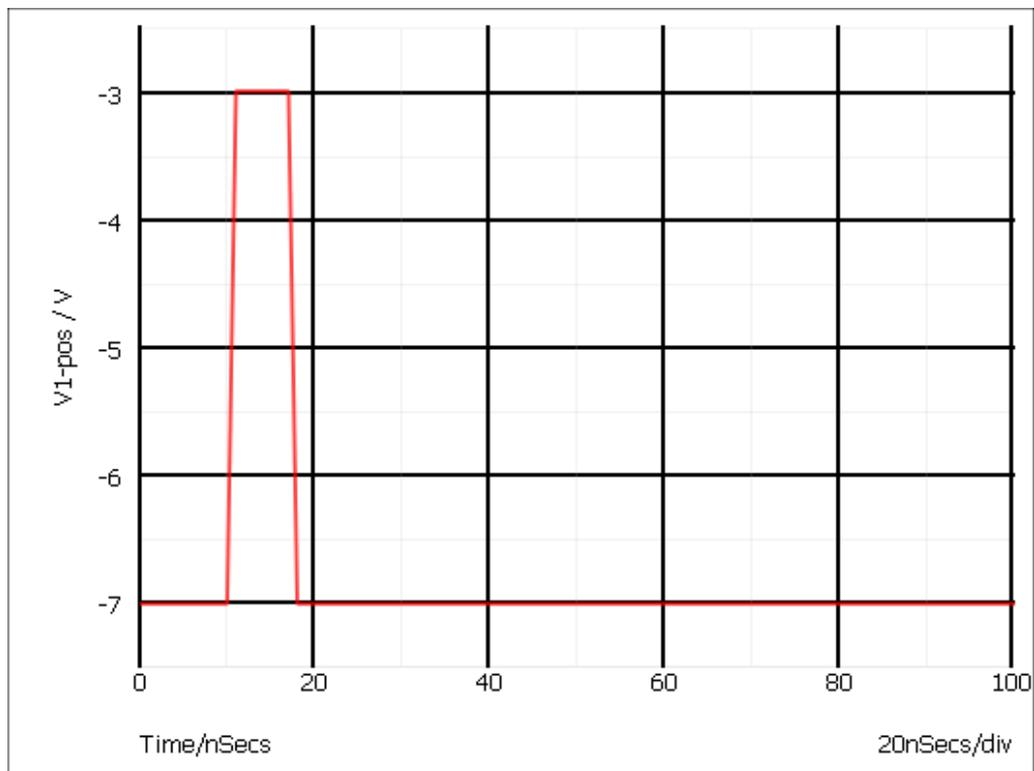
This applies to both voltage and current sources.

```
pw1 ( T1 V1 [T2 V2 [T3 V3 [... ]]] )
```

Each pair of values (Ti Vi) specifies that the value of the source is Vi at time = Ti. The value of the source at intermediate values of time is determined by using linear interpolation on the input values.

**Example**

Produces this waveform:

**PWLS Definition**

The PWLS definition allows more advanced signal sources to be created using a simple language syntax. The syntax is described below. For some example, see [“PWLS Examples” on page 142](#)

The syntax is in the following form:

```
PWLS [TIME_SCALE_FACTOR=time_factor] [VALUE_SCALE_FACTOR=value_factor]
pwls_spec [ pwls_spec ... ]
```

Where:

<i>time_factor</i>	Scales all time values in definition by <i>time_factor</i>	
<i>value_factor</i>	Scales all magnitude values by <i>value_factor</i>	
<i>pwls_spec</i>	may be one of the following:	
	<i>(time, value)</i>	Creates a single data point. <i>time</i> is relative to the current context.
	<i>(+time, value)</i>	Creates a single data point. <i>time</i> is relative to the previous point.
	REPEAT FOR <i>n pwls_spec</i> ENDREPEAT	Repeats <i>pwls_spec</i> <i>n</i> times.
	REPEAT FOREVER <i>pwls_spec</i> ENDREPEAT	Repeats <i>pwls_spec</i> forever
	SIN <i>sine_parameters</i> END	Creates a sinusoid. See table below for definition of <i>sine_parameters</i>
	PULSE <i>pulse_parameters</i> END	Creates a pulse train. See table below for definition of <i>pulse_parameters</i>
	FILE <i>file_name</i>	Loads a definition from file <i>file_name</i> . If the name contains spaces, it must be quoted

### Sine Parameters

Name	Description	Default	Compulsory
<b>FREQ</b>	Frequency	N/A	Yes
<b>PEAK</b>	Peak value of sine	1.0	No
<b>OFFSET</b>	Offset	0.0	No
<b>DELAY</b>	Delay before sine starts.	0.0	No
<b>PHASE</b>	Phase	0.0	No
<b>CYCLES</b>	Number of cycles. Use -1.0 for infinity	-1.0	No
<b>MINPOINTS</b>	Minimum number of timesteps used per cycle	13	No
<b>RAMP</b>	Frequency ramp factor	0.0	No

The sine value is defined as follows:

```

if t>0 OR DELAY<0
PEAK * SIN(f*2PI*t+PHASE*PI/180) + OFFSET
else
PEAK * SIN(PHASE*PI/180) + OFFSET

```

Where:

$$f = \text{FREQ} + t \times \text{RAMP}$$

$$t = \text{time} - \text{tref} - \text{DELAY}$$

*time* is the global simulation time

*tref* is the reference time for this spec

### Pulse Parameters

Name	Description	Default	Compulsory
<b>V0</b>	Offset	0	No
<b>V1</b>	Positive pulse value	1.0	No
<b>V2</b>	Negative pules value	-1.0	No
<b>RISE</b>	Rise time i.e time to change from V2 to V1	PERIOD/1000	No
<b>FALL</b>	Fall time i.e time to change from V1 to V2	PERIOD/1000	No
<b>WIDTH</b>	Positive pulse width	(PERIOD-RISE-FALL)/2	No
<b>PERIOD</b>	Period	N/A	Yes
<b>DELAY</b>	Delay before start	0	No
<b>CYCLES</b>	Number of complete cycles. -1 means infinity	-1	No

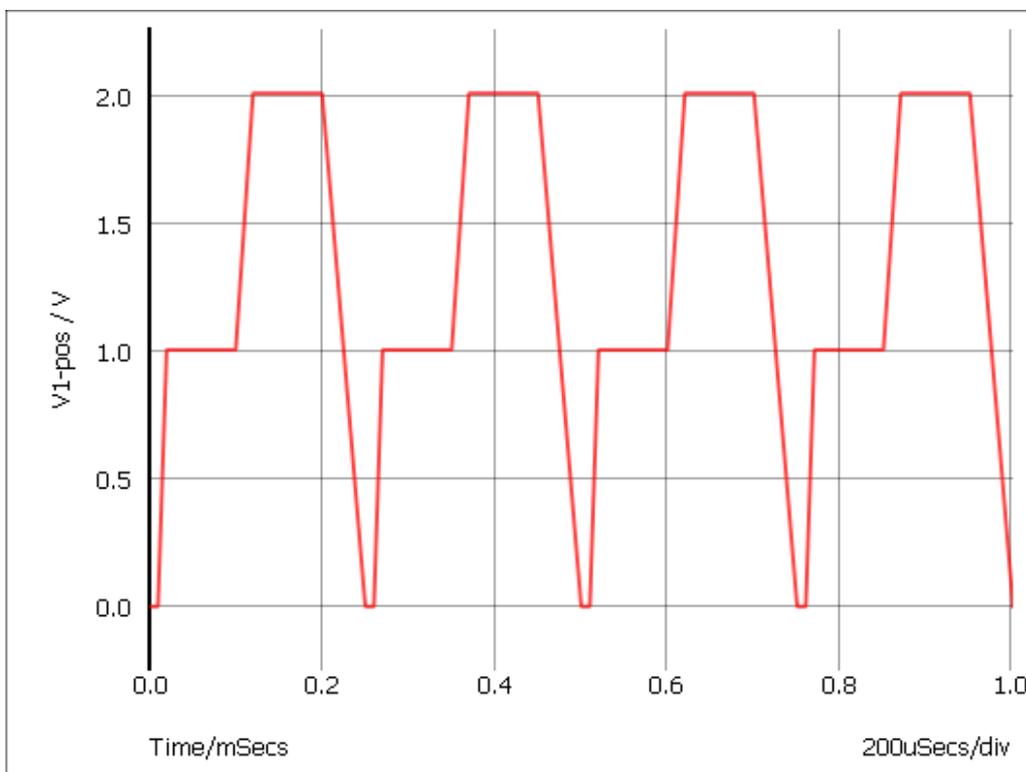
RISE, FALL, WIDTH and PERIOD must be greater than zero. DELAY must be greater than or equal to zero

## PWLS Examples

### Example 1

```
PWLS REPEAT FOREVER 0 0 10u 0 20u 1 100u 1 120u 2 200u 2 250u 0 ENDREPEAT
```

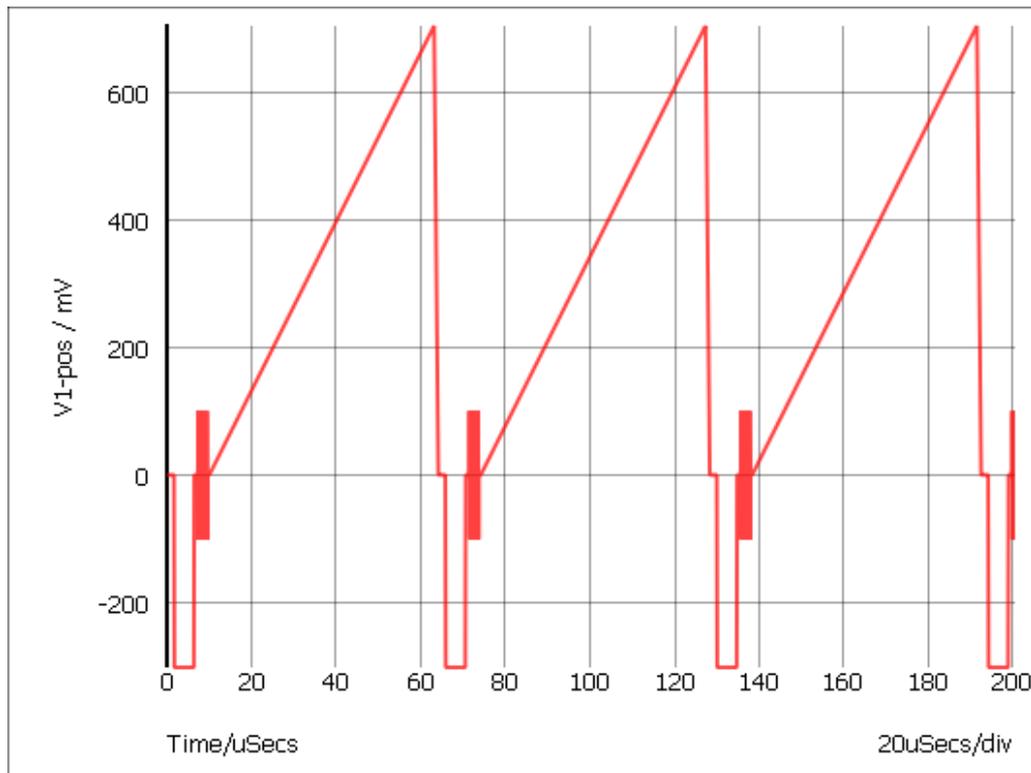
Produces a repeated signal as shown below:



**Example 2**

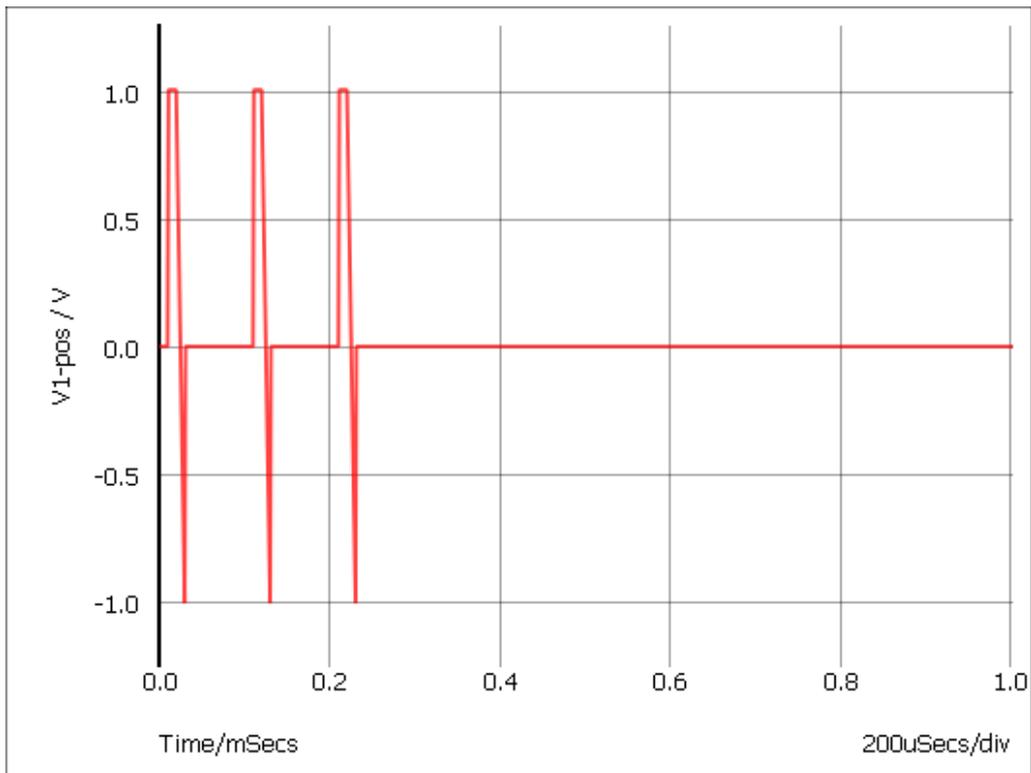
The following defines a simple grey scale PAL test signal.

```
PWLS REPEAT FOREVER
+ 0 0 1.65u 0 1.75u -0.3 6.35u -0.3 6.45u 0 7.3u 0
+ SIN FREQ=4.43Meg CYCLES=10 PEAK=0.1 END
+ 10u 0 63u 0.7 64u 0
+ ENDREPEAT
```

**Example 3**

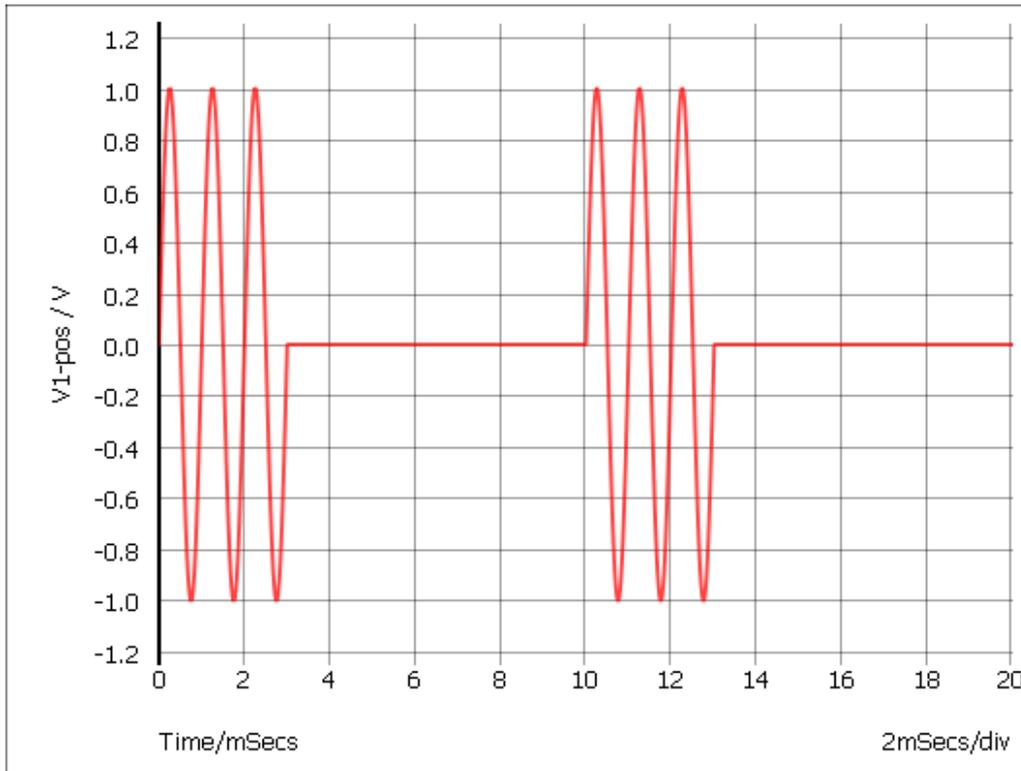
The following repeats a PWL definition 3 times only

```
PWLS REPEAT FOR 3 0 0 10u 0 +1u 1 20u 1 20u -1 30u -1 31u 0 100u 0 ENDREPEAT
```

**Example 4**

The following describes a tone burst signal

```
PWLS REPEAT FOREVER
+ SIN peak=1 delay=0 freq=1k cycles=3 minpoints=50 offset=0 END 0.01 0 ENDREPEAT
```



## PWLFILE Definition

```
PWLFILE filename
```

This performs the same function as the normal piece wise linear source except that the values are read from a file named *filename*.

The file contains a list of time voltage pairs in text form separated by any whitespace character (space, tab, new line). It is not necessary to add the '+' continuation character for new lines but they will be ignored if they are included. Any non-numeric data contained in the file will also be ignored.

Be aware that numbers in engineering format, i.e. using suffixes such as u for 1e-6 and k for 1e3, are not recognised in a PWLFILE definition. If your data does include such values, you can instead use the PWLS definition in the form `PWLS file=filename` which does accept such values.

### 8.6.6 Digital Sources

#### Digital Pulse

The digital pulse device creates a single shot or repetitive pulse in the digital event driven domain. As digital and analog nodes can be freely interconnected, analog parts may be connected to it but it is more efficient to use an analog pulse generator for this purpose.

#### To Place a Digital Pulse

1. Select menu **Place | Digital | Digital Pulse** or **Place |Digital |Digital Pulse (open emitter)** and place. The open emitter version can be "wire-ored" but requires a pull down resistor. You can use menu **Place | Digital | Pull-down for this**.
2. Select device then press **F7** or equivalent menu. Enter parameters as required. For a single pulse enter '0' for the period.

**Device Operation** This device supplies a repetitive or single pulse of defined period, delay and width. Optionally, the device may be specified to have an open emitter output allowing several pulse sources to be wire OR'ed to create complex pulses. All 5 main .MODEL parameters may also be specified on the device line as instance parameters in which case they override any values specified in the .MODEL control.

## Digital Signal Source

The digital signal source provides a multi bit arbitrary digital signal defined in a file. There is no standard schematic part for this device as it can be defined with an arbitrary number of outputs. For more information, see *Simulator Reference Manual/Digital Mixed Signal Device Reference/Digital Signal Source*.

### 8.6.7 Other Sources

#### Sine Tone Burst

Generates a sequence of sinusoidal bursts with a user defined number of cycles per burst, burst frequency and tone frequency.

Use menu **Place | Voltage Sources | Sine Tone Burst** then place device in the usual way. Editing the device will bring up a dialog with 6 parameters:

Parameter	Description
Burst Freq.	Burst frequency
Tone Freq.	Frequency of the sinusoidal tone
Num Tone Cycles	Number of sinusoidal cycles in each burst
Peak	Peak voltage
Offset	Offset voltage
Points Per Cycle	Minimum number of time steps in each sinusoidal cycle. Increasing this number will improve the accuracy of the simulation at the expense of simulation speed

#### Swept Sinusoid

Generate a sinusoidal signal with linearly increasing frequency.

Use menu **Place | Voltage Sources | Swept Sine** then place in the usual manner. Editing the device will bring up a dialog with 6 parameters:

Parameter	Description
Start Frequency	Starting frequency
End Frequency	Frequency at the end of the ramp
Interval	Time taken to ramp from start frequency to end frequency
Peak	Peak voltage
Offset	Offset voltage

Parameter	Description
Points Per Cycle	Minimum number of time steps in each sinusoidal cycle. Increasing this number will improve the accuracy of the simulation at the expense of simulation speed

## Bidirectional Pulse

Generates a symmetrical bidirectional pulse waveform.

Use menu **Place | Voltage Sources | Bidirectional Pulse** then place device in the usual way. Editing the device will bring up a dialog with 3 parameters:

Parameter	Description
P-P Voltage	Peak to peak voltage
Frequency	Pulse frequency
Delay	Delay after start

## 8.7 Generic Parts

As explained in the overview generic parts are devices that are defined by one or more parameters entered by the user after the part is placed. The following generic parts are available:

Device	SIMPLIS support?
Ideal Transformer	Yes
Saturable Inductors and Transformer	No
Inductor	Yes
Capacitor	Yes
Resistor	Yes
Potentiometer	Yes
Transmission Line (Lossless)	No
Transmission Line (Lossy)	No
Infinite capacitor	No
Infinite inductor	No
Voltage source	Yes
Current source	Yes
Voltage controlled voltage source	Yes
Voltage controlled current source	Yes
Current controlled voltage source	Yes
Current controlled current source	Yes
Voltage controlled switch	No

Device	SIMPLIS support?
Voltage controlled switch with Hysteresis	Yes
Delayed Switch	No
Parameterised Opamp	Yes
Parameterised Opto-coupler	No
Parameterised Comparator	No
VCO	No

Device
Non-linear transfer function
Laplace transfer function
Non-linear resistor
Non-linear capacitor
Non-linear inductor
Analog-Digital converter
Digital-Analog converter
Digital counter
Digital shift register
NAND/NOR/OR/AND gates
Digital bus register

### 8.7.1 SIMPLIS Primitive Parts

The following parts are only available with the SIMetrix/SIMPLIS product and when in SIMPLIS. They can all be found under the menu **Place | SIMPLIS Primitives**. For full details, see the *SIMPLIS Reference Manual*.

Device
Comparator
Set-reset flip-flop
Set-reset flip-flop clocked
J-K flip-flop
D-type flip-flop
Toggle flip-flop
Latch
Simple switch - voltage controlled
Simple switch - current controlled
Transistor switch - voltage controlled
Transistor switch - current controlled
VPWL Resistor

---

**Device**


---

IPWL Resistor

PWL Capacitor

PWL Inductor

## 8.7.2 Saturable Inductors and Transformers

SIMetrix is supplied with a number of models for inductors and transformers that correctly model saturation and, for most models, hysteresis. As these parts are nearly always custom designed there is no catalogue of manufacturers parts as there is with semiconductor devices. Consequently a little more information is needed to specify one of these devices. This section describes the facilities available and a description of the models available.

### Core Materials

The available models cover a range of ferrite and MPP core materials for inductors and transformers with any number of windings. The complete simulation model based on a library core model is generated by the user interface according to the winding specification entered.

### Placing and Specifying Parts

1. Select the menu **Place | Magnetics | Saturable Transformer/Inductor...** You will see the following dialog box:

The dialog box 'Define Saturable Transformer/Inductor' contains the following configuration:

**Configuration**

# Primaries: 1 | # Secondaries: 1 | Coupling factor: 1

	Winding	Number of turns	Polarity	Inductance
1	Primary 1	100	+	125.664mH
2	Secondary 1	100	+	125.664mH

**Define core**

Select core type  
 Manual entry

Core type: E14/3.5-3C90-A63-P

Core material: 3C81

Units:

- mm
- cm
- inches
- metres

Ae: 100  
 Le: 10  
 Ue: 1k

Primary inductance: 125.664mH  
 Saturation current: 35.8099mA

Buttons: Ok, Cancel, Help

2. Specify the number of windings required for primary and secondary in the **Configuration** section. If you just want a single inductor, set primary turns to 1 and secondaries to 0.
3. Specify the number of turns for each winding in the windings list below the **Configuration** section. You can also define positive or negative polarity which will control the orientation of the winding.
4. Specify the coupling factor in the **Configuration** section. The coupling factor is the same for all windings. You can define different coupling factors for each winding by adding ideal inductors in series with one or more windings. In some instances it may be necessary to add coupled inductors in series. This is explained in more detail in [“Coupling Factor” on page 152](#).
5. Specify the core characteristics in the **Define Core** section. A number of standard core sets are pre-programmed and can be selected from the **Select Core Type** list at the top. If the part you wish to use is not in the list or if you wish to use a variant with a - say - different air gap, you can manually enter the characteristics by clicking on the **Manual Entry** check box.

The values you need to enter are

Ae	Effective Area
Le	Effective Length
Ue	Relative Permeability
Core Material	

## Model Details

The models for saturable magnetic parts can be found in the file `cores.lb`. Most of the models are based on the Jiles-Atherton magnetic model which includes hysteresis effects. The MPP models use a simpler model which does not include hysteresis. These models only define a single inductor. To derive a transformer model, the user interface generates a subcircuit model that constructs a non-magnetic transformer using controlled sources. The inductive element is added to the core which then gives the model its inductive characteristics.

The model does not currently handle other core characteristics such as eddy current losses nor does it handle winding artefacts such as resistive losses, skin effect, inter-winding capacitance or proximity effect.

### 8.7.3 Ideal Transformers

Ideal transformers may be used in both SIMetrix and SIMPLIS modes. Note that SIMPLIS operation is more efficient if the coupling factors are set to unity.

To define an ideal transformer, select the menu **Place | Passives | Ideal Transformer....** This will open the following dialog box:

**Define Ideal Transformer**

**Configuration**

# Primaries  # Secondaries

**Coupling**

Inter-primary coupling  Primary-secondary coupling  Inter-secondary coupling

	Winding	Number of turns	Polarity	Inductance
1	Primary 1	<input type="text" value="1"/>	<input type="text" value="+"/>	<input type="text" value="10u"/>
2	Secondary 1	<input type="text" value="1"/>	<input type="text" value="+"/>	<input type="text" value="10uH"/>

## Configuration

Specify the number of primaries and secondaries. You can specify up to 20 of each.

## Define winding turns and inductance

Specify the number of turns for each winding in the list at the bottom of the window. Also set the inductance for the first primary. The inductances for the other windings will automatically update. These are calculated from the primary inductance and the relevant turns ratio.

You may also specify the polarity for each winding. The polarity of the winding will be shown on the schematic symbol using the conventional dot at one end of the winding.

## Coupling factor

- Inter-primary coupling      Coupling factor between primaries.
- Inter secondary coupling      Coupling factor between secondaries.
- Primary-secondary coupling      Coupling factor from each primary to each secondary.

This method of implementing an ideal transformer is not totally general purpose as you cannot arbitrarily define inter winding coupling factors. If you need a configuration not supported by the above method, you can define any ideal transformer using ideal inductors and the Mutual Inductance device. The SIMetrix version is explained in [“Mutual Inductors” on page 152](#). For the SIMPLIS equivalent, see the SIMPLIS reference manual.

### 8.7.4 DC Transformers

The DC transformer model is like a normal transformer except that it has - in effect - infinite inductance. So it will work at DC and has zero magnetisation current. To place a DC transformer, select menu **Place | Magnetics | Ideal DC Transformer....** This will open the following dialog box:

**Define Ideal DC Transformer**

Configuration

# Primaries  # Secondaries

	Winding	Number of turns	Polarity
1	Primary 1	<input type="text" value="1"/>	+ ▼
2	Secondary 1	<input type="text" value="1"/>	+ ▼

Ok Cancel Help

## Configuration

Specify the number of primaries and secondaries. You can specify up to 20 of each.

## Define winding turns

Define the number of turns for each winding in the list. Note that the behaviour of the part depends only on the ratios between the turns. You may also specify the polarity for each winding. The polarity of the winding will be shown on the schematic symbol using the conventional dot at one end of the winding.

### 8.7.5 Coupling Factor

The standard user interface for both saturable and ideal transformers provide only limited flexibility to specify inter-winding coupling factor. In the majority of applications, coupling factor is not an important issue and so the standard model will suffice.

In some applications, however, the relative coupling factors of different windings can be important. An example is in a flyback switched mode supply where the output voltage is sensed by an auxiliary winding. In this instance, best performance is achieved if the sense winding is strongly coupled to the secondary. Such a transformer is likely to have a different coupling factor for the various windings.

You can use external leakage inductances to model coupling factor and this will provide some additional flexibility. One approach is to set the user interface coupling factor to unity and model all non-ideal coupling using external inductors. In some cases it may be necessary to couple the leakage inductors. Consider for example an E-core with 4 windings, one on each outer leg and two on the inner leg. Each winding taken on its own would have approximately the same coupling to the core and so each would have the same leakage inductance. But the two windings on the centre leg would be more closely coupled to each other than to the other windings. To model this, the leakage inductances for the centre windings could be coupled to each other using the mutual inductor method described in the next section.

### 8.7.6 Mutual Inductors

You can specify coupling between any number of ideal inductors, using the mutual inductor device. There is no menu or schematic symbol for this. It is defined by a line of text that must be added to the netlist. (See [“Manual Entry of Simulator Commands” on page 42](#)). The format for the mutual inductance line is:

```
Kxxxx inductor_1 inductor_2 coupling_factor
```

Where:

*inductor\_1* Part reference of the first inductor to be coupled.  
*inductor\_2* Part reference of the second inductor to be coupled.  
*coupling\_factor* Value between 0 and 1 which defines strength of coupling.

## Note

If more than 2 inductors are to be coupled, there must be a K device to define every possible pair.

## Examples

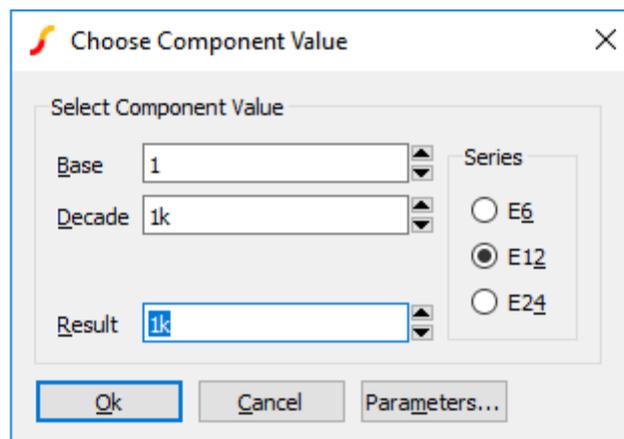
```
** Couple L1 and L2 together
K12 L1 L2 0.98
** Couple L1, L2 and L3
K12 L1 L2 0.98
K23 L2 L3 0.98
K13 L1 L3 0.98
```

### 8.7.7 Resistors

Resistors may be used in both SIMetrix and SIMPLIS modes. Note that in SIMetrix mode a number of additional parameters may be specified. These will not work with SIMPLIS and must not be specified if dual mode operation is required.

Select from **Place | Passives** menu.

To edit value use **F7** or select popup menu **Edit Part...** menu as usual. This will display the following dialog for resistors.



You can enter the value directly in the Result box or use the Base and Decade up/down controls.

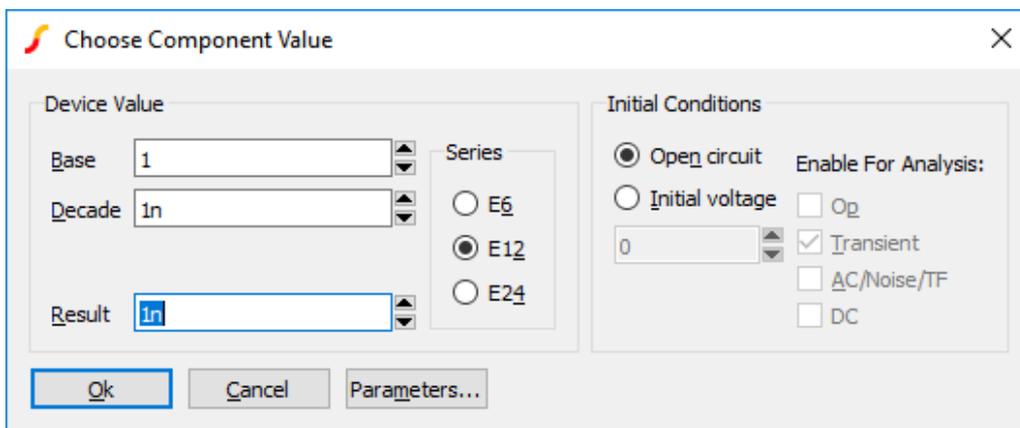
## Additional Parameters

Press **Parameters...** button to edit additional parameter associated with the device such as temperature coefficients (TC1, TC2). Refer to device in the *Simulator Reference Manual* for details of all device parameters.

### 8.7.8 Ideal Capacitors and Inductors

Capacitors and inductors may be used in both SIMetrix and SIMPLIS modes. Note that in SIMetrix mode a number of additional parameters may be specified. These will not work with SIMPLIS and must not be specified if dual mode operation is required.

The following dialog will be displayed when you edit a capacitor or inductor:



The device value is edited in the same manner as for resistors. You can also supply an initial condition that defines how the device behaves while a DC operating point is calculated. For capacitors you can either specify that the device is open circuit or alternatively you can specify a fixed voltage. For inductors, the device can be treated as a short circuit or you can define a constant current.

### 8.7.9 Infinite Capacitors and Inductors

The infinite capacitors and inductors are often useful for AC analysis.

To place an infinite capacitor, select menu **Place | Passives | Infinite Capacitor** To place an infinite inductor, select menu **Place | Magnetics | Infinite Inductor**

The infinite capacitor works as follows:

1. During the DC bias point calculation, it behaves like an open circuit, just like a regular finite capacitor.
2. During any subsequent analysis it behaves like a voltage source with a value equal to the voltage achieved during the DC bias point calculation.

The infinite inductor behaves as follows:

1. During the DC bias point calculation, it behaves like a short circuit, just like a regular finite inductor.
2. During any subsequent analysis it behaves like a current source with a value equal to the current achieved during the DC bias point calculation.

These parts allow you to close a feedback loop during the DC bias point then open it for any subsequent analysis.

The infinite capacitor is a built in primitive part and is actually implemented by the voltage source device. The infinite inductor is a subcircuit using an infinite capacitor and some controlled sources.

### 8.7.10 Multi-Level Capacitors

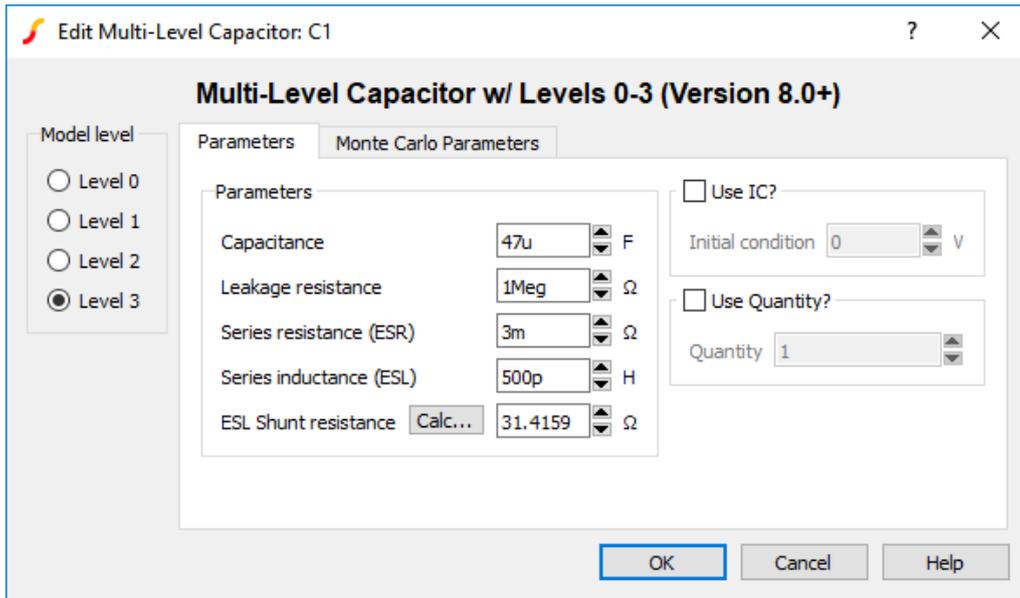
The Multi-Level Capacitor has four model levels. As the level increases, additional parasitic circuit elements are added. This capacitor, available with Version 8.0 or later, is used to model a wide variety of capacitor types where the number and type of parasitic circuit elements can be configured.

Model Name:	Multi-Level Capacitor (Level 0-3 w/Quantity)
Simulator:	This device is compatible with both the SIMetrix and SIMPLIS simulators
Parts Selector Menu Location (SIMetrix):	<b>Passives   Capacitors   Multi-Level Capacitor, US Style, (V8+)</b>
Parts Selector Menu Location (SIMetrix):	<b>Passives   Capacitors   Multi-Level Capacitor, EUR Style, (V8+)</b>
Parts Selector Menu Location (SIMetrix):	<b>Passives   Capacitors   Multi-Level Capacitor, Non-Polarised, (V8+)</b>
Parts Selector Menu Location (SIMPLIS):	<b>Commonly Used Parts   Multi-Level Capacitor (Level 0-3 w/Quantity)(Version 8.0+)</b>
Symbol Library:	passives.sxslb
Model File (SIMetrix):	preproc.lb
Model File (SIMPLIS):	simplis_param.lb
Subcircuit Name:	MULTI_LEVEL_CAP_QTY
Multiple Selections:	Multiple devices can be selected and edited simultaneously

### Editing the Multi-Level Capacitor

To configure the Multi-Level Capacitor, follow these steps:

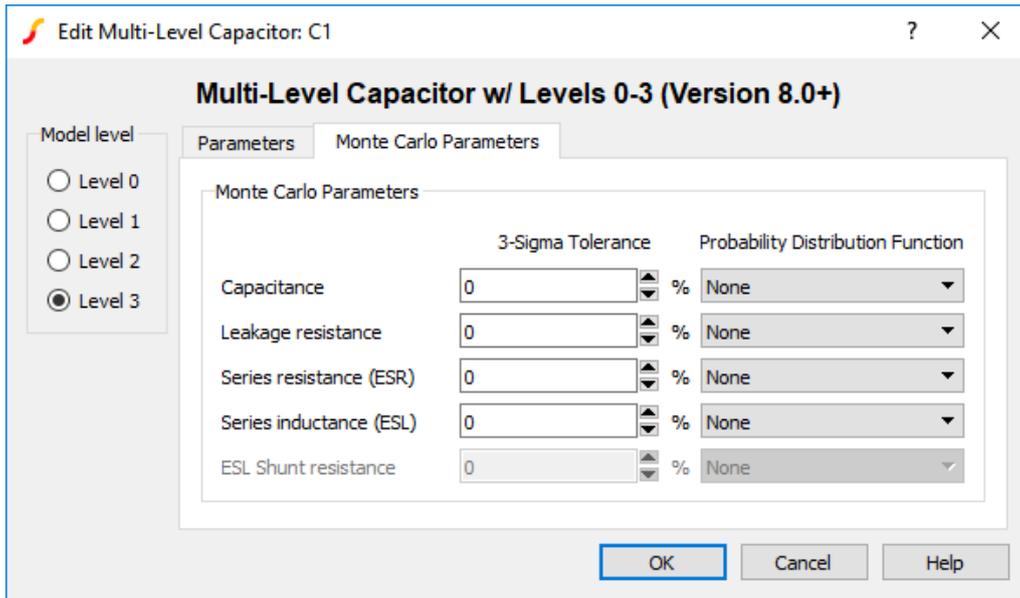
1. Double click the symbol on the schematic to open the editing dialog to the Parameters tab.
2. Choose a model level, and then make the appropriate changes to the fields described in the table below the image.



Label	Levels	Units	Description
Capacitance	0,1,2,3	F	Capacitance value
Leakage resistance	1,2,3	$\Omega$	Leakage resistance across the capacitor
Series resistance (ESR)	2,3	$\Omega$	The equivalent series resistance (ESR)
Series inductance (ESL)	3	H	The equivalent series inductance (ESL)
ESL Shunt resistance	3	$\Omega$	The shunt resistance in parallel with the ESL. This resistance is included to limit the maximum frequency response of the parasitic ESL, which maximizes the simulation speed in SIMPLIS mode. Click Calc... to calculate this value.
Use IC?	all	n/a	<ul style="list-style-type: none"> <li>If checked, the model uses the specified Initial condition.</li> <li>If unchecked, the capacitor initial condition is an open circuit.</li> </ul>
Initial condition	all	V	Initial voltage of the capacitor at time=0
Use Quantity?	all	n/a	If checked, the model uses the specified Quantity of capacitors in parallel.
Quantity	all	n/a	Number of capacitors in parallel. Note: To maximize simulation speed when using SIMPLIS, place a single symbol and specify a value for this parameter instead of placing multiple capacitors.

The model has built-in provisions for setting the Monte Carlo Parameters for all parameters except the ESL Shunt resistance. The Monte-Carlo Parameters are entered on the Monte Carlo Parameters tab shown below. To define the Monte Carlo Parameters follow these steps:

1. Click on the Monte Carlo Parameters tab.
2. Make the appropriate changes to the fields described in the table below the image.



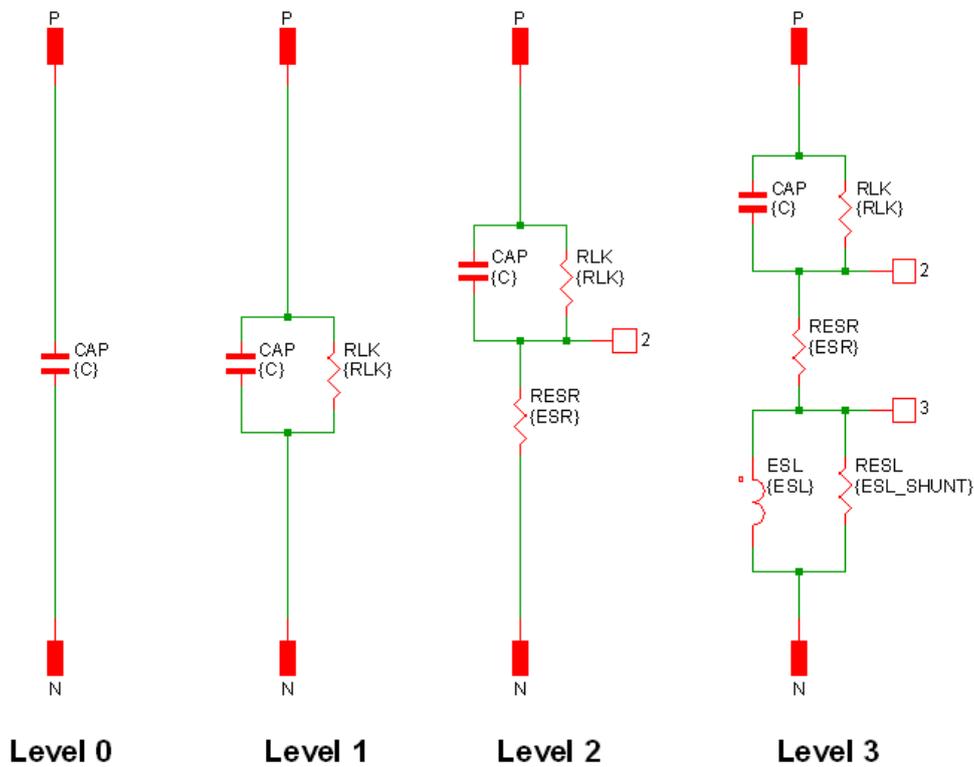
Label	Levels	Units	Description
Capacitance	0,1,2,3	%	3-sigma tolerance for the capacitance value
Leakage resistance	1,2,3	%	3-sigma tolerance for the leakage resistance across the capacitor
Series resistance (ESR)	2,3	%	3-sigma tolerance for the equivalent series resistance (ESR)
Series inductance (ESL)	3	%	3-sigma tolerance for the equivalent series inductance (ESL)
Probability Distribution Function	all	n/a	The choices for this function for each available parameter are the following: <ul style="list-style-type: none"> <li>• None The parameter has no tolerance and the value is fixed for Monte Carlo simulations.</li> <li>• Gauss The parameter has a Gaussian distribution with the specified 3 sigma tolerance. Can return values greater than or less than the 3 sigma tolerance.</li> <li>• Truncated Gauss As with Gauss but truncates all values outside the +/- 3 sigma tolerance.</li> <li>• Uniform The parameter has a Uniform distribution within the specified 3 sigma tolerance.</li> <li>• Worst Case The parameter has random values at the nominal value +/- the specified 3 sigma tolerance.</li> </ul>

## Multi-Level Capacitor Model Levels

The Multi-Level capacitor has four levels: 0, 1, 2, and 3. As the model level increases, additional parasitic circuit elements are added to the model.

The diagram below shows the circuit used for each of the four levels:

## Multi-level Capacitor Models



The following table summarises the model parameters:

Parameter	Levels	Description
CAP	all	Linear capacitance
RLK	1,2,3	Leakage resistance
RESR	2,3	Equivalent series resistance
LESL	3	Equivalent series inductance
RESL	3	ESL shunt resistance

### Quantity Implementation - SIMPLIS Mode

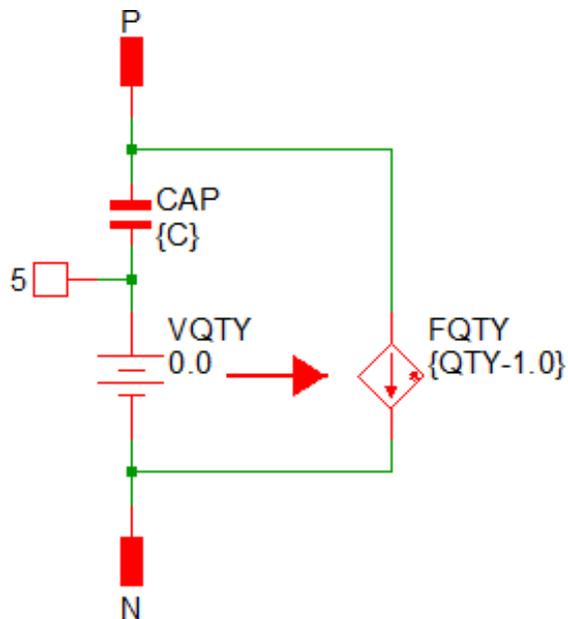
The Multi-Level Capacitor model has two quantity parameters, USE\_QTY and QTY, which specify the number of capacitors in parallel.

Configuring these parameters minimizes the number of reactive circuit elements in the model and, therefore, provides a maximum simulation speed.

The implementation of the quantity parameter uses a "DC Transformer" technique where the capacitor's terminal current is multiplied by a constant using a Current Controlled Current Source (CCCS). This technique effectively divides all resistance and inductance values by the quantity parameter (QTY) and

multiplies the capacitance by that same value. This circuitry is added if the USE\_QTY parameter is set to 1.

A schematic of the quantity implementation for a Level 0 Multi-Level Capacitor is shown below:



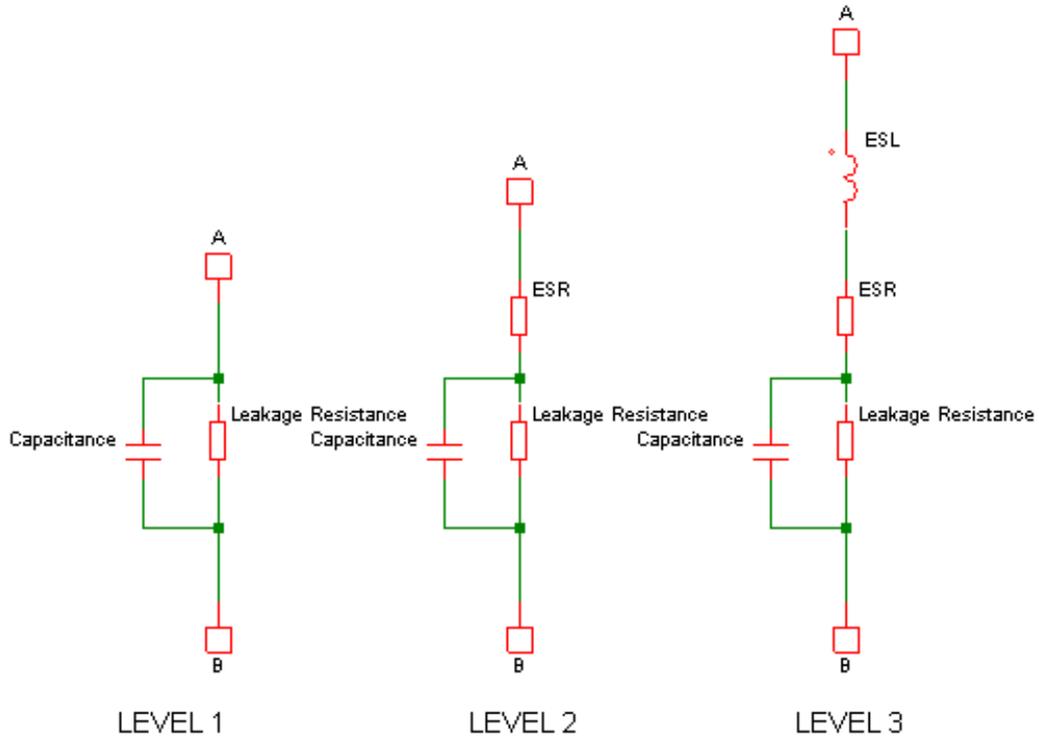
## Quantity Implementation - SIMetrix Mode

In SIMetrix, the quantity parameter is implemented with the "M" parameter which is common to the primitive SPICE components. This improves SIMetrix simulation speed and correctly scales resistances for noise analysis.

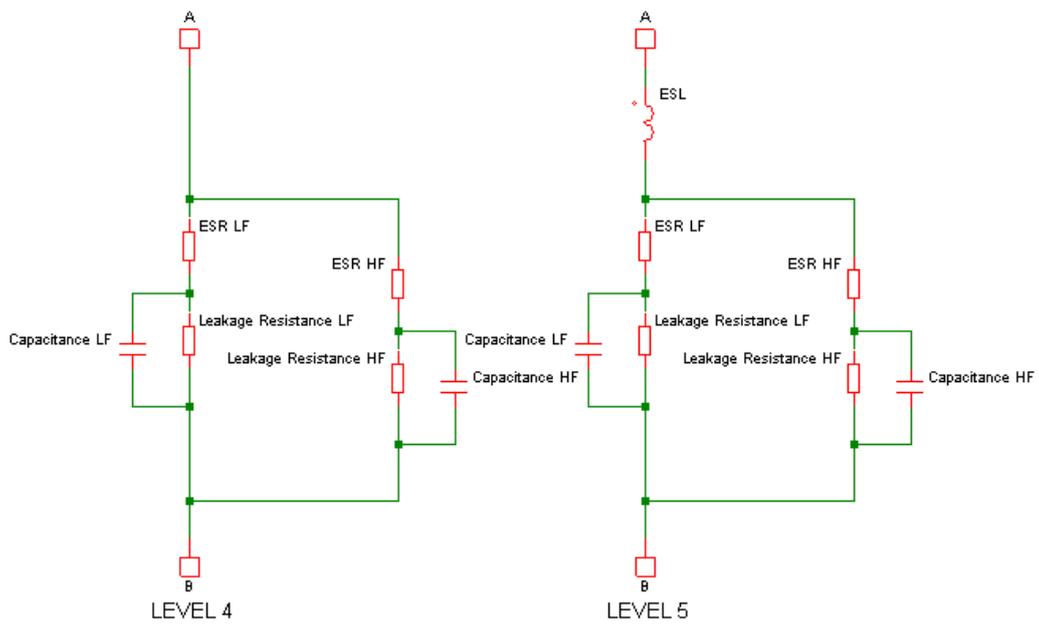
### 8.7.11 Electrolytic Capacitors

The electrolytic capacitor model has been superseded by the [“Multi-level Capacitor”](#) on page 155. The electrolytic capacitor is still available from the parts selector at locations Passives -> Capacitors, Simple (Legacy) and Passives -> Capacitors, Detailed (Legacy)

Electrolytic capacitor models are shown below:



Simple Model

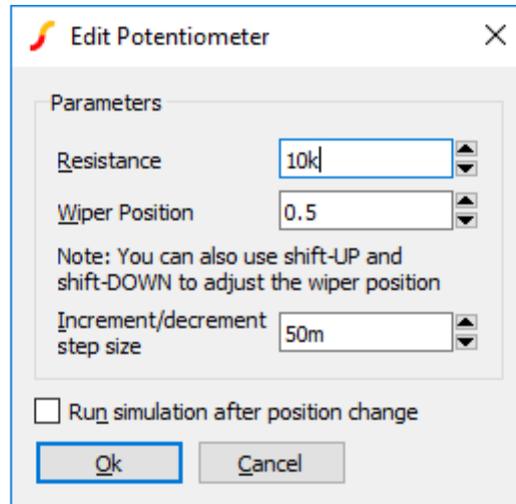


Detailed Model

Passives -> Capacitors, Simple (Legacy)

### 8.7.12 Potentiometer

The potentiometer may be used in both SIMetrix and SIMPLIS modes. To place, select the menu **Place | Passives | Potentiometer**. This device can be edited in the usual manner with **F7/Edit Part...** popup. This will display:



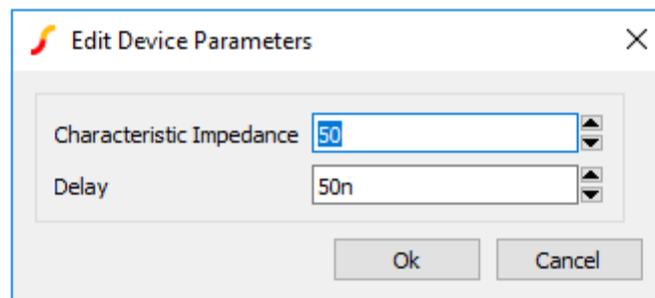
Enter Resistance and Wiper position as required.

Check Run simulation after position change if you wish a new simulation to be run immediately after the wiper position changes.

The potentiometer's wiper position may also be altered using the shift-up and shift-down keys while the device is selected. Edit Inc/dec step size to alter the step size used for this feature.

### 8.7.13 Lossless Transmission Line

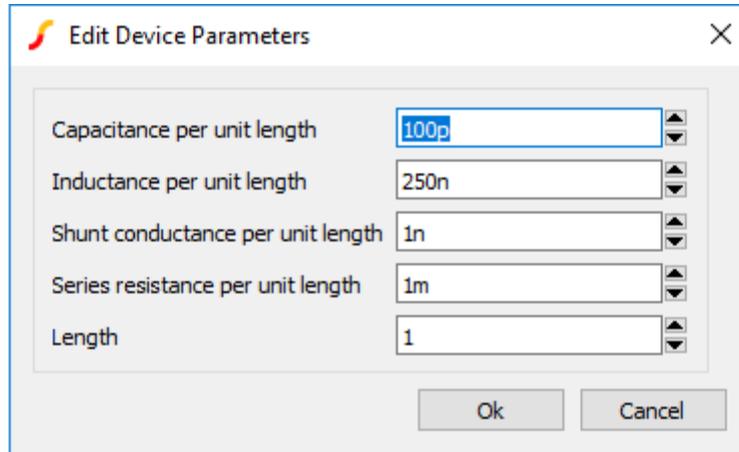
Select from menu **Place | Passives | Trans. Line (Lossless)** or press hot key 'T'. Editing in the usual way will display:



Enter the Characteristic Impedance ( $Z_0$ ) and Delay as indicated.

### 8.7.14 Lossy Transmission Line

Select from menu **Place | Passives | Trans. Line (Lossy RLGC)**. Editing in the usual way will display:



Lossy lines must be defined in terms of their per unit length impedance characteristics.

The lossy line model is implemented in a subcircuit that uses the Laplace transfer device. See [“Laplace Transfer Function”](#) on page 198 for details.

### 8.7.15 Fixed Voltage and Current Sources

See [“Circuit Stimulus”](#) on page 129.

### 8.7.16 Controlled Sources

There are four types which can be found under menu **Place | Controlled Sources**:

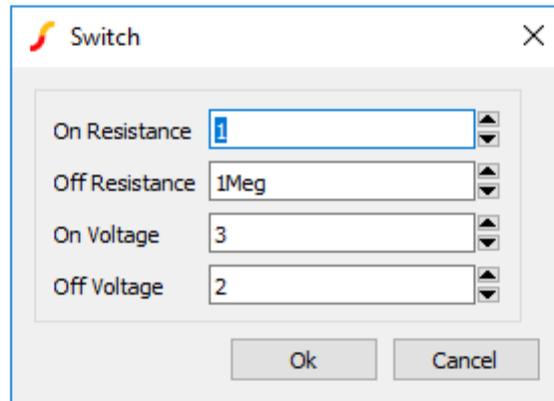
- Voltage controlled voltage source or VCVS
- Voltage controlled current source or VCCS
- Current controlled voltage source or CCVS
- Current controlled current source or CCCS

These have a variety of uses. A VCVS can implement an ideal opamp; current controlled devices can monitor current; voltage controlled devices can convert a differential signal to single ended.

They require just one value to define them which is their gain. Edit value in the usual way and you will be presented with a dialog similar to that used for resistors, capacitors and inductors but without the Parameters... button.

### 8.7.17 Voltage Controlled Switch

This is essentially a voltage controlled resistor with two terminals for the resistance and two control terminals. Place one on the schematic with **Place | Analog Functions | Switch**. Editing using **F7** or equivalent menu displays:



```

If On Voltage > Off Voltage
  If control voltage > On Voltage
    Resistance = On Resistance
  else if control voltage < Off Voltage
    Resistance = Off Resistance

```

```

If Off Voltage > On Voltage
  If control voltage > Off Voltage
    Resistance = Off Resistance
  else if control voltage < On Voltage
    Resistance = On Resistance

```

If the control voltage lies between the **On Voltage** and **Off Voltage** the resistance will be somewhere between the on and off resistances using a law that assures a smooth transition between the on and off states. Refer to *Simulator Reference Manual/Analog Device Reference/Voltage Controlled Switch*

### 8.7.18 Switch with Hysteresis

An alternative switch device is available which abruptly switches between states rather than following a continuous V-I characteristic. This device can be used with both SIMetrix and SIMPLIS although the behaviour is slightly different in each. The switching thresholds are governed by an hysteresis law and, when used with the SIMetrix simulator, the state change is controlled to occur over a defined time period which can be edited.

This device can be placed on a schematic with the menu **Place | Analog Functions | Switch with Hysteresis**.

Parameters are:

Parameter	Description
Off Resistance	Switch resistance in OFF state.
On Resistance	Switch resistance in ON state.
Threshold	Average threshold. Switches to on state at this value plus half the hysteresis. Switch to off state at this value less half the hysteresis.
Hysteresis	Difference between upper and lower thresholds.

Parameter	Description
Switching Time	Time switch takes to switch on. Note that this is the total time from the point at which the switch starts to switch on to the point when it is fully switched on.
Initial condition	Sets the initial state of the switch at the start of the simulation.

If you need to control switch on times and switch off times independently, you can use the delayed switch model. See [“Delayed Switch” on page 164](#).

### 8.7.19 Delayed Switch

Implements a voltage-controlled switch with defined on and off delay. This model can be used to implement relays. Switch action is similar to the Switch with Hysteresis described above. The delayed switch model has all the functionality of the hysteresis switch with the addition of time delay parameters and separate parameters for switch on and switch off times. Unlike the hysteresis switch, this device does not have an exact SIMPLIS equivalent.

This device can be placed on a schematic with the menu **Place | Analog Functions | Delayed Switch**.

Parameters are:

Parameter	Description
Off Resistance	Switch resistance in OFF state
On Resistance	Switch resistance in ON state
Threshold Low	Switch switches off when control voltage drops below this threshold
Threshold High	Switch switches on when control voltage rises above this threshold
On Delay	Delay between high threshold being reached and switch starting to switch on
Off Delay	Delay between low threshold being reached and switch starting to switch off
Switching Time (On)	Total time switch takes to switch on
Switching Time (Off)	Total time switch takes to switch off

Older versions of this model did not include the switching time parameters. If you wish to update a delayed switch already placed on a schematic to include this parameter, use the **Edit/Add Properties** menu to change the PARAM\_MODEL\_NAME property to delayed\_switch\_V4.

### 8.7.20 Parameterised Opamp

Implements an operational amplifier and is available from menu **Place | Analog Functions | Parameterised Opamp**. It is defined by the parameters listed below.

Parameter	units	Description
Offset Voltage	V	Input voltage needed for zero output
Bias Current	A	Average of input currents when out=0

Parameter	units	Description
Offset Current	A	Difference between input currents when out=0
Open-loop gain		$V_{out}/(V_{in+} - V_{in-})$ (Simple ratio - not dB)
Gain-bandwidth	Hz	Gain x bandwidth product
Pos. Slew Rate	V/s	Slew rate for positive output transistions
Neg. Slew Rate	V/s	Slew rate for negative output transistions(SIMPLIS only. In SIMetrix the negative slew rate is the same as the positive slew rate)
CMRR		Common mode rejection ratio. Expressed as straight ratio - not in dB
PSRR		Power supply rejection ratio. Expressed as straight ratio - not in dB
Input Resistance	$\Omega$	Differential input resistance
Output Res.	$\Omega$	Series output resistance. For SIMetrix this interacts with the quiescent current; The output resistance must satisfy: $R_{out} > 0.0129/IQ$ Where IQ is the Quiescent current. If the above is not satisfied, there is a high risk that the model will not converge. This limitation is a consequence of the way the output stage is implemented. The limitation does not apply to SIMPLIS
Quiescent Curr.	A	Supply current with no load. This must satisfy the relation shown in Output Res. above
Headroom Pos.	V	Difference between positive supply voltage and maximum output voltage
Headroom Neg.	V	Difference between minimum output voltage and negative supply rail.
Offset V. (Statistical)	V	For Monte-Carlo analysis only. Specifies the 1-sigma offset voltage tolerance. (SIMetrix only)

### 8.7.21 Parameterised Opto-coupler

Implements a 2-in 2-out optically isolated coupler. This is available from menu **Place | Analog Functions | Parameterised Opto-coupler**. It is defined by just two parameters described in the following table:

Parameter	Description
Current transfer ratio	Ratio between output current and input current
Roll-off frequency	-3dB point

### 8.7.22 Parameterised Comparator

Implements a simple differential comparator. This is available from menu **Place | Analog Functions | Parameterised Comparator**. Its parameters are defined in the following table:

Parameter	Description
Input Resistance	Differential input resistance.
Output Resistance	Series output resistance.
Hysteresis	Difference between switching thresholds. The output will switch from low-high when the differential input voltage rises above half the hysteresis. The output will switch from high-low when the differential input voltage falls below half the hysteresis.
Output Low Voltage	Unloaded output voltage in low state.
Output High Voltage	Unloaded output voltage in high state.
Delay	Delay between threshold crossing and start of the output changing state.
Rise/Fall Time	Output rise and fall time.

### 8.7.23 VCO

Implements a simple voltage controlled oscillator with a digital output. You can place a VCO on the schematic using menu **Place | Digital Generic | VCO (Analog in, digital out)**. Its parameters are:

Parameter	Description
Frequency at VC=0	Output frequency for a control voltage of zero
Gain Hz/V	Change in frequency vs change in input voltage

### 8.7.24 Verilog-A Library

If you have a SIMetrix Pro or SIMetrix Elite you may also use one of the Verilog-A implemented devices available under the **Place | Analog Functions | Verilog-A Library**. These devices are defined using the Verilog-A language. The Verilog-A code for these devices may be found in the support\valibrary directory under the SIMetrix root.

Currently there are 4 Verilog-A library devices as described in the following paragraphs.

#### Voltage Controlled Delay

Implements a variable analog delay.

This device has three parameters as defined in the table below. Double click the device to edit its parameters.

Parameter	Description
Max delay	The maximum delay that the device may provide in seconds.

Parameter	Description
Voltage for minimum delay	Input voltage for minimum delay (i.e. zero)
Voltage for maximum delay	Input voltage for maximum delay. The delay when between the minimum and maximum voltages will be calculated following a linear characteristic.

## Fixed Delay

Implements a fixed analog delay

This device has just a single parameter defining its delay in seconds. Double click the device to edit.

## Sinewave VCO

Implements a sinewave voltage controlled oscillator. This has four parameters as defined below:

Parameter	Description
Amplitude	Peak amplitude of sine wave.
Centre Frequency	Frequency for zero volts input.
Gain Hz/Volt	Change in frequency for each volt change in the input.
Minimum steps per cycle	Minimum number of time points per cycle. The simulator will force time points to ensure that each cycle has at least the number specified.

## Pulse Width Modulator

Implements a voltage controlled pulse width. Defined by 6 parameters as follows:

Parameter	Description
Frequency	Frequency of pulse
Input low voltage	Voltage for zero duty cycle
Input high voltage	Voltage for maximum duty cycle
Output low voltage	Low voltage of output pulse
Output high voltage	High voltage of output pulse
Maximum duty cycle	Maximum duty cycle. This may not be higher than 0.999

### 8.7.25 ISO 16750-2 and 7637-2 Waveform Sources

A range of waveform sources designed for testing automotive electronics is available for SIMetrix simulations. These parts are implemented in Verilog-A and require a SIMetrix Pro or SIMetrix Elite license.

## Description of Sources Available

Reference	Description
ISO7637 Pulse 1	High voltage negative-going transient to test disconnection from inductive loads
ISO7637 Pulse 2a	High voltage positive-going transient due to sudden interruption of currents in a device connected in parallel with the device under test due to the inductance of the wiring harness
ISO7637 Pulse 2b	Transients due to motors acting as generators after the ignition is switched off
ISO7637 Pulse 3a	Negative-going high frequency high voltage transients, which occur as a result of the switching processes
ISO7637 Pulse 3b	Positive-going high frequency high voltage transients, which occur as a result of the switching processes
ISO16750 4.4	Swept frequency source from 50Hz to 25kHz representing the residual alternating current on the direct current supply
ISO16750 4.5	Slow decrease and increase of supply representing battery discharge and charge
ISO16750 4.6.1	Negative going pulse arising from the melting of a fuse element in another circuit
ISO16750 4.6.2	Succession of increasing negative-going pulses to test reset behaviour
ISO16750 4.6.3	Waveform representing supply voltage during and after electric starter cranking
ISO16750 4.6.4 Test A	Positive going high voltage transient. Simulates the load dump transient occurring in the event of a discharged battery being disconnected while the alternator is generating charging current with other loads remaining on the alternator circuit at this moment
ISO16750 4.6.4 Test B	As 4.6.4 Test A but transient is clamped

## Place and Edit Sources

All sources are available from the Part Selector “[Part Selector](#)” on page 124. Open the Part Selector, select menu **View | Show Part Selector** then navigate to Sources -> ISO 16750/7637 Source. Alternatively, open the search dialog and type "ISO" as the search term. Select the desired source; note that each source has both 12V and 24V versions.

All parts have at least one editable parameter. The definition of the parameter is defined in the relevant ISO standard document.

### 8.7.26 Generic ADCs and DACs

Generic data conversion devices are available from the menus **Place | Digital Generic | ADC...** and **Place | Digital Generic | DAC...**

**Define A-D Converter**

Number of bits: 8

**Timings**

Convert time: 1u

Max conversion rate: 2Meg

**Input**

Offset/volts: 0

Range/volts: 5

Ok Cancel

**Define D-A Converter**

Number of bits: 8

**Timings**

Output slew time: 100n

**Output**

Offset/volts: 0

Range/volts: 5

Ok Cancel

These devices are implemented using the simulator's ADC and DAC models. For details of these refer to *Simulator Reference Manual/Digital Mixed Signal Device Reference/Analog-Digital Converter* and *Simulator Reference Manual/Digital Mixed Signal Device Reference/Digital-Analog Converter*

The controls in these boxes are explained below.

### Number of bits

Resolution of converter. Values from 1 to 32

### Convert time (ADC)

Time from start convert active (rising edge) to data becoming available

### Max conversion rate (ADC)

Max frequency of start convert. Period (1/f) must be less than or equal to convert time.

### Output slew time

Whenever the input code changes, the output is set on a trajectory to reach the target value in the time specified by this value.

### Offset voltage

Self-explanatory.

### Range

Full scale range in volts.

## 8.7.27 Generic Digital Devices

A number of generic digital devices are provided on the **Place | Digital Generic** menu. Each will automatically create a symbol using a basic spec. provided by your entries to a dialog box. Functions provided are, counter, shift register, AND, OR, NAND and NOR gates, and bus register.

## 8.7.28 Arbitrary Non-linear Passive Devices

Each of these will place a part which looks exactly like its linear counterpart. The difference is that when you try and edit its value with **F7** or menu **Edit Part...** you will be prompted to enter an expression. In the case of the resistor and capacitor, this relates its value to the applied voltage and for inductor the expression relates its inductance to its current. For resistors and capacitors, the terminal voltage is referred in the equation as 'V(N1)' and for inductors the device's current is referred to as 'I(V1)'.

## 8.8 Creating Models

### 8.8.1 Overview

SIMetrix provides a soft recovery diode model for use in power electronics circuits. As this model is not a SPICE standard, there are no models available from device manufacturers or other sources. So, we therefore also developed a soft recovery diode "parameter extractor" that allows the creation of soft recovery diode models from data sheet values.

The parameter extraction tool works directly within the schematic environment and may be used in a similar manner to other parameterised devices such as the parameterised opamp. However, there is also an option to save a particular model to the device library and so making it available as a standard part.

In addition, SIMetrix offers a tool for creating a saturating inductor model from a data sheet Inductance vs Current curve.

## 8.8.2 Creating Soft Recovery Diode Models

1. Select menu **Place | Create Model | Soft Recovery Diode...** . You will see this dialog box:

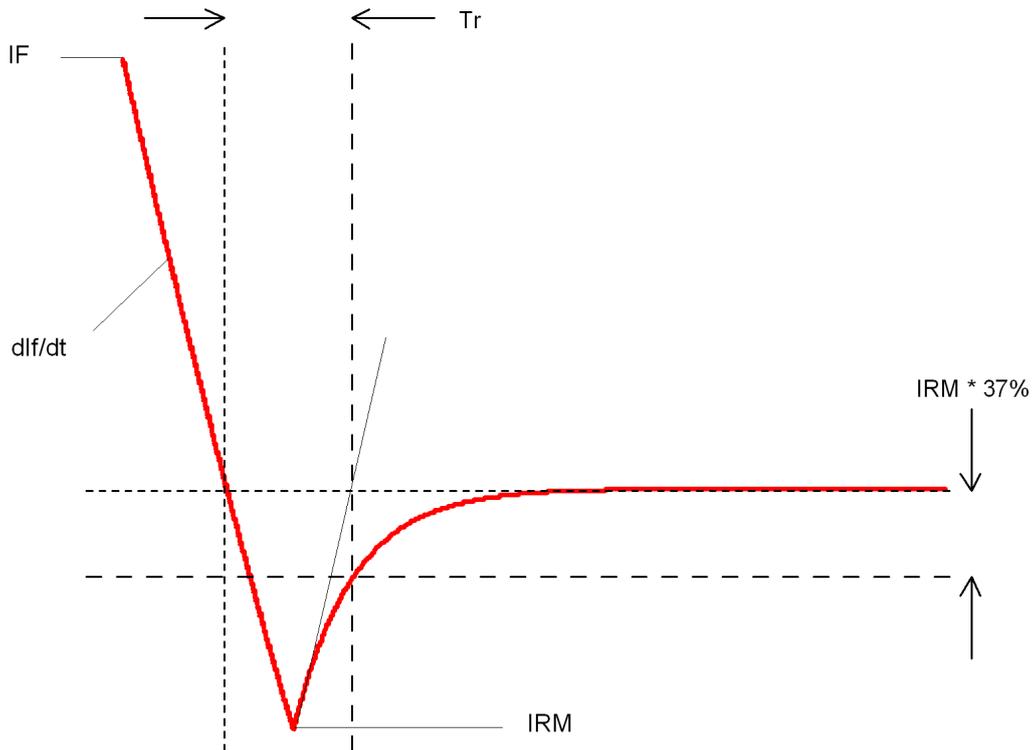
2. Enter the required specification in the DC Forward bias spec, Reverse recovery specification and Capacitance sections. See below for technical details of these specifications.
3. Select **Save to schematic symbol** if you wish to store the specification and model parameters on the schematic symbol. This will allow to you to modify the specification later. If you select **Save to model library**, then the definition will be written to a library file and installed in the parts library. This will make the new model available as a standard part, but you will not be able to subsequently modify it other than by re-entering the specification manually. If you choose this option, you must specify a device name in the box below.
4. Click **Ok** to place diode on the schematic. If you selected **Save to model library**, the model file for the device will also be created at this point. The file will be saved in your user models directory. This is located at “My Documents\SIMetrix\Models”.

### Soft Recovery Diode Specification

The parameter extractor allows the specification of three important characteristics of the diode. These are the DC forward bias voltage, reverse recovery and capacitance. Currently, reverse leakage and breakdown characteristics are not modelled.

To specify the forward bias characteristics, simply enter the coordinates of two points on the graph showing forward drop versus diode current which is found in most data sheets. You should choose values at the extremes. The low current value will essentially determine the value of the IS parameter while the high current value defines the series resistance of the device.

The reverse recovery characteristics are explained in the following diagram.



The values quoted in data sheets vary between manufacturers. The value given for  $T_r$  is sometimes taken from the reverse peak rather than the zero crossing. If this is the case you can calculate the time from the zero crossing to the reverse peak using the values for  $IRM$  and  $dI_f/dt$  and so arrive at the value of  $T_r$  as shown above.

Some data sheets do not give the value of  $IRM$ . In these cases the best that can be done is to enter an intelligent guess.

Capacitance is the measured value at zero bias. Unfortunately this is not always quoted in data sheets in which case you can either enter zero (which may speed simulation times) or enter an estimated value. Of course an alternative would be to measure the capacitance of an actual device.

### Notes of Soft Recovery Diode Model

The soft recovery diode does not use the standard SPICE model but a model based on work at the University of Washington. Full details of the model can be found in the Simulator Reference Manual.

#### 8.8.3 Creating Power Inductor Models

The power inductor model models these effects:

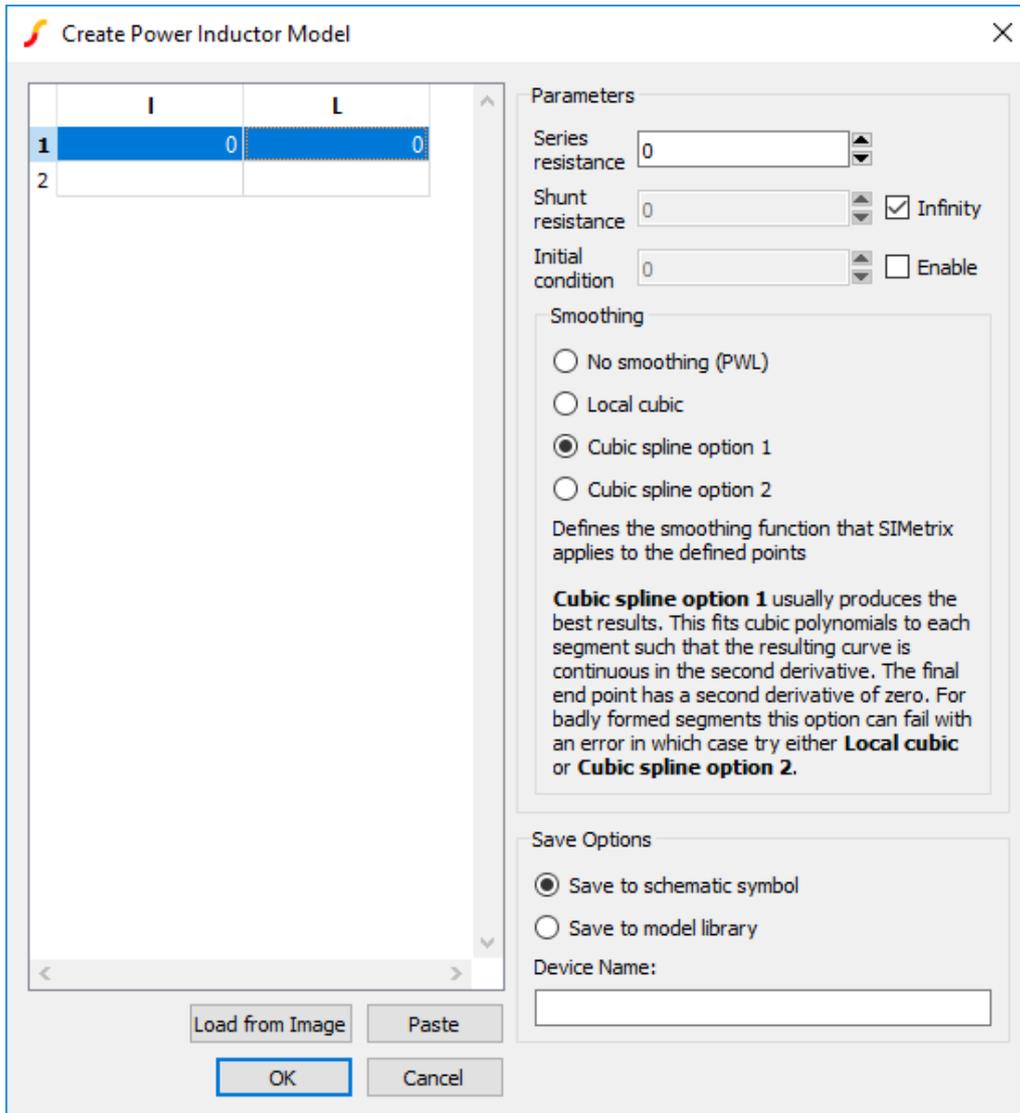
- Saturation
- Series and shunt resistance

Saturation effects are entered as a lookup table and may optionally be defined by digitising a graph image.

For technical details about the underlying simulation model please refer to *Simulator Reference Manual/Analog Device Reference/Inductor (Table lookup)*

## Power Inductor Specification

To create a power inductor model, select menu **Place | Create Model | Power Inductor**. This will show the following dialog box:



The various elements of the dialog box are explained below

Dialog element	Description								
IL Table	Table of values that defines the inductance vs current. The table should only define a single quadrant; the power inductor model implementation will automatically generate inductance values for negative current. You can enter these values manually, paste the values from an external source or you can digitise a graph image. To digitise a graph image, click on Load From Image. This will invoke the data sheet curve digitiser. See <a href="#">“Digitising Data Sheet Curves” on page 175</a> . When complete the data from the data sheet curve digitiser will be entered into the IL Table.								
Series resistance	Series resistance of the inductor								
Shunt resistance	Parallel resistance applied across the inductor								
Initial condition	If enabled the inductor will behave like a current source during the DC operating point calculation with the current specified here. If disabled the inductor will be a short circuit during the operating point calculation								
Smoothing	<p>Sets the smoothing function of the inductor model. The points entered in the IL Table may optionally be smoothed using a mathematical function. Options are:</p> <table border="0"> <tbody> <tr> <td style="padding-left: 20px;">No smoothing</td> <td>The points defined by the table are joined by straight lines</td> </tr> <tr> <td style="padding-left: 20px;">Local cubic</td> <td>Fits a cubic between all points with derivative average of slope either side of points. This is not continuous in the second derivative</td> </tr> <tr> <td style="padding-left: 20px;">Cubic spline option 1</td> <td>Cubic spline with boundary conditions: lower <math>dL/di=0</math>, upper <math>d^2L/di^2 = 0</math>. Continuous in second derivative</td> </tr> <tr> <td style="padding-left: 20px;">Cubic spline option 2</td> <td>Cubic spline with boundary conditions: lower <math>dL/di=0</math>, upper <math>dl/di = \text{slope of final segment}</math>. Continuous in second derivative</td> </tr> </tbody> </table> <p>Cubic spline option 1 usually produces the best results but can sometimes fail during the simulation with error message:</p> <pre>***ERROR*** instance &lt;ref&gt;: Cannot fit spline to table using SMOOTH=2 strategy. Try using SMOOTH=1 or SMOOTH=3"</pre> <p>In this case try either Cubic spline option 2 or Local cubic</p>	No smoothing	The points defined by the table are joined by straight lines	Local cubic	Fits a cubic between all points with derivative average of slope either side of points. This is not continuous in the second derivative	Cubic spline option 1	Cubic spline with boundary conditions: lower $dL/di=0$ , upper $d^2L/di^2 = 0$ . Continuous in second derivative	Cubic spline option 2	Cubic spline with boundary conditions: lower $dL/di=0$ , upper $dl/di = \text{slope of final segment}$ . Continuous in second derivative
No smoothing	The points defined by the table are joined by straight lines								
Local cubic	Fits a cubic between all points with derivative average of slope either side of points. This is not continuous in the second derivative								
Cubic spline option 1	Cubic spline with boundary conditions: lower $dL/di=0$ , upper $d^2L/di^2 = 0$ . Continuous in second derivative								
Cubic spline option 2	Cubic spline with boundary conditions: lower $dL/di=0$ , upper $dl/di = \text{slope of final segment}$ . Continuous in second derivative								

Dialog element	Description
Save options	Use <b>Save to schematic symbol</b> to save model parameters directly on the schematic symbol. Use <b>Save to model library</b> and enter a model name to save a model in the model library. This will be available to be placed in other schematics using the menu <b>Place   From Model Library....</b> Note that you will not be able to make subsequent edits to the model if using this option.

### 8.8.4 Digitising Data Sheet Curves

The graph data extraction tool can be used to extract values from datasheet curves that can be imported for use within models in Pro and Elite versions of SIMetrix and SIMetrix/SIMPLIS. One such use is in the [“Power Inductor Model” on page 172](#). It is also available as a free-standing tool to digitise any graph found in an image file. The free-standing version can be initiated using the schematic menu **Tools | Digitise Data Sheet Curve** or directly with the script command *GraphImageCapture*.

To use the tool, first an image of a graph is required. These graphs will often come from manufacturer data sheets or websites. In some cases you may be able to copy the images directly from the website or you may need to use a screen capture from a PDF image and save that image to file. The tool supports images in PNG, JPG, BMP and GIF formats, where PNG is the preferred format.

#### Image Requirements

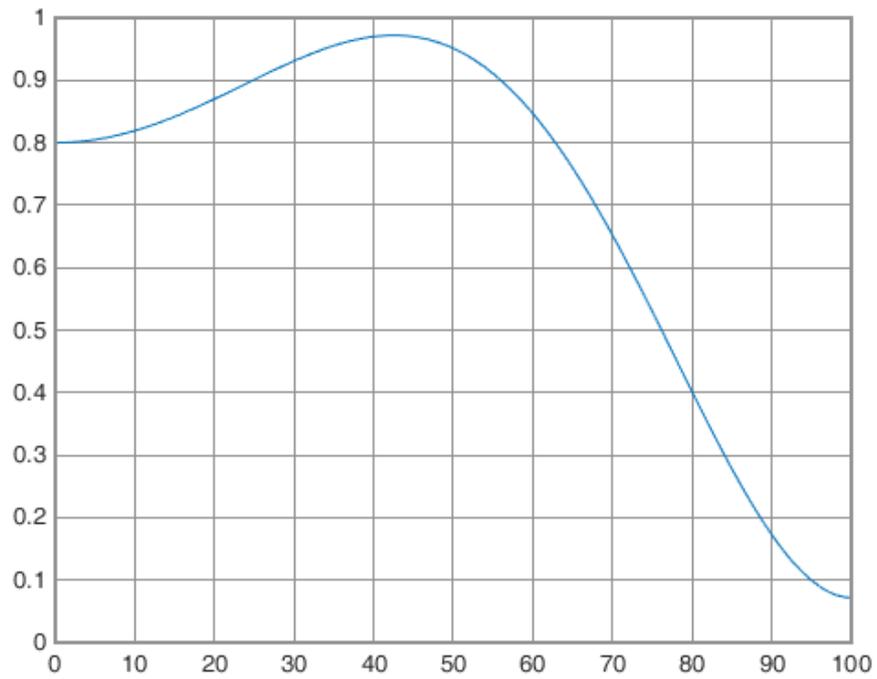
The tool is designed to handle many different layouts of graphs and quality of images. The tool can accept graphs with:

- Single or multiple plotted data series
- Linear or logarithmic axes
- Grid marker lines
- Poor quality text (text labels are not required for the tool to operate)
- Graphs on light or dark backgrounds

The tool will perform poorly and may fail to give reasonable values for graphs that are at an angle.

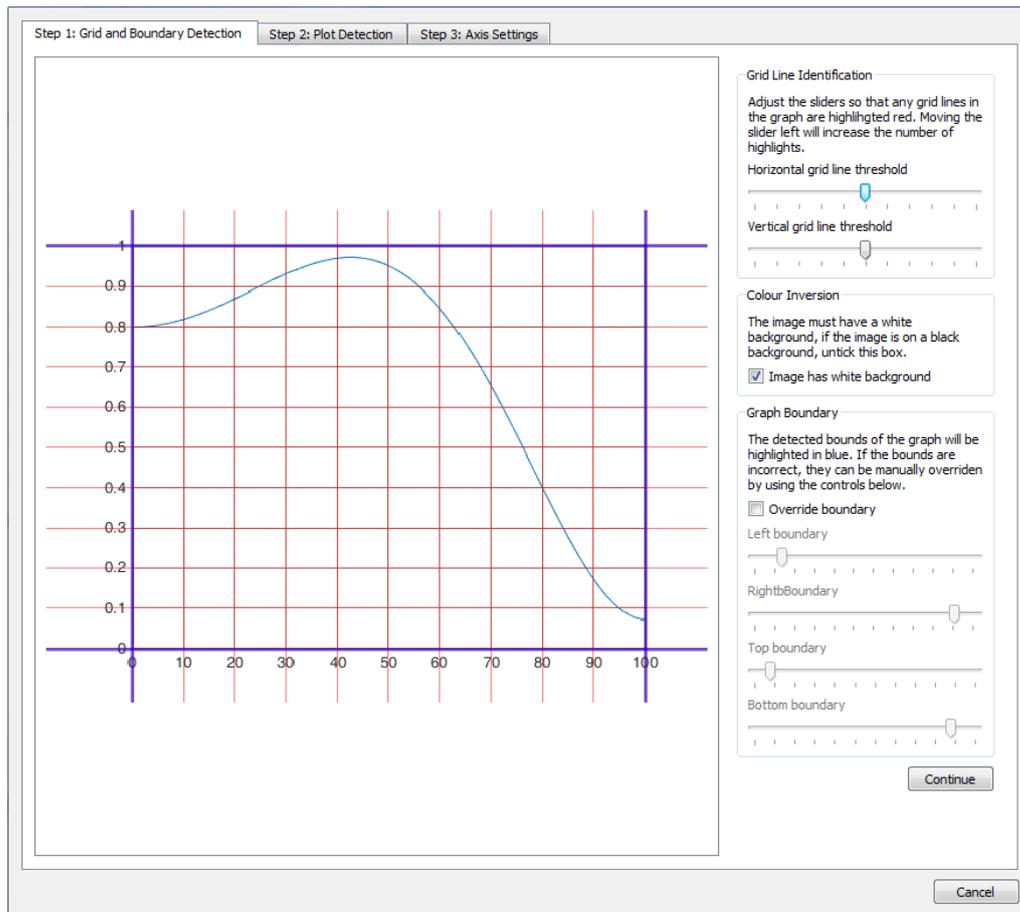
#### Step 1: Grid and Boundary Detection

The first step of the tool is to determine the boundary of the graph along with any grid lines. For this demonstration, we will use the following image:



The graph we want to capture data from, with grid lines and a single data series.

When the image is loaded, the tool will make an initial estimate of where the graph bounds are and any grid lines there might be. The bounds will be highlighted in blue, whilst the grid lines will be highlighted in red, as shown in the next figure.



View of the first step window after the example image is first loaded.

In many cases no action on the users part will be required in Step 1. Before continuing to step 2, ensure that the boundary of the graph is highlighted in blue, as many grid lines as possible are highlighted red and that if the image is on a black background, the colour inversion tick box is unchecked. Of these requirements, ensuring the bounds of the graph are correct is the most important as this is used by the final data point extraction. Getting the grid lines fully identified is not essential, this just makes the user interaction in step 2 easier. To make adjustments to the boundary and grid line identification, there are a number of sliders to the right of the window.

For *Grid Line Identification*, you can control how sensitive the selection of grid lines are in the horizontal and vertical plane. By moving the sliders the to left, their selection becomes more sensitive and a greater number of lines will be detected, whilst moving to the right will cause less grid lines to be identified. Adjusting these sensitivity bars can lead to grid lines not being detected or false detections from occurring.

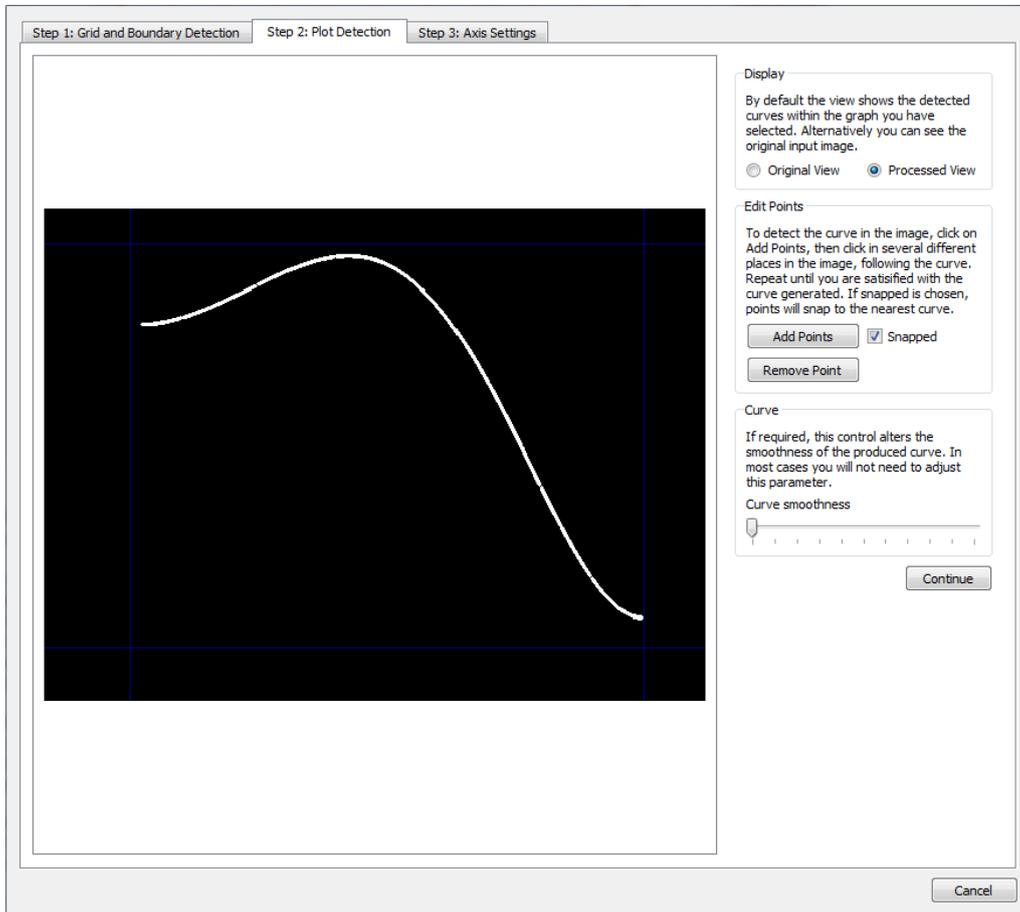
For *Graph Boundary*, you control the position of the boundary lines and consequently the region of interest within the graph. This can be useful for not only correcting issues where the boundary detection has failed, but also for restricting the part of the graph that you are interested in generating data from if you only want to extract data from part of a graph.

**Note:** If there are lines not displaying, try resizing the window horizontally as due to image compression in displaying the resulting images.

Once this step is complete, press **Continue**.

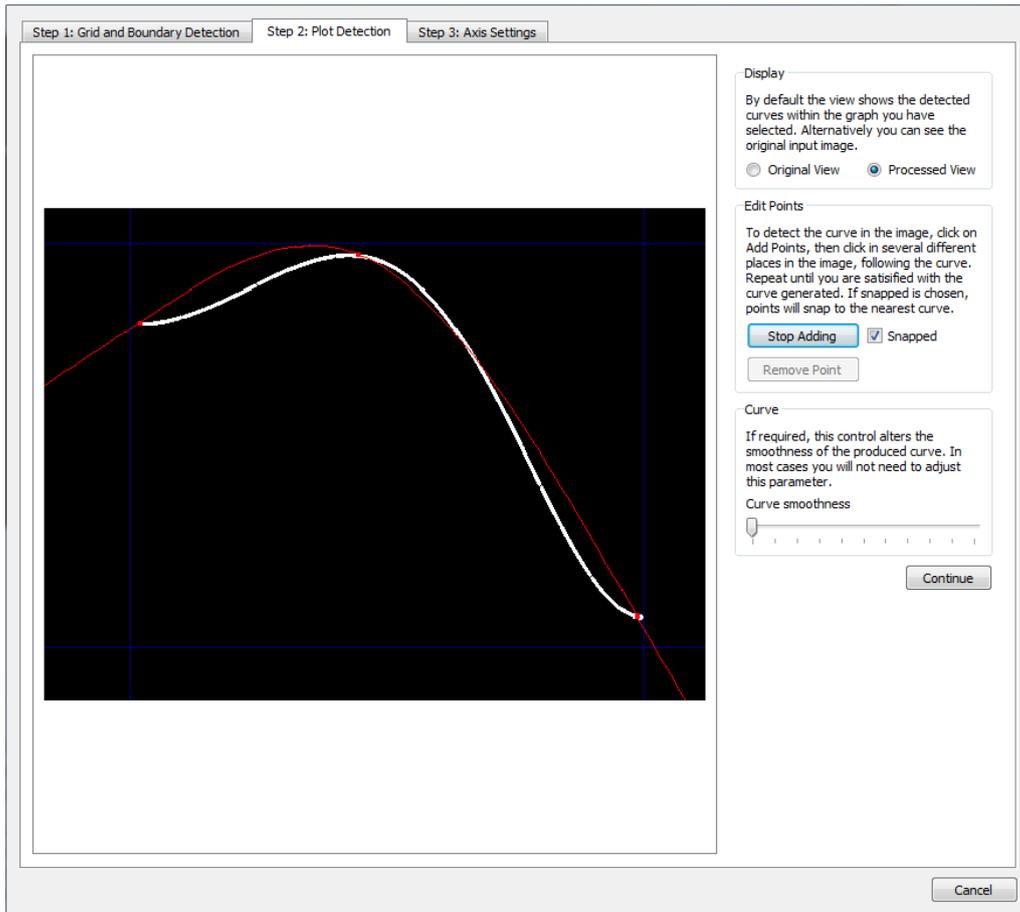
## Step 2: Plot Detection

The second step is to pick out the curve that you want to extract the data from. If any grid lines were all successfully detected and the bounds set, you will begin with a preview similar to that shown in the next figure.



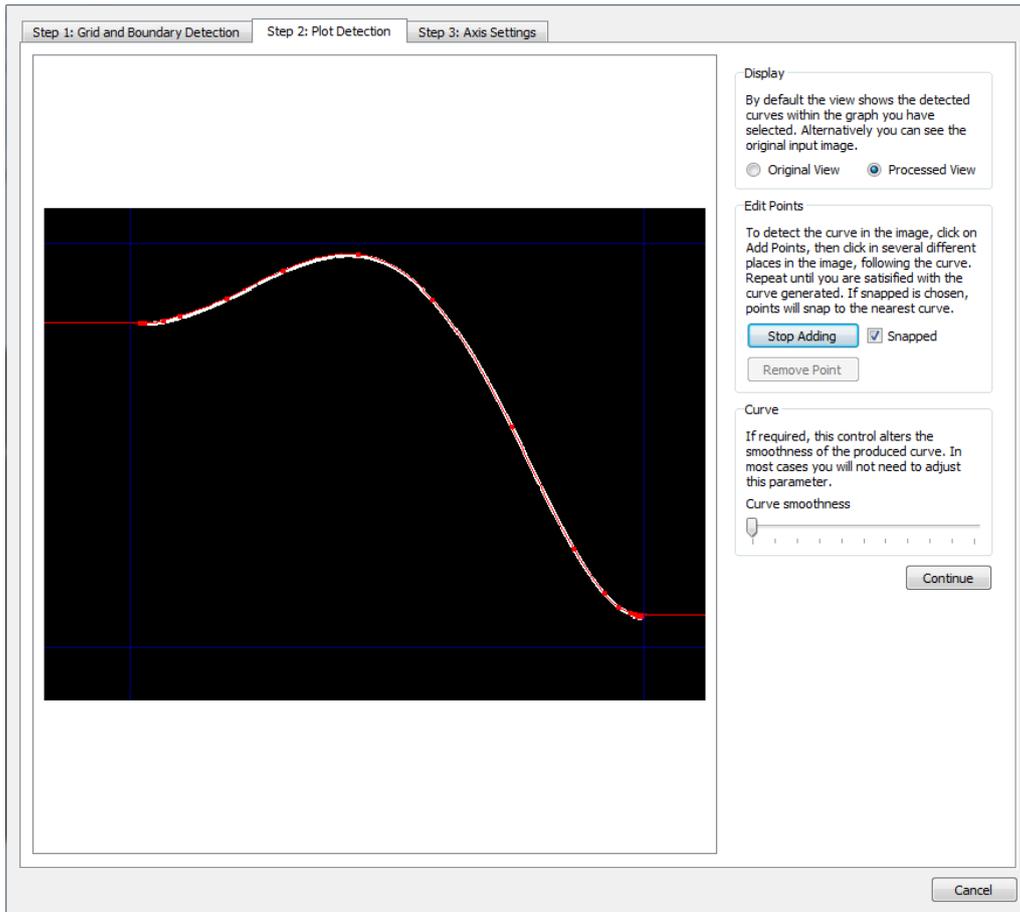
View of the second step window after completing step 1. Currently no curve has been marked out for use.

To mark out the curve to use, press the button **Add Points** and click along the line you want to use, a red square will be placed for each point added. By default the points will snap to the curve nearest where you have clicked, so you should not need to be highly precise in where you click. To disable the snapping behaviour, uncheck the **Snapped** check box. As you add points, a red line will preview the line that will be used to generate data from, as shown in the next figure. This red line will not snap to a curve, because although the solution may look obvious for full automatic-detection in the single line case presented, in cases where there are multiple lines of the image is of a lower quality, such automation would be highly error prone. Instead, after adding a few initial points, add further points in places where the red line does not match the curve you want to capture.



Line detection after three points have been added. The red line shows the line that data will be obtained from in the final step. This line does not currently match the actual curve that was on the graph shown in white, meaning further points are required to be added in places where the curves do not match.

Generally you will not require many points to be placed to obtain a red line that well resembles the curve you want to capture. As shown in the next figure, you may not need to place many points along regions of the graph with consistent behaviour, such as continuous curves or straight regions. In areas where greater precision is required, you may require several closely positioned points.



A completed line detection. Points are spaced out for the majority of the curve but are more dense at the ends of the curve due to the curve levelling out at those regions.

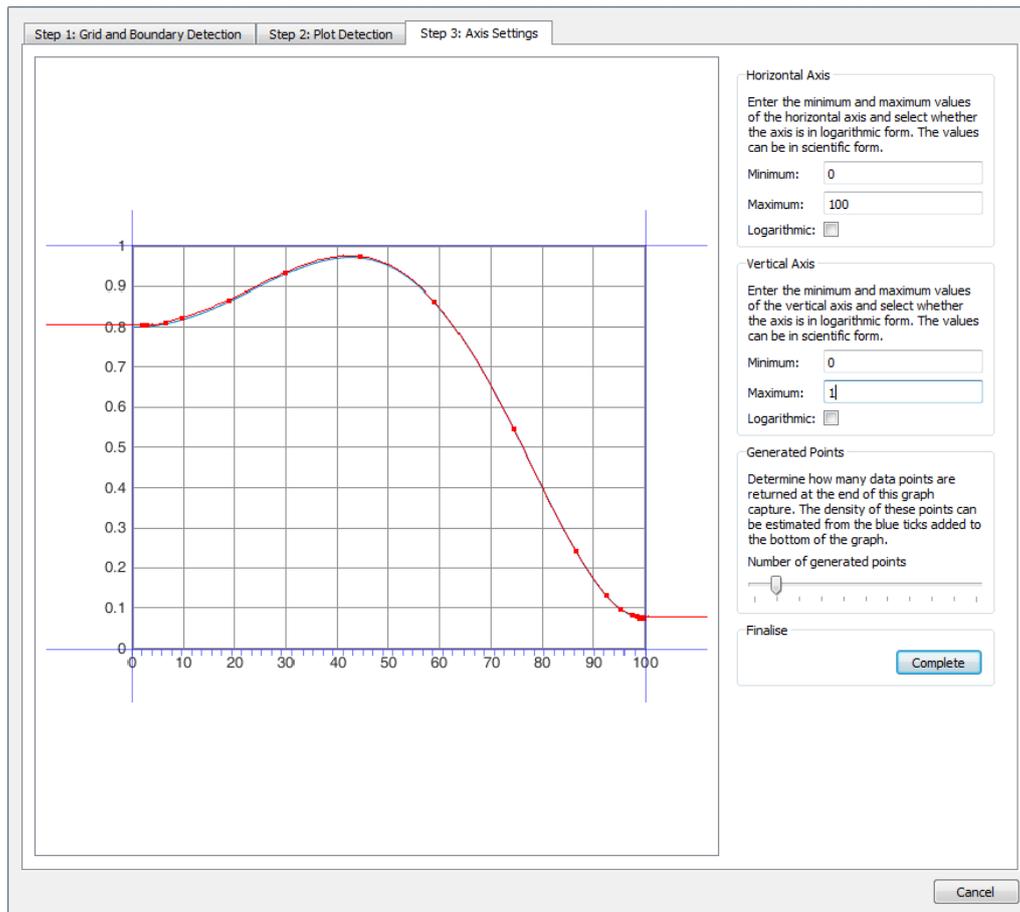
If a point is added erroneously, press the **Stop Adding** button then press the **Remove Point** button and click on the point you want to remove. The remove point behaviour only works for one point at a time. If multiple points are to be removed, the **Remove Point** button will need to be pressed each time.

Whilst the curves produced by this method should generally provide a good representation through simply adding points alone, there may be situations where the red curve generated is not smooth enough to fit those points. In these cases the **Curve smoothness** slider can be used to make the curve smoother. For most usage the slider can remain in the it's default position of fully to the left. Moving the slider fully to the right will make the curve become more smooth and with the slider fully to the right the curve will become a straight line.

When the curve detection is complete, press continue.

### Step 3: Axis Settings

Now that the bounds of the graph and a curve has been selected, the final step is to determine the values of the axes. As shown in the next figure, the final step will show the originally loaded graph image, along with the graph bounds highlighted in blue and the selected curve in red. If either the bounds or curve do not appear to be correct, you can go back to the previous stages and update them.



A completed line detection. Points are spaced out for the majority of the curve but are more dense at the ends of the curve due to the curve levelling out at those regions.

To submit values for the axes, set the minimum and maximum values for the graph bounds in the text boxes to the right of the graph image. In this example the minimum value on the horizontal axis is at 0, with maximum at 100. For the vertical we have minimum of 0 and maximum of 1. If either of the graphs are a logarithmic axis, select the logarithmic scale check box.

Finally you can determine the granularity of the points generated by adjusting the **Generated Points** slider. Moving the slider to the left increases the number of points. An indication of where the samples will be taken from can be seen by the blue ticks on the horizontal axis.

When all stages are complete, press the **Complete** button. If the button is disabled, messages will appear stating what stages have not been completed that are required. When the **Complete** button is pressed, the dialog will return vectors of data points that have been extracted from the curve.

## 8.9 Subcircuits

### 8.9.1 Overview

Subcircuits are a method of defining a circuit block which can be referenced any number of times by a single netlist line or schematic device. Subcircuits are the method used to define many device models such as op-amps. It is also the underlying mechanism of the hierarchical schematic entry system.

You don't need to know anything about subcircuits unless you wish to define your own device models, perhaps to build up a library of tested blocks for general distribution. If you just wish to enter your circuit

in a modular manner, hierarchical schematic entry is probably the more appropriate method. See [“Hierarchical Schematic Entry” on page 65](#) for details.

This section explains how to create a subcircuit from a schematic and how to reference one in netlist or schematic. For the .SUBCKT control syntax see the “Command Reference” chapter of the *Simulator Reference Manual*.

## 8.9.2 Creating a Sub-circuit from a Schematic

Subcircuits must be defined in text form as a netlist. However the schematic editor can be used to generate the netlist. To create a sub-circuit from a schematic, you need to identify which nodes are to be connected externally. This is done using the same Module Port symbol used for hierarchical schematic entry (see [“Hierarchical Schematic Entry” on page 65](#)).

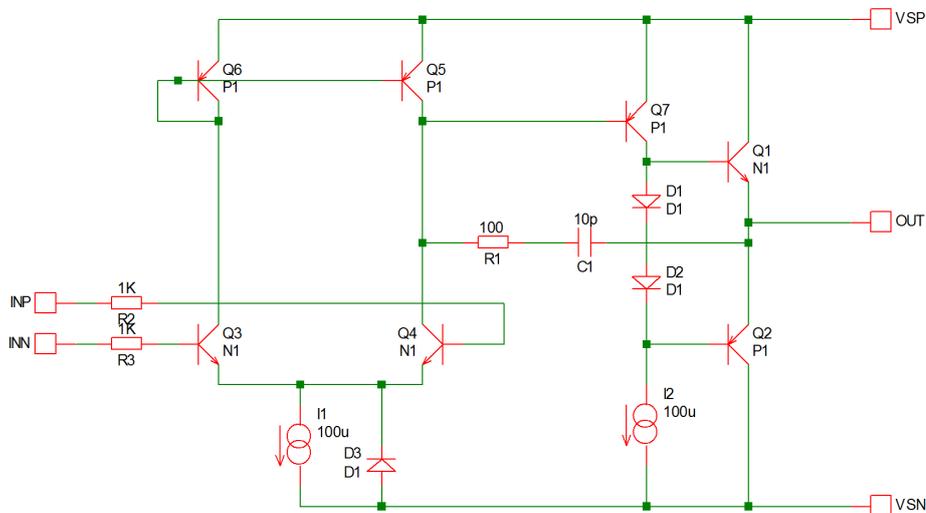
The procedure for defining a subcircuit is as follows:

1. Draw circuit using schematic editor including module port symbols to identify external connections.
2. Create netlist for circuit.

To describe the procedure, we will use an example.

### Stage 1 - Draw Schematic

This is circuit of a simple op-amp. In fact it is the circuit of our fictitious SXOA1000 op-amp used in [“Tutorial 3” on page 26](#).



The five terminal symbols, e.g.



are the connections to the outside world. This is a module port symbol which can be found in the schematic menu **Hierarchy | Place Module Port**. *Important* - do not use the normal Terminal symbol.

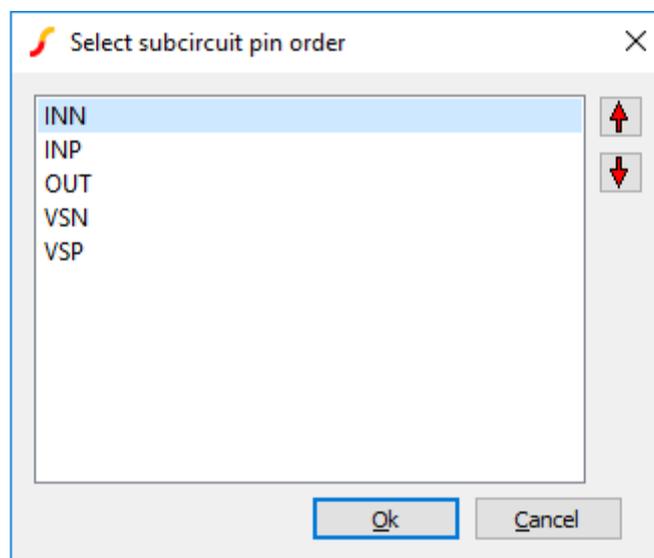
It is recommended that any model definitions are included in the subcircuit definition. This makes the subcircuit self-contained. If you have referenced models in the device library, you can import them into the schematic automatically using the schematic menu **Simulator | Import Models...** They will be placed in the simulator command window which can be opened by pressing **F11**. Alternatively you can enter them in the command window manually.

## Stage 2 - Netlist Circuit

To create a subcircuit netlist, select schematic menu **Simulator | Create Netlist as Subcircuit...**

You will be first be prompted for a subcircuit name. This name will also be used for the file name with extension .MOD.

After entering the name, you will be asked to specify the subcircuit pin order:



When you close this box, the subcircuit will be created and its text will be displayed.

### 8.9.3 Calling a Sub-circuit in a Schematic

To call a sub-circuit in a schematic, you must choose or create a symbol for it. The symbol *must* have the same number of pins and, ideally, it would also have the same pin order. In other words, the order of the nodes in the .SUBCKT line would be the same as the pin order of the symbol. The matching of .SUBCKT node order and symbol pin order is not absolutely essential, but it makes things much easier. If they are not the same there is method of overcoming the problem using the MAPPING property. This is explained in the section [“Properties” on page 87](#).

Creating symbols for the schematic is covered in [“Creating Schematic Symbols - Overview” on page 109](#). The symbol must have the following properties (see [“Properties” on page 87](#)).

Property name	Property value	Purpose
Model	X	Ensures netlist line starts with X. Identifies part as a subcircuit. Should be <i>hidden</i> and <i>protected</i>
Value	subcircuit_name	Name used to reference subcircuit definition. Can be changed by user after placing on schematic.
Ref	component_reference	E.g. U?. Automatically allocated when placing symbol on schematic.

Most symbols possess these properties anyway, the important fact is that the MODEL property must be set to X. When defining a symbol from scratch, these properties can be defined in one go in the graphical symbol editor with **Property/Pin | Add Standard Properties...**

To use the sub-circuit definition, SIMetrix must be able to find it. There are various places where it can be put and means of telling SIMetrix of the location. These are the choices.

1. Place the definition directly in the simulator command - or **F11** - window (see [“Manual Entry of Simulator Commands” on page 42](#)). If placed at that location, it will be read in unconditionally and SIMetrix will not need to search for it.
2. Put in a separate file and pull in to the schematic with .INC control (see *Simulator Reference Manual/Command Reference/.INC*) placed in simulator command (**F11**) window. As 1., this will be read in unconditionally.
3. Put in a library file and reference in schematic with .LIB control (see *Simulator Reference Manual/Command Reference/.LIB*) placed in simulator command (**F11**) window. Similar to 2. but more efficient if library has many models not used in the schematic. Only the devices required will be read in.
4. Put in a library file and install it using the procedure described in [“Full Model Installation Procedure” on page 210](#). This will make the device globally available to all schematics. You can also install it into the model library browser system. These topics are covered in [“Device Library and Parts Management” on page 209](#) and are also the subject of [“Tutorial 3” on page 26](#).

To place the device on the schematic, find the symbol in menu **Place | From Symbol Library...** and place in the normal way. After it is placed, select the device and press **Shift-F7** and enter the subcircuit's name.

If you installed the device into the model library browser system, as mentioned in choice 4 above, you will be able to place the device by pressing control-G and selecting the device from the appropriate category. The model library browser system also provides a simple to use means of overcoming the problem mentioned above that occurs if the symbol's pin order does not match the subcircuit's node order. This is explained in [“Associating Multiple Models with Symbols” on page 216](#).

## Passing Parameters

You can pass parameters to a subcircuit. This subject is covered in detail in the *Simulator Reference Manual/Simulator Devices/Subcircuits/Passing Parameters to Subcircuits*. To specify the parameters for a sub-circuit device in a schematic, you must enter the values manually using **Shift-F7**. Enter the values after the subcircuit name. E.g. suppose you wished to specify the parameters: 'FREQ=12k Q=15'. To enter these, select the sub-circuit, press shift-F7 and append the sub-circuit name with:

```
FREQ=12k Q=15
```

You can add 'params:' to emphasise where the parameters start and also for compatibility with some other simulators. E.g:

```
params:  FREQ=12k  Q=15
```

Note for information about passing parameters to a hierarchical block, please refer to [“Passing Parameters Through a Hierarchy”](#) on page 70.

## 8.10 Special Parts

### 8.10.1 Initial Conditions

Initial conditions force a node to a fixed voltage or current during the calculation of the DC bias point. There are two types of initial condition namely *soft* and *hard*. Soft initial conditions apply a voltage through a fixed resistance. Hard initial conditions, apply a voltage directly without any resistance.

#### To Place a Soft Initial Condition

1. Select menu **Place | Connectors | Initial Condition**.
2. Place device at the desired location then select and press **F7**. Enter a suitable voltage.

#### To Place a Hard Initial Condition

1. Select menu **Place | From Symbol Library**
2. Select device Connections → Ics and Nodesets → Initial Condition (Hard).
3. Place device at the desired location then select and press **F7**. Enter a suitable voltage.

### Notes

Soft initial conditions are implemented using the .IC control and will also correctly apply an initial condition when Skip DC bias point is specified for a transient analysis. The driving resistance for a soft initial condition is 1 by default but can be altered using the ICRES simulator option. To do this, add .OPTIONS ICRES=nnn to the **F11** window (see [“Manual Entry of Simulator Commands”](#) on page 42).

Hard Initial conditions are implemented using a voltage source with the DCOP parameter specified. This feature is proprietary to SIMetrix and is not compatible with other SPICE simulators. Refer to the *Simulator Reference Manual/Analog Device Reference/Voltage Source* for more information on voltage sources and the DCOP parameter.

### 8.10.2 Nodesets

Nodesets are used to help convergence and also to coerce a particular state for circuits that have more than one possible DC solution. More information about nodesets is given in the *Simulator Reference Manual/Command Reference/.NODESET*.

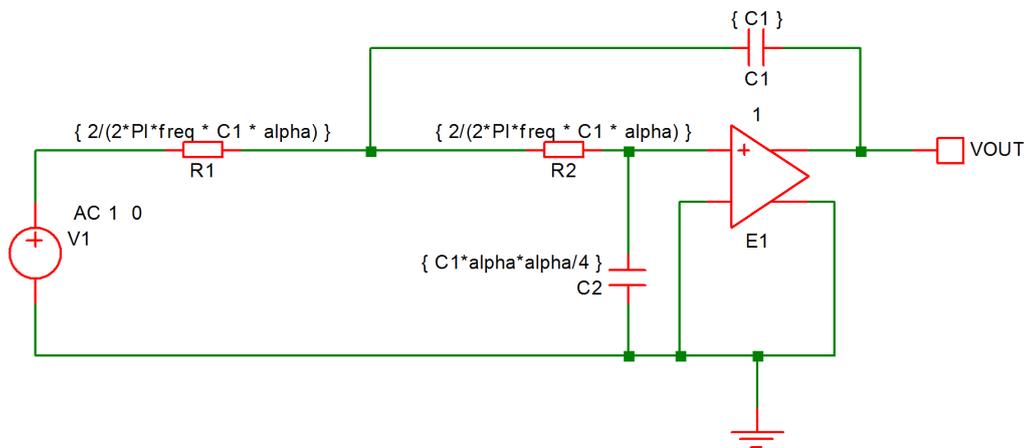
#### To Place a Nodeset

1. Select menu **Place | Connectors | Nodeset**
2. Place device at the desired location then select and press **F7**. Enter a suitable voltage.

## 8.11 Parameters and Expressions

You can specify both device and model parameters using an arithmetic expression containing user defined variables. The variables may be defined using the .PARAM simulator control, which must be placed in the netlist, or globally in a script using the Let command. A variable may also be swept using the parameter sweep mode for the swept analyses and stepped for multi-step analyses. Complete documentation on this subject can be found in the “Simulator Devices” chapter of the *Simulator Reference Manual*. Below are brief details of how to use expressions with a schematic based design. We explain this with an example.

### 8.11.1 Example



The above circuit is that of a two pole low-pass filter. C1 is fixed and R1=R2. The design equations are:

$$R1=R2=2 / (2*\pi*f0*C1*\alpha)$$

$$C2=C1*\alpha*\alpha/4$$

where freq is the cut off frequency and alpha is the damping factor.

Expressions for device values must be entered enclosed in curly braces (‘{’ and ‘}’).

Before running the above circuit you must assign values to the variables. This can be done by one of three methods:

1. With the .PARAM control placed in the netlist.
2. With Let command from the command line or from a script. (If using a script you must prefix the parameter names with ‘global:’)
3. By sweeping the value using the parameter mode of a [swept analysis](#) or [multi-step analysis](#).

Expressions for device values must be entered enclosed in curly braces (‘{’ and ‘}’).

Suppose we wish a 1kHz roll off for the above filter.

Using the .PARAM control, add these lines to the netlist (using the **F11** window - see “[Manual Entry of Simulator Commands](#)” on page 42):

```
.PARAM f0 1k
.PARAM alpha 1
.PARAM C1 10n
```

Using the Let command, you would type:

```
Let f0=1k
Let alpha=1
Let C1=10n
```

If you then wanted to alter the damping factor to 0.8 you only need to type in its new value:

```
Let alpha=0.8
```

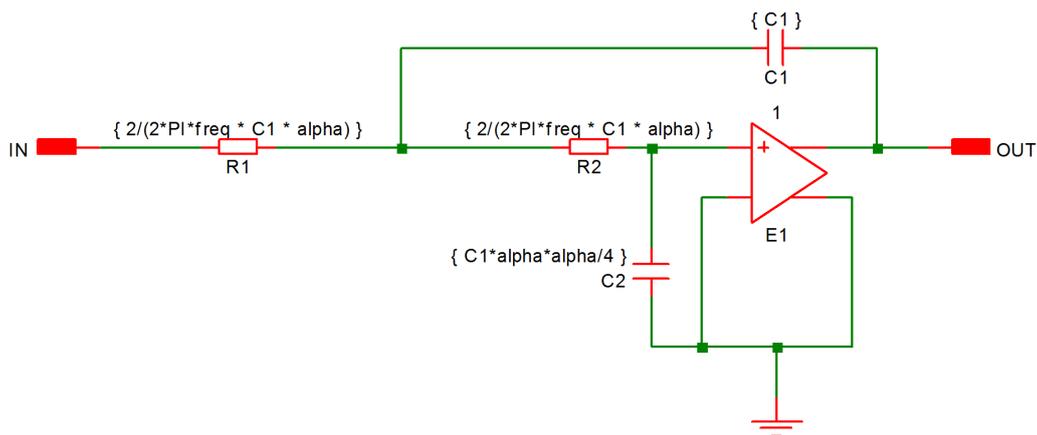
then re-run the simulator.

To execute the Let commands from within a script, prefix the parameter names with 'global:'. E.g. 'Let global:f0=1k'.

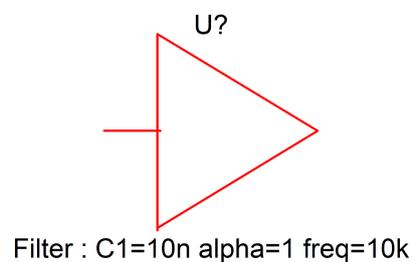
In many cases the .PARAM approach is more convenient as the values can be stored with the schematic.

### Example passing parameters to subcircuits

If the filter example above was implemented as a subcircuit, different values of the parameters freq, alpha and C1 could be passed to each instance of the subcircuit. This allows several filters with differing roll-off frequencies and damping factors to be quickly drawn.



You can set the values of the parameters for each instance of the above subcircuit by appending the expressions to the VALUE property of the symbol in the main circuit separated by a ':'. E.g.:-



## Chapter 9

# Analog Behavioural Modelling

### 9.1 Introduction

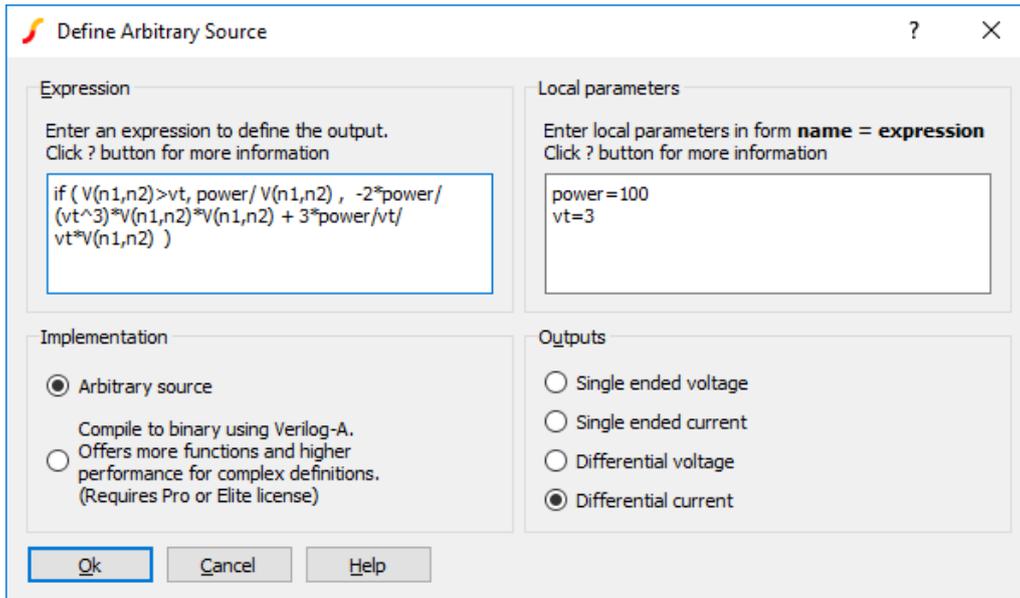
Analog Behavioural Modelling is the process of creating an analog device from a description of its behaviour rather than its low-level design.

SIMatrix offers three way to create Analog Behavioural Models:

- Verilog-A Modelling. Verilog-A is a compiled language that can be used to create sophisticated analog devices defined using non-linear equations, frequency domain definitions, time-domain events and other features. For further information refer to the *Verilog-A Manual*
- Non-linear Transfer Function. This is a single device with a single-ended or differential output which generates a voltage or current expressed as a function of an arbitrary number of voltage and current inputs. See [“Non-linear Transfer Function” on page 188](#)
- Laplace Transfer Function. This defines a linear device in the frequency domain using a Laplace transform with the ‘s’ variable.. See [“Laplace Transfer Function” on page 198](#)

### 9.2 Non-linear Transfer Function

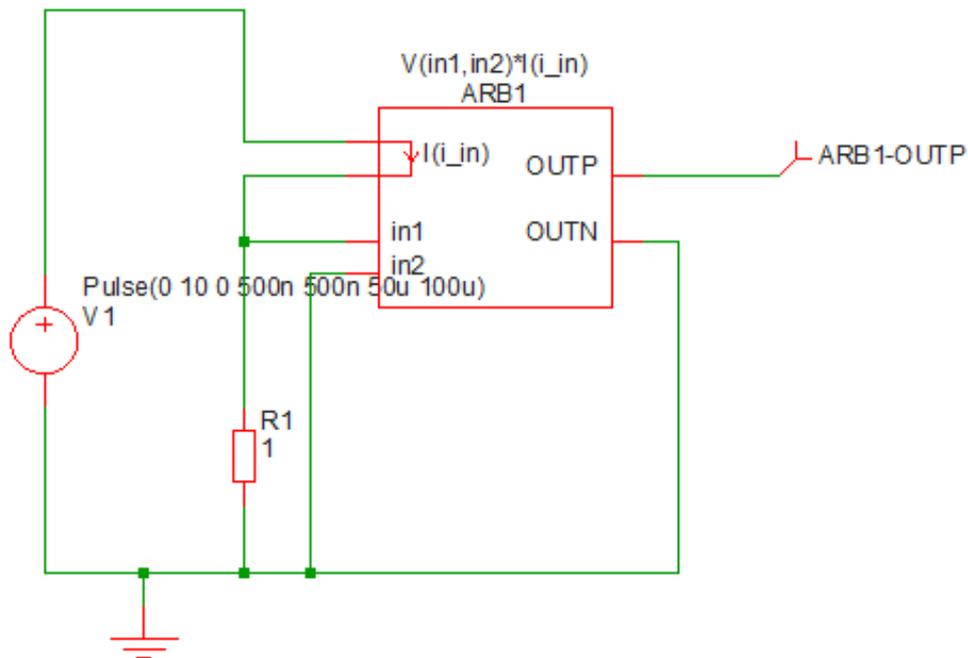
Select menu **Place | Analog Behavioural | Non-linear Transfer Function**. This displays:



You may specify an equation that defines an output voltage or current in terms of any number of input voltages and currents. Input voltages are specified in the form  $V(a)$  or  $V(a,b)$  where  $a$  and  $b$  may be any arbitrary name of your choice. Input currents are specified in the form  $I(a)$ . On completion, SIMetrix will generate a schematic symbol complete with the input voltages and/or currents that you reference in the equation. There is no need to specify how many input voltages and currents you wish to use. SIMetrix will automatically determine this from the equation.

As well as input voltage and currents, you can also reference the output voltage or current in your equation. A single ended output voltage is accessed using  $V(out)$  while a differential output voltage is accessed using  $V(outp,outn)$ . If you specify an output voltage, you may also access the current flowing through it using  $I(out)$ .

In the example above, the expression shown -  $V(vina,vinb)*I(iin)$  - multiplies a voltage and current together. This could be used to monitor the power in a two terminal device as shown in the following schematic.:



In the above,  $ARB1$  is the device created from the Non-linear Transfer Function menu.  $ARB1-OUTP$  will carry a voltage equal to the power dissipation in  $R1$ .

### 9.2.1 Expression

The expression entered defines the output of the device in terms of its inputs. Expressions generally comprise the following elements:

- Circuit variables
- Parameters
- Constants.
- Operators
- Functions
- Look up tables

These are described in the following sections.

#### Circuit Variables

Circuit variables allow the expression to reference input voltages and currents.

Voltages are of the form:

$V(\text{node\_name1})$

OR

$V(\text{node\_name1}, \text{node\_name2})$

Where *node\_name1* and *node\_name2* are input connections. The second form above returns the difference between the voltages on *node\_name1* and *node\_name2*. The special names *out*, *outp* and *outn* may be used to access the voltage on the output. Use *out* for the single ended output configuration and *outp* and/or *outn* for the differential output configuration.

Currents are of the form:

$$I(\text{source\_name})$$

Where *source\_name* is the name of an input current. For output voltage configurations, you can use  $I(\text{out})$  to access the output current of the device itself. For example

$$100 * I(\text{OUT})$$

Implements a 100 ohm resistor for a device configured with a differential output voltage.

## Parameters

The expression may use any number of parameters. These can be local or global. Local parameters are defined in the Local parameters edit box on the right hand side. Global parameters must be defined within the schematic, usually in the F11 window.

To enter local parameters, enter one or more definitions in the form **name=expression** in the Local parameters edit box. **expression** may be a constant or an expression referring to other local or global parameters. **expression** may not contain circuit variables. (E.g.  $V(\text{in1})$ ). Comment lines may be entered and must start with a '\*’.

## Built-in Parameters

A number of parameter names are assigned by the simulator. These are:

Parameter name	Description
TIME	Resolves to time for transient analysis. Resolves to 0 otherwise including during the pseudo transient operation point algorithm
TEMP	Resolves to current circuit temperature in Celsius
HERTZ	Resolves to frequency during AC sweep and zero in other analysis modes
PTARAMP	Resolves to value of ramp during pseudo transient operating point algorithm. Otherwise this value resolves to 1.

## Constants

Apart from simple numeric values, arbitrary expressions may also contain the following built-in constants:

Constant name	Value	Description
PI	3.14159265358979323846	$\pi$
E	2.71828182845904523536	$e$
TRUE	1.0	

Constant name	Value	Description
FALSE	0.0	
ECHARGE	1.6021918e-19	Charge on an electron in coulombs
BOLTZ	1.3806226e-23	Boltzmann's constant

## Operators

These are listed below and are listed in order of precedence. Precedence controls the order of evaluation. So  $3*4 + 5*6 = (3*4) + (5*6) = 42$  and  $3+4*5+6 = 3 + (4*5) + 6 = 29$  as '\*' has higher precedence than '+'.

Operator	Description
~! -	Digital NOT, Logical NOT, Unary minus
^ or **	Raise to power.
* /	Multiply, divide
+ -	Plus, minus
>= <=, > <	Comparison operators
== != or <>	Equal, not equal
&	Digital AND (see below)
	Digital OR (see below)
&&	Logical AND
	Logical OR
test ? true_expr : false_expr	Ternary conditional expression (see below)

## Comparison, Equality and Logical Operators

These are Boolean in nature either accepting or returning Boolean values or both. A Boolean value is either TRUE or FALSE. FALSE is defined as equal to zero and TRUE is defined as not equal to zero. So, the comparison and equality operators return 1.0 if the result of the operation is true otherwise they return 0.0.

The arguments to equality operators should always be expressions that can be guaranteed to have an exact value e.g. a Boolean expression or the return value from functions such as SGN. The == operator, for example, will return TRUE only if both arguments are *exactly* equal. So the following should never be used:

```
v(n1)==5.5
```

v(n1) may not ever be exactly 5.0. It may be 5.499999999 or 5.5000000001 but only by chance will it be 5.5.

These operators are intended to be used with the IF() function described in [“Ternary Conditional Expression” on page 193](#).

## Digital Operators

These are the operators '&', '|' and '~'. These are now considered to be obsolete and may not be supported in future versions of SIMetrix. These operators implement logic operations in the analog domain and are

continuous in nature.

## Ternary Conditional Expression

This is of the form:

$$\text{test\_expression} ? \text{true\_expression} : \text{false\_expression}$$

The value returned will be *true\_expression* if *test\_expression* resolves to a non-zero value, otherwise the return value will be *false\_expression*. This is functionally the same as the IF() function described in the functions table below.

## Functions

Function	Description
ABS(x)	$ x $
ACOS(x), ARCCOS(x)	$\cos^{-1}(x)$
ACOSH(x)	$\cosh^{-1}(x)$
ASIN(x), ARCSIN(x)	$\sin^{-1}(x)$
ASINH(x)	$\sinh^{-1}(x)$
ATAN(x), ARCTAN(x)	$\tan^{-1}(x)$
ATAN2(x,y)	$\tan^{-1}(x/y)$ . Valid if $y \neq 0$
ATANH(x)	$\tanh^{-1}(x)$
COS(x)	$\cos(x)$
COSH(x)	$\cosh(x)$
DDT(x)	$dx/dt$
EXP(x)	$e^x$
FLOOR(x), INT(x)	Next lowest integer of $x$ .
IF(cond, x, y[, maxslew])	if cond is TRUE $result = x$ else $result = y$ . If $maxslew > 0$ the rate of change of the result will be slew rate controlled. See <a href="#">“IF() Function” on page 195</a> .
IFF(cond, x, y[, maxslew])	As IF(cond, x, y, maxslew)
LIMIT(x, lo, hi)	if $x < lo$ $result = lo$ else if $x > hi$ $result = hi$ else $result = x$
LIMITS(x, lo, hi, sharp)	As LIMIT but with smoothed corners. The ‘sharp’ parameter defines the abruptness of the transition. A higher number gives a sharper response. LIMITS gives better convergence than LIMIT. See <a href="#">“LIMITS() Function” on page 195</a> . below
LN(x)	$\ln(x)$ If $x < 10^{-100}$ $result = -230.2585093$
LNCOSH(x)	$\ln(\cosh(x)) = \int \tanh(x) dx$
LOG(x)	$\log_{10}(x)$ . If $x < 10^{-100}$ $result = -100$
LOG10(x)	$\log_{10}(x)$ . If $x < 10^{-100}$ $result = -100$
MAX(x, y)	Returns larger of $x$ and $y$
MIN(x,y)	Returns smaller of $x$ and $y$
PWR(x,y)	$x^y$

Function	Description
PWRS(x,y)	$x \geq 0: x^y, x < 0: -x^y$
SDT(x)	$\int x dt$
SGN(X)	$x > 0: 1, x < 0: -1, x = 0: 0$
SIN(x)	$\sin(x)$
SINH(x)	$\sinh(x)$
SQRT(x)	$x \geq 0: \sqrt{x}, x < 0: \sqrt{-x}$
STP(x)	$x \leq 0: 0, x > 0: 1$
TABLE(x,xy_pairs)	lookup table. Refer to <a href="#">“Lookup tables” on page 196</a>
TABLEX(x,xy_pairs)	Same as TABLE except end points extrapolated. Refer to <a href="#">“Lookup tables” on page 196</a>
TAN(x)	$\tan(x)$
TANH(x)	$\tanh(x)$
U(x)	as STP(x)
URAMP(x)	$x < 0: 0, x \geq 0: x$

## Monte Carlo Distribution Functions

The following functions return a randomly generated value when running a Monte Carlo analysis. They can be used to define component tolerances.

Name	Distribution	Lot?
DISTRIBUTION	User defined distribution	No
DISTRIBUTIONL	User defined distribution	Yes
UD	Alias for DISTRIBUTION()	No
UDL	Alias for DISTRIBUTIONL()	Yes
GAUSS	Gaussian	No
GAUSSL	Gaussian	Yes
GAUSSTRUNC	Truncated Gaussian	No
GAUSSTRUNCL	Truncated Gaussian	Yes
UNIF	Uniform	No
UNIFL	Uniform	Yes
WC	Worst case	No
WCL	Worst case	Yes
GAUSSE	Gaussian logarithmic	No
GAUSSEL	Gaussian logarithmic	Yes
UNIFE	Uniform logarithmic	No
UNIFEL	Uniform logarithmic	Yes
WCE	Worst case logarithmic	No
WCEL	Worst case logarithmic	Yes

A full discussion on the use of Monte Carlo distribution functions is given in *Simulator Reference Manual/Monte Carlo, Sensitivity and Worst-case/Specifying Tolerances*.

## IF() Function

```
IF(condition, true-value, false-value[, max-slew])
```

The result is:

if *condition* is non-zero, result is *true-value*, else result is *false-value*.

If *max-slew* is present and greater than zero, the result will be slew-rate limited in both positive and negative directions to the value of *max-slew*.

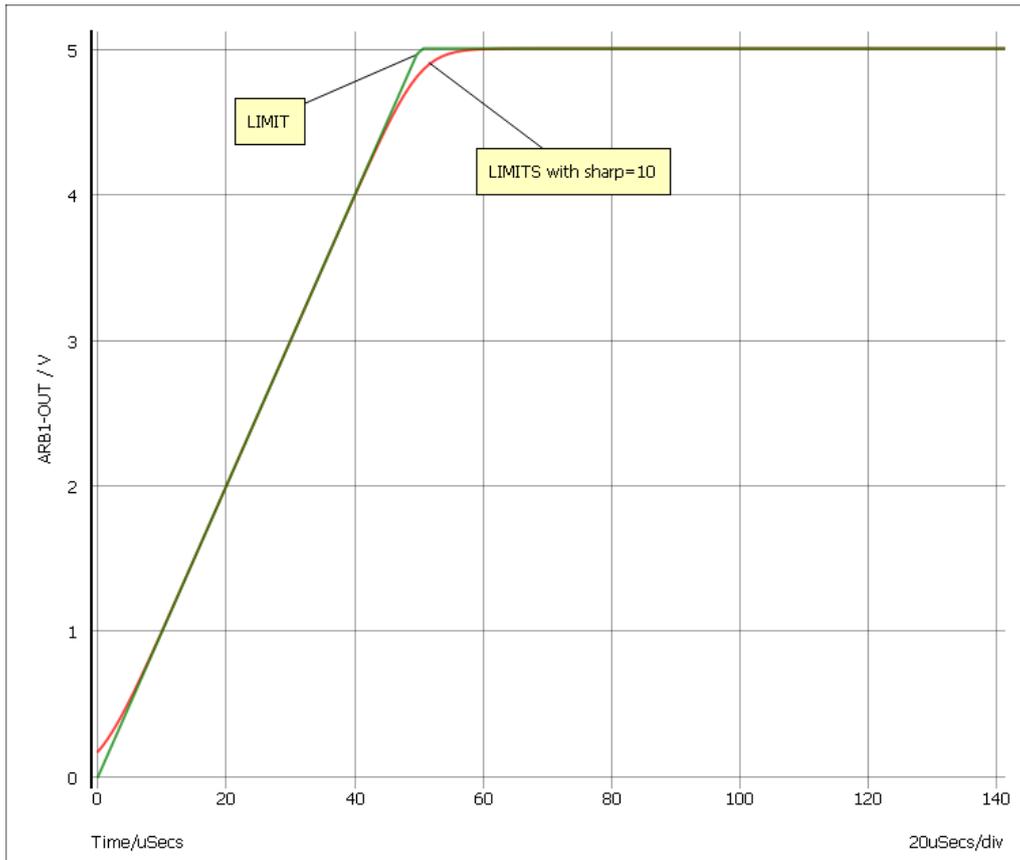
In some situations, for example if *true-value* and *false-value* are constants, the result of this function will be discontinuous when *condition* changes state. This can lead to non-convergence as there is no lower bound on the time-step. In these cases a *max-slew* parameter can be included. This will limit the slew rate so providing a controlled transition from the *true-value* to the *false-value* and vice-versa.

If the simulator setting `.OPTIONS DISCONTINUOUSIFSLEWRATE=max-slew`, is provided, SIMetrix will automatically apply a *max-slew* parameter to all occurrences of the IF() function if both *true-value* and *false-value* are constants. This provides a convenient way of resolving convergence issues with third-party models that make extensive use of discontinuous if expressions. Note that the `DISCONTINUOUSIFSLEWRATE` option is also applied to conditional expressions using the C-style `condition ? true-value : false-value` syntax.

## LIMITS() Function

```
LIMITS(x, low, high, sharp)
```

The LIMITS() function is similar to LIMIT but provides a smooth response at the corners which leads to better convergence behaviour. The behaviour is shown below



The LIMITS function follows this equation:

$$\text{LIMITS}(x, \text{low}, \text{high}, \text{sharp}) = 0.5 * ((\ln(\cosh(v1)) - \ln(\cosh(v2)))/v3 + (\text{low} + \text{high}))$$

Where

$$v1 = \text{sharp}/(\text{high}-\text{low})*(x-\text{low})$$

$$v2 = \text{sharp}/(\text{high}-\text{low})*(x-\text{high})$$

$$v3 = \text{sharp}/(\text{high}-\text{low})$$

## Look-up Tables

Expressions may contain any number of look-up tables. This allows a transfer function of a device to be specified according to - say - measured values without having to obtain a mathematical equation. Look-up tables are specified in terms of x, y value pairs which describe a piece-wise linear transfer function. There are three ways to create a lookup table:

Method 1	Syntax in form: <code>TABLE[xy_pairs](input_expression)</code> .				
	<table border="0"> <tr> <td style="vertical-align: top;"><code>xy_pairs</code></td> <td>A sequence of comma separated pairs of constant values that define the input and output values of the table. For each pair, the first value is the x or input value and the second is the y or output value. Only explicit numeric constants may be used. Even internal constants such as PI may not be used.</td> </tr> <tr> <td style="vertical-align: top;"><code>input_expression</code></td> <td>Expression defining the input or x value of the table.</td> </tr> </table>	<code>xy_pairs</code>	A sequence of comma separated pairs of constant values that define the input and output values of the table. For each pair, the first value is the x or input value and the second is the y or output value. Only explicit numeric constants may be used. Even internal constants such as PI may not be used.	<code>input_expression</code>	Expression defining the input or x value of the table.
<code>xy_pairs</code>	A sequence of comma separated pairs of constant values that define the input and output values of the table. For each pair, the first value is the x or input value and the second is the y or output value. Only explicit numeric constants may be used. Even internal constants such as PI may not be used.				
<code>input_expression</code>	Expression defining the input or x value of the table.				
Method 2	Syntax in form: <code>TABLE(input_expression, xy_pairs)</code> .				
	<table border="0"> <tr> <td style="vertical-align: top;"><code>input_expression</code></td> <td>Expression defining the input or x value of the table.</td> </tr> <tr> <td style="vertical-align: top;"><code>xy_pairs</code></td> <td>A sequence of comma separated pairs of values that define the input and output values of the table. For each pair, the first value is the x or input value and the second is the y or output value. Unlike method 1, values do not need to be literal constants and may be expressions containing parameters and <a href="#">“circuit variables” on page 190</a> that reference input voltages and currents.</td> </tr> </table> <p>The TABLE function retains the value of the end point for x values beyond the range of the table. For example, if the last point for the TABLE function is (10,5) then any input greater than 10 will give an output of 5.</p> <p>If any value references a circuit variable, the expression in which the TABLE function is used must ultimately be evaluated in an arbitrary source</p>	<code>input_expression</code>	Expression defining the input or x value of the table.	<code>xy_pairs</code>	A sequence of comma separated pairs of values that define the input and output values of the table. For each pair, the first value is the x or input value and the second is the y or output value. Unlike method 1, values do not need to be literal constants and may be expressions containing parameters and <a href="#">“circuit variables” on page 190</a> that reference input voltages and currents.
<code>input_expression</code>	Expression defining the input or x value of the table.				
<code>xy_pairs</code>	A sequence of comma separated pairs of values that define the input and output values of the table. For each pair, the first value is the x or input value and the second is the y or output value. Unlike method 1, values do not need to be literal constants and may be expressions containing parameters and <a href="#">“circuit variables” on page 190</a> that reference input voltages and currents.				
Method 3	Syntax in form: <code>TABLEX(input_expression, xy_pairs)</code> .				
	<table border="0"> <tr> <td style="vertical-align: top;"><code>input_expression</code></td> <td>Expression defining the input or x value of the table.</td> </tr> <tr> <td style="vertical-align: top;"><code>xy_pairs</code></td> <td>Same as method 2 except that out-of-range points are extrapolated from the final segment. See <a href="#">“Table Lookup Example” on page 197</a></td> </tr> </table>	<code>input_expression</code>	Expression defining the input or x value of the table.	<code>xy_pairs</code>	Same as method 2 except that out-of-range points are extrapolated from the final segment. See <a href="#">“Table Lookup Example” on page 197</a>
<code>input_expression</code>	Expression defining the input or x value of the table.				
<code>xy_pairs</code>	Same as method 2 except that out-of-range points are extrapolated from the final segment. See <a href="#">“Table Lookup Example” on page 197</a>				

Method 1 is more efficient at handling large tables (hundreds of values). However, method 2 is generally more flexible and is the recommended choice for most applications. Method 2 is also compatible with other simulators whereas method 1 is proprietary to SIMetrix.

For an example see [“Table Lookup Example” on page 197](#)

## 9.2.2 Examples

### Table Lookup Example

The following arbitrary source definition implements a soft limiting function

#### Using method 1:

```
table[-10, -5, -5, -4, -4, -3.5, -3, -3, 3, 3, 4, 3.5, 5, 4, 10, 5](v(N1))
```

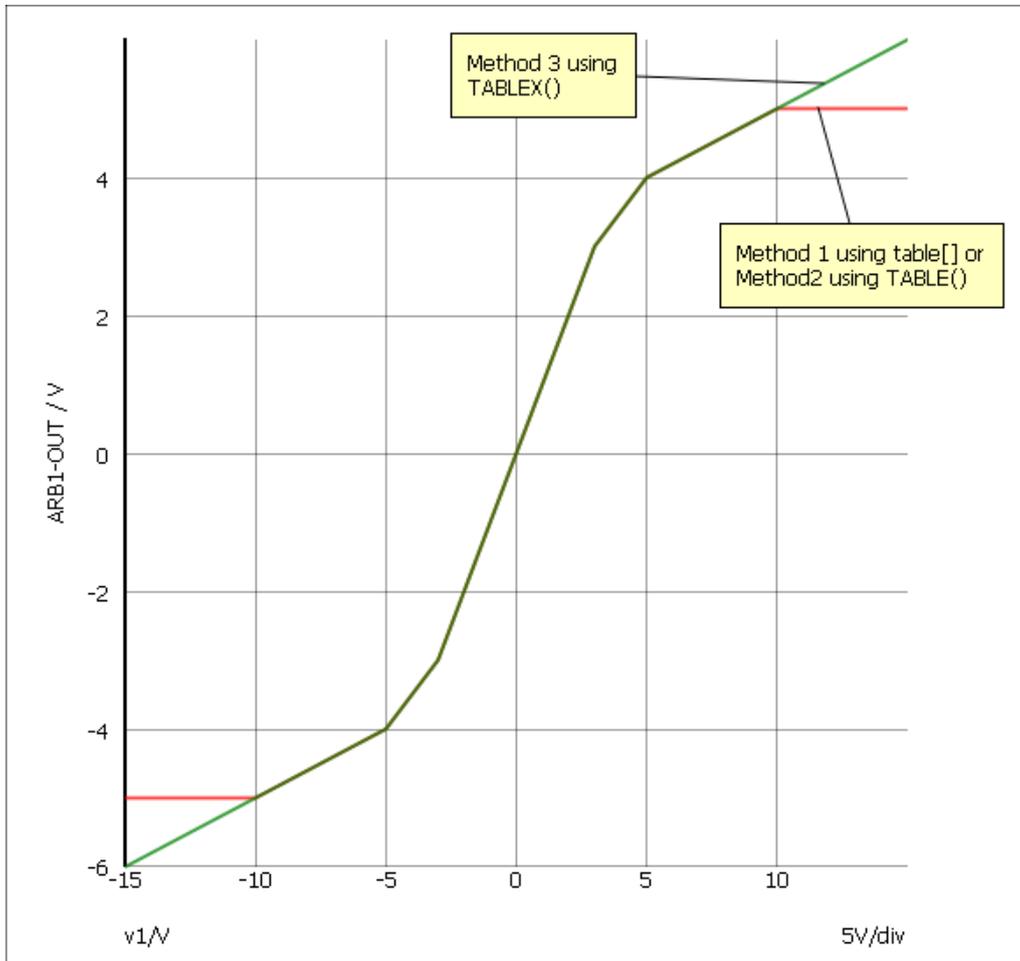
#### Using method 2:

TABLE (v (N1), -10, -5, -5, -4, -4, -3.5, -3, -3, 3, 3, 4, 3.5, 5, 4, 10, 5)

Using method 3:

TABLEX (v (N1), -10, -5, -5, -4, -4, -3.5, -3, -3, 3, 3, 4, 3.5, 5, 4, 10, 5)

The resulting transfer functions are illustrated in the following picture:



### 9.2.3 Verilog-A Implementation

If you have *SIMetrix Pro* or *SIMetrix Elite* you may select the Compile to binary using Verilog-A option. This will build a definition of the device using the Verilog-A language then compile it to a binary DLL. This process takes place when you run the simulation. The benefit of using Verilog-A is that there is a wider range of functions available and for complex definitions there will also be a performance benefit. Refer to the Verilog-A Manual for details of available functions.

## 9.3 Laplace Transfer Function

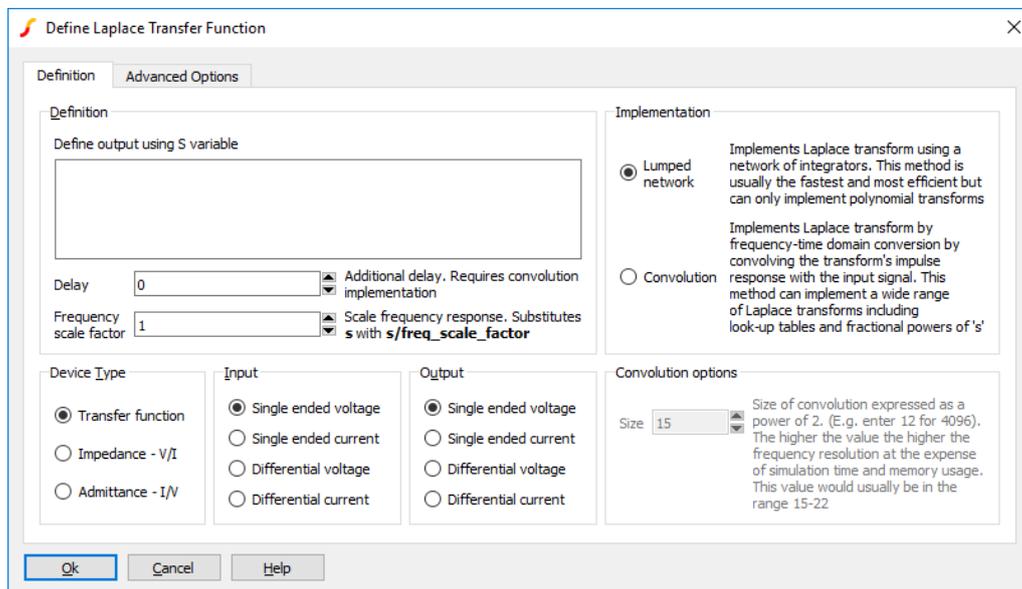
### 9.3.1 Overview

The Laplace transfer function device implements a linear device defined in the frequency domain by a Laplace transform. For example the Laplace transform  $\frac{1}{s+1}$  defines a first order low pass filter while  $exp(-s)$  defines a 1 second delay.

The SIMetrix Laplace transfer function device features two different methods of implementation, namely "lumped network" and "convolution". The lumped network implementation is fast and efficient and may be used for transfer functions that can be represented in polynomial form. The convolution method is more general purpose and can model a wide range of transfer functions. High performance is achieved by using a fast FFT-based convolution method.

### 9.3.2 User Interface

Selecting the menu **Place | Analog Behavioural | Laplace Transfer Function** brings up the following dialog

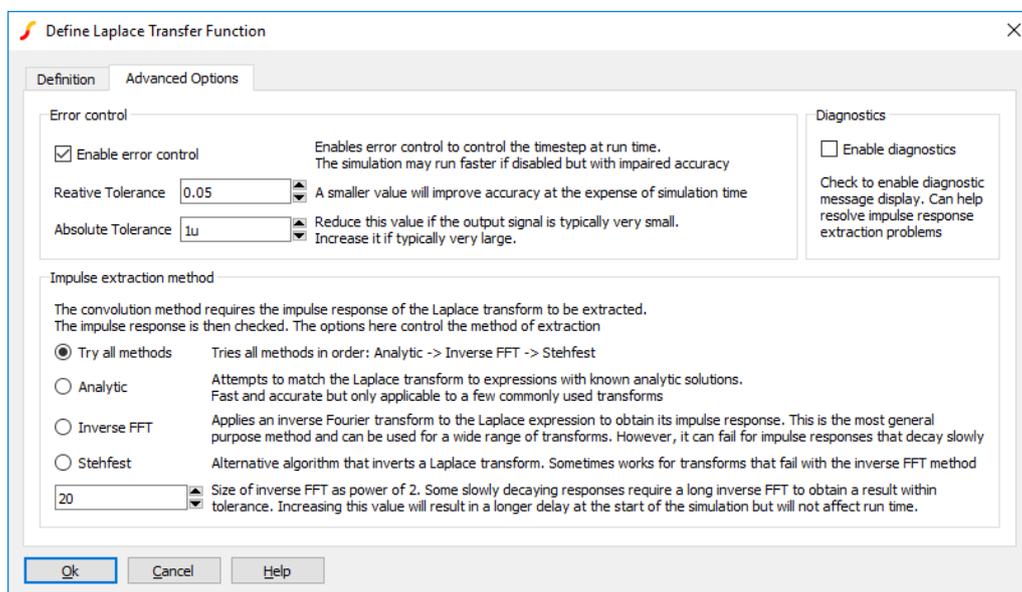


The operation of the various controls is described below.

Definition	See <a href="#">“Laplace expression” on page 201</a>						
Delay	Adds delay in seconds. Requires convolution method. Equivalent to multiplying transfer functions by $exp(-delay.s)$						
Frequency scale factor	Multiplier for frequency						
Device type	<table> <tbody> <tr> <td>Transfer function</td> <td>Expression defines output/input.</td> </tr> <tr> <td>Impedance V/I</td> <td>Two terminal device, expression defines voltage/current.</td> </tr> <tr> <td>Admittance I/V</td> <td>Two terminal device, expression defines current/voltage.</td> </tr> </tbody> </table>	Transfer function	Expression defines output/input.	Impedance V/I	Two terminal device, expression defines voltage/current.	Admittance I/V	Two terminal device, expression defines current/voltage.
Transfer function	Expression defines output/input.						
Impedance V/I	Two terminal device, expression defines voltage/current.						
Admittance I/V	Two terminal device, expression defines current/voltage.						
Input	Input configuration for transfer function						

Output	Output configuration for transfer function
Implementation	See “Implementation” on page 204
Convolution Options	Convolution size. Increase this value to increase the upper frequency limit. This will slow down the simulation and increase memory consumption. Memory usage for a single device is approximately $96 * 2^N$ bytes where $N$ is the value entered here. Note that memory is partly shared for multiple devices. The upper frequency is of the order of $2^{N-1}/T$ where $T$ is the simulation time. So for example, with a simulation time of 1ms and $N=15$ , the upper frequency limit will be in the region of 16MHz. Higher frequencies will be gradually attenuated. As well as increasing memory consumption, higher values of convolution size will slow the simulation. This does not become significant until $N$ exceeds about 17 to 18 but increases markedly for higher values.

The Advanced Options tab displays the following:



Error control	Configures time-step control algorithm. For rapidly changing inputs the time step must be kept small to retain good accuracy. The parameters entered here control this mechanism.
Diagnostics	Check box to enable some diagnostic run-time messages. These provide information on the extraction of the impulse response.

Impulse extraction method	Parameters controlling extraction of the impulse response.	
	Try all methods	All methods attempted in sequence
	Analytic	Attempt analytic method only
	Inverse FFT	Attempt inverse FFT method only
	Stehfest	Attempt Stehfest method only
	Size of inverse FFT as power of 2	Using a longer inverse FFT to extract an impulse response can increase the accuracy of the results especially if the impulse response decays slowly

### 9.3.3 Laplace Expression

The Laplace expression defines the behaviour of the device in the frequency domain. For example '1 / (s+1)', defines a simple single pole low-pass filter. The expression may contain arithmetic operators and a number of functions as described in the following sections. Be aware that not any expression is physically realisable; for example 'exp (s)' defines a negative delay, that is a device whose output responds to an input in the future.

#### Operators

+ - \* / ^

where ^ means raise to power. For lumped network implementation, the power must be an integer. Non-integral powers may be entered for convolution implementation.

#### Constants

Any decimal number following normal rules. SPICE style engineering suffixes are accepted.

#### s Variable

This can be raised to a power using ^, for example s^2. If the power is an integer between 0 and 9 the ^ may be omitted. For example: s2 is the same as s^2.

#### Functions

The following functions may be used in convolution mode only.

Function Syntax	Description
sqrt(x)	$\sqrt{x}$
exp(x)	$e^x$
ln(x)	$\log_e(x)$
log10(x)	$\log_{10}(x)$
sin(x)	$\sin(x)$
cos(x)	$\cos(x)$

Function Syntax	Description
$\tan(x)$	$\tan(x)$
$\text{acos}(x)$	$\cos^{-1}(x)$
$\text{asin}(x)$	$\sin^{-1}(x)$
$\text{atan}(x)$	$\tan^{-1}(x)$
$\sinh(x)$	$\sinh(x)$
$\cosh(x)$	$\cosh(x)$
$\tanh(x)$	$\tanh(x)$
$\text{asinh}(x)$	$\sinh^{-1}(x)$
$\text{acosh}(x)$	$\cosh^{-1}(x)$
$\text{atanh}(x)$	$\tanh^{-1}(x)$
$\text{atan2}(x,y)$	$\tan^{-1}\left(\frac{y}{x}\right)$
$\text{pow}(x,y)$	$x^y$

## Filter response functions

Filter response functions may be used in both lumped network implementation and convolution implementation.

These are described in the following table:

Function Syntax	Filter Response
$\text{BesselLP}(\text{order}, \text{cut-off})$	Bessel low-pass
$\text{BesselHP}(\text{order}, \text{cut-off})$	Bessel high-pass
$\text{ButterworthLP}(\text{order}, \text{cut-off})$	Butterworth low-pass
$\text{ButterworthHP}(\text{order}, \text{cut-off})$	Butterworth high-pass
$\text{ChebyshevLP}(\text{order}, \text{cut-off}, \text{passband\_ripple})$	Chebyshev low-pass
$\text{ChebyshevHP}(\text{order}, \text{cut-off}, \text{passband\_ripple})$	Chebyshev high-pass

Where:

<i>order</i>	Integer specifying order of filter. There is no maximum limit but in practice orders larger than about 50 tend to give accuracy problems.
<i>cut-off</i>	-3dB Frequency in Hertz.
<i>passband_ripple</i>	Chebyshev only. Passband ripple spec. in dB.

## Lookup tables

The frequency response of a system may be defined in tabular form using lookup tables. The lookup table consists of a sequence of values arranged in triplets. Each triplet is in the form *frequency, value1, value2* where *value1* and *value2* define the magnitude and phase in various ways as described below.

There are five variants of the lookup table as described below:

Function Name	value1	value2
Table	dB	phase in degrees
Table_M	magnitude	phase in degrees
Table_R	dB	phase in radians
Table_MR	magnitude	phase in radians
Table_RI	real part	imaginary part

**Example**

```
Table_M(0.01, 1, 30, 1, 1, 30, 10, 0.1, 30, 100, 0.01, -30)
```

The above defines a response with a gain of 1 and a phase shift of 30 degrees from 0.01Hz to 1Hz. From 1Hz to 10Hz the gain falls from 1 to 0.1 with the phase shift remaining at 30 degrees. From 10Hz to 100Hz the gain falls from 0.1 to 0.01 and the phase changes from 30 to -30 degrees.

In real life it is not possible to implement a characteristic with the above behaviour as it is non-causal. However, it can be implemented if a suitable delay is added to the characteristic. The delay may be added in the Define Laplace Transfer Function dialog box in the Definition section - see the Delay entry. Alternatively it may be added to the Laplace expression using  $\exp(-delay.s)$ . E.g:

```
Table_M(0.01, 1, 30, 1, 1, 30, 10, 0.1, 30, 100, 0.01, -30) * exp(-s*2)
```

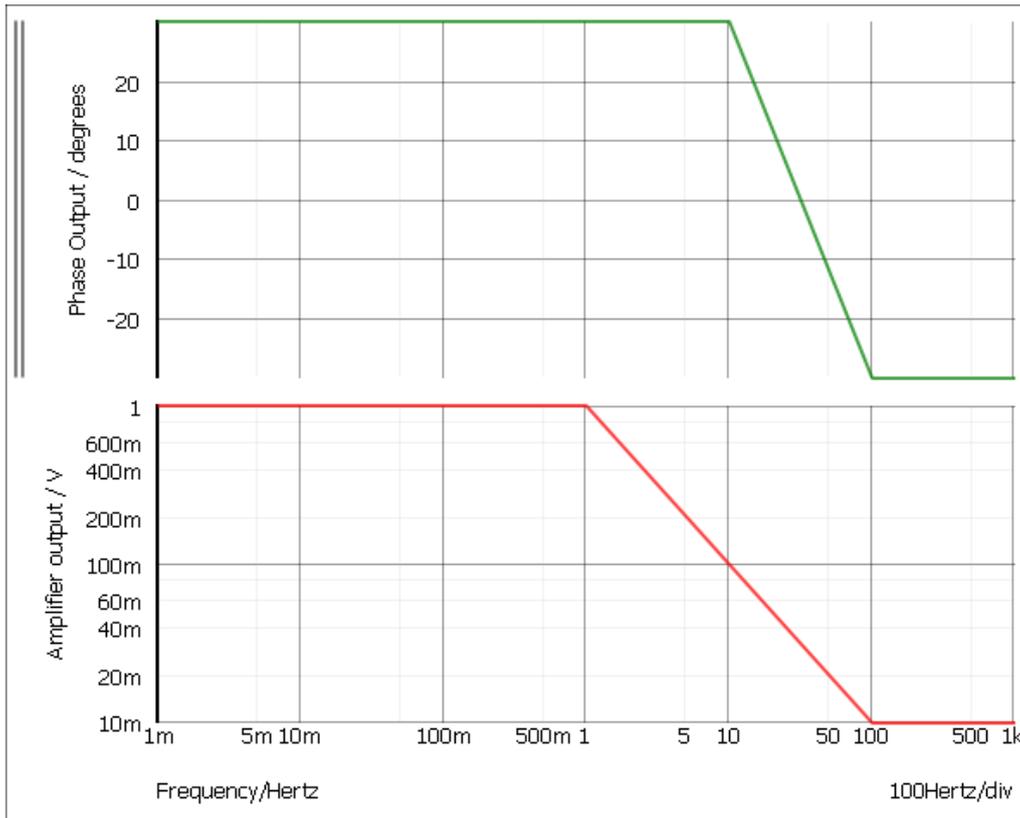
The above adds a 2 second delay.

For ease of reading, each triplet in the table may be placed on a separate line by prefixing each line with a '+' character. E.g.

```
Table_M(
+ 0.01,      1,      30,
+ 1,         1,      30,
+ 10,        0.1,    30,
+ 100,       0.01,   -30)
```

**Interpolation** Between defined frequency points, the Laplace transfer function finds the magnitude and phase using interpolation. The interpolation is performed logarithmically on the magnitude data. Values outside the table frequency range are defined by their respective end points.

For the above example the magnitude and phase are shown below:



## Parameters

Parameters (defined with .PARAM) may be used in Laplace expressions but currently only if convolution implementation is selected.

For the lumped network implementation you can use parameters in the underlying simulator device if you define the expression in terms of the numerator and denominator coefficients. Refer to the S-domain Transfer Function Block in the Simulator Reference Manual for further details.

### 9.3.4 Device type

Transfer function	Expression defines output/input.
Impedance V/I	Two terminal device, expression defines voltage/current.
Admittance I/V	Two terminal device, expression defines current/voltage.

### 9.3.5 Implementation

Describes the method the simulator uses to implementation the Laplace expression. There are two choices:

Lumped network	The Laplace transform is implemented using a network of integrators.
----------------	--

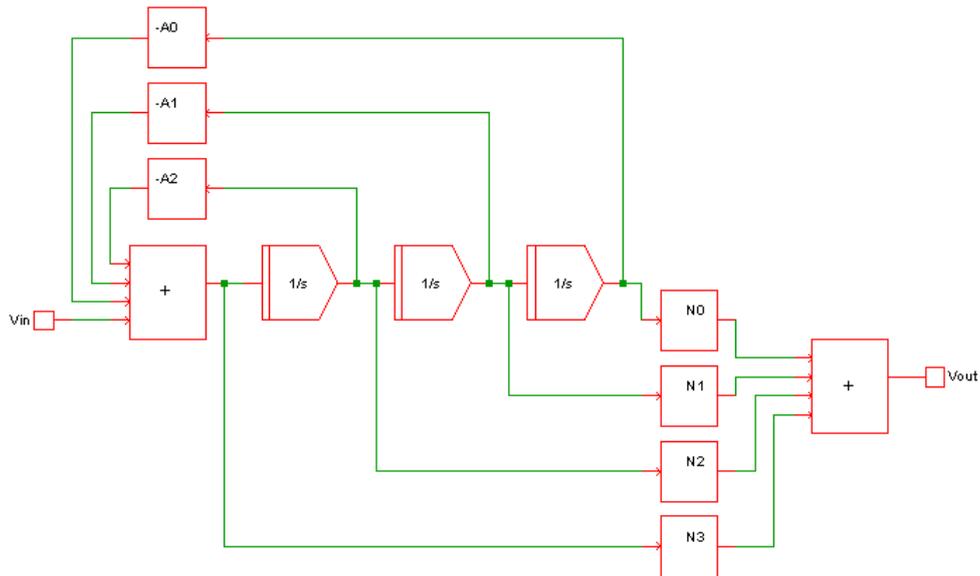
**Convolution**

The Laplace transform is implemented by convolving the impulse response of the Laplace transform with the input signal.

See below for further details.

**Lumped Network**

The diagram below shows the configuration used for a third order network.



$$V_{out} = V_{in} \cdot (N_3 \cdot s^3 + N_2 \cdot s^2 + N_1 \cdot s + N_0) / (s^3 + D_2 \cdot s^2 + D_1 \cdot s + D_0)$$

This method can implement a quotient of polynomials in the  $s$  variable and any expression that can be reduced to a quotient of polynomials. For such expressions it is fast and efficient.

This method cannot implement expressions containing arbitrary functions such as square-root or exponentials. Such Laplace expressions must be implemented using the convolution method.

**Convolution**

This is a general purpose method and can implement a wide range of Laplace transforms.

In the frequency domain, a linear system may be represented by:

$$V_{out}(s) = V_{in}(s) \cdot f(s)$$

where  $f(s)$  is the transfer function.

In the time domain the same system may be represented by:

$$V_{out}(t) = V_{in}(t) * f(t)$$

Where  $f(t)$  is the impulse response of  $f(s)$  and  $*$  is the convolution operator.

This method is challenging to implement as simple convolution applied to simulation data in its raw form is an  $\mathcal{O}(n^2)$  algorithm. This means that the number of computations required at each step is proportional to the square of the number of steps. In practice this becomes unacceptably slow when there are more than about 10000 time steps.

SIMetrix overcomes this performance limitation by using an FFT based fast convolution method with a computation speed of  $\mathcal{O}(n \log^2 n)$ . Although dramatically faster, this algorithm requires data to be presented to it at fixed-interval steps which has the effect of placing an upper frequency limit. The actual upper frequency limit can be arbitrarily increased by increasing the number of steps (the value of  $n$ ) at the expense of speed and the memory consumed. In practice the default settings work well in most applications and give good performance and accuracy.

To perform the convolution, the simulator must also extract the impulse response of the Laplace transform. SIMetrix has three methods to do this:

Analytical	Matches the Laplace transform to known analytical solutions. For example $\frac{1}{\sqrt{s}}$ has the impulse response of $\frac{1}{2\pi\sqrt{t}}$ . SIMetrix will automatically detect this and other known Laplace transforms.
Inverse FFT	Calculates the inverse Fourier transform of the Laplace expression. This is a general purpose method that can be applied to a wide-range of Laplace transforms. However, it has the condition that the impulse response must decay to zero during the time interval.
Stehfest method	A method to calculate the inverse Laplace transform. Sometimes this method works for slowly decaying transforms that fail with the inverse FFT method.

### 9.3.6 Impulse Response

#### Problems with Impulse Response Extraction

For the convolution method an impulse response of the Laplace transfer function must be extracted. As mentioned above, three methods are attempted. The inverse FFT method and Stehfest methods are approximate in nature and for this reason, the result from each method is tested. If the test fails the impulse response is rejected and another method is attempted. If all methods fail the simulation will abort.

Transfer functions that decay slowly are likely to fail the inverse FFT method as this method requires that the impulse response decay to zero or nearly to zero. If the Stehfest method also fails, the problem can sometimes be resolved by increasing the inverse FFT size. This increases the duration over which the inverse FFT is evaluated so providing a longer time to decay.

In many situations the impulse response extraction fails because the transfer function is non-causal. This means that it has a response in negative time. Non-causal transfer functions can be made causal by adding a delay.

#### Impulse Response Cache

The impulse response is cached to speed up subsequent simulations with the same response. The cache is located here:

C:\Users\

where <login-name> is the name used to log in to your system. This can be deleted at any time. The cache is size limited to 250Mbytes by default. This can be changed using the option variable

LaplaceCacheSizeMBytes. Type 'Set LaplaceCacheSizeMBytes=nnn' at the command line to set a new value.

### Impulse Response Analytical Extraction

The impulse response analytical extraction attempts to match the transfer function to one of the following patterns. The parameters  $a, b, c$  etc represent constant values.

Laplace expression	Impulse response
$a$	$a.\delta(t)$
$a/(b.s)$	$a/b$
$a/\sqrt{b.s}$	$a/\sqrt{2.\pi.b.t}$
$a/(b.s + c)$	$a/b.e^{-c/b.t}$
$a/\sqrt{b.s + c}$	$a * e^{-c.t/b} / \sqrt{b.\pi.t}$
$\sqrt{a/(b.s + c)}$	$\sqrt{a} * e^{-c.t/b} / \sqrt{b.\pi.t}$
$\frac{a}{b.s^2 + c.s + d}$	$\frac{-(e^{-t/2(\sqrt{p^2-4q+p}} - e^{t/2(\sqrt{p^2-4q-p}})})}{\sqrt{p^2 - 4q}}$ $p = c/b$ and $q = d/b$
$a.e^{b.s}$	$a.\delta(b + t)$
$a/(b.s^c)$	$\frac{a.t^{c-1}}{b.\Gamma(c)}$
$\frac{a.s^2 + b.s + c}{d.s^2 + g.s + f}$	$[\frac{-(e^{t.arg2} - e^{t.arg3}) * (-p.u + 2q + u^2 - 2v)}{2arg1} - \frac{(u-p)(e^{t.arg2} + e^{t.arg3})}{2}] \frac{a}{d}$ $p = b/a$ $q = c/a$ $u = g/d$ $v = f/d$ $arg1 = \sqrt{u^2 - 4v}$ $arg2 = -arg1/2 - u/2$ $arg3 = arg1/2 - u/2$
$\frac{a.s + b}{c.s^2 + d.s + f}$	$\frac{a}{c} \cdot \frac{(2q - u)(arg2 - arg3) + \sqrt{u^2 - 4v}(arg3 + arg2)}{2\sqrt{u^2 - 4v}}$ $q = b/a$ $u = d/c$ $v = f/c$ $arg2 = e^{t.(\sqrt{u^2-4v}/2-u/2)}$ $arg3 = e^{t.(-\sqrt{u^2-4v}/2-u/2)}$
$\frac{a.s + b}{c.s + d}$	$\frac{e^{-d.t/c}(b.c - a.d)}{c^2} + \frac{a.\delta(t)}{c}$
$\frac{\sqrt{a.s + b}}{\sqrt{c.s + d}}$	$(\alpha.(I1(\alpha.t) - I0(\alpha.t)) + \delta(t)).e^{-\beta.t}.Y$ $Y = \sqrt{a/c}$ $\alpha = (d/c - b/a)/2$ $\beta = (d/c + b/a)/2$ $I0$ and $I1$ are Bessel functions

The analytic matching algorithm will attempt to match partial expressions as well as the whole expression if they are combined by addition, subtraction or multiplication. For example:

$$(1/(s + 1)) * exp(-s)$$

will be matched to the product of  $1/(s + 1)$  and  $exp(-s)$ . Respectively these have impulse responses of  $e^{-t}$  and  $\delta(1 + t)$  and those two impulse responses will be convolved to yield the final result. Note that this only succeeds if all partial expressions can be matched analytically.

To exploit this method, ensure that the expression is entered in a manner that keeps each partial expression distinct. For example, the following would be matched correctly:

$$1/(s + 1) * 1/(s + 2)$$

as this would be seen as the two partial expressions  $1/(s + 1)$  and  $1/(s + 2)$  each of which can be resolved analytically. The final result would be obtained by convolving the two impulse responses.

However the following mathematically identical expression would not be matched:

$$1/((s + 1) * (s + 2))$$

The analysis algorithm does not attempt to decompose denominator products so this will not be recognised analytically. The above would be extracted using inverse FFT which would usually be successful but might not be if the run time is considerably shorter than the time constant.

# Chapter 10

## Device Library and Parts Management

The electrical characteristics for semiconductor devices such as transistors and for more complex devices such as operational amplifiers are not built in to the simulator program but are defined in separate library files. These are text files containing .MODEL and .SUBCKT controls. Some libraries have been supplied with SIMetrix but you can obtain others from device manufacturers (usually at no cost) and other model vendors. You can also create them yourself using a text editor.

Many vendor libraries may be downloaded from the Internet. Our World Wide Web site carries a page with links to vendor sites. URL is:

*<http://www.simetrix.co.uk/app/models.htm>*

This section explains how to use, install and manage parts libraries.

**Important:** The library and parts management systems described in this chapter work with discrete devices defined using subcircuits and .MODEL statements. It currently does not support process corner selection and process binning used by many models supplied by integrated circuit process foundries. See [“Using Schematic Editor for CMOS IC Design” on page 102](#) for details on how to handle such libraries.

### 10.1 Using Model Library Browser

The model library browser provides a convenient method of selecting a part from the model library. Parts are arranged in categories to allow for rapid searching.

To open the model library browser select schematic menu **Place | From Model Library...** All devices for which models have been installed will be displayed and listed under an appropriate category.

If you can't find a device under the expected category, select the \* All Devices \* category. Every single device currently installed will be displayed here. (Note for large libraries you may have to wait a second or two to see the list of devices when selecting this category).

If you have installed your own models (see [“Parts Management - Installing Models” on page 209](#)) you will always find them listed under the category \* All User Models \* and, if installed within the last 30 days, under \* Recently Added Models \*.

If you select a part under \* All User Models \* or \* Recently Added Models \* you may be presented with the Associate Model dialog box. This will happen if SIMetrix is unable to determine what symbol to use for the model. This is explained in [“Placing New Model on Schematic” on page 211](#).

### 10.2 Parts Management - Installing Models

### 10.2.1 Overview

The process of installing third party SPICE models has always been a fundamentally tricky one.

The difficulty has been associating the SPICE model - which is the electrical definition of the device - with the schematic symbol - which is the pictorial representation of it.

A model provides an electrical description of the device but not what schematic symbol to use nor what category it should be in the model library browser. SIMetrix is able to determine this for itself if the device is implemented using a .MODEL statement as all .MODELs refer to a particular type of device (NPN, NMOS, Diode etc.). Devices implemented as subcircuits however remain a problem as there is nothing in a .SUBCKT definition which tells SIMetrix what the device is. For example a three terminal regulator and a power MOSFET use identical syntax - SIMetrix can't tell the difference from the syntax alone. To resolve this SIMetrix is shipped with a database of known part numbers providing a named schematic symbol, part category and if relevant a pin mapping order. If the part is in the database, no further action is required by the user and the part will appear in the browser under the correct category and select the correct symbol.

If the model is not in the database and has 2 or 3 terminals, then SIMetrix will attempt to determine the type of device by performing some tests on the model using simulation. If this process is successful, SIMetrix will choose an appropriate schematic symbol without further action required.

If SIMetrix cannot determine what the device is then, in order to use the device on a schematic, you will need to provide the association information. You will be prompted for this information when you place a part on the schematic for the first time and this is often the most convenient method. However, there is also a method of providing the association information in bulk which is advantageous in many cases.

### 10.2.2 Procedure

There are two stages to installing SPICE models.

1. Install the model itself so that the program knows where to look for it. This is simply a matter of dropping files or folders on the command shell. See [“Installing Electrical Model” on page 210](#).
2. Associate the model(s) with a schematic symbol. This process is often automatic and you don't need to do anything - see explanation in the [Overview](#). If this is necessary, you will be prompted for the information required. See [“Placing New Model on Schematic” on page 211](#).

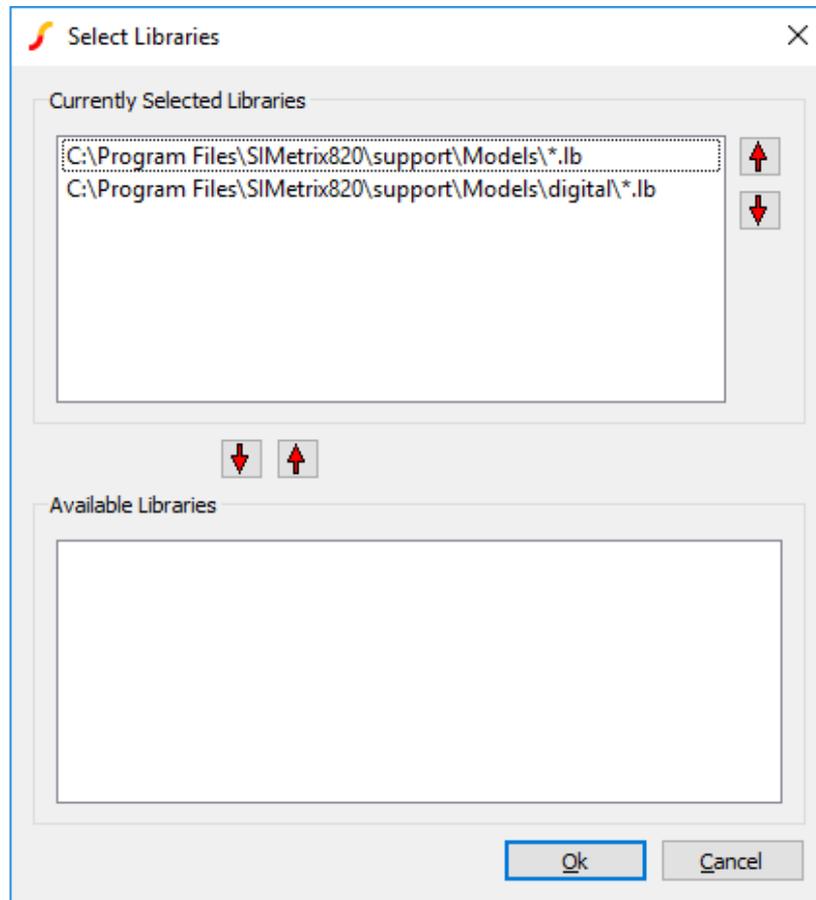
### 10.2.3 Full Model Installation Procedure

The following is the full procedure for installing models including association if required.

#### Installing Electrical Model

1. Open a suitable file manager program such as windows explorer in Windows. Locate the folder where your model files are located.
2. Select the items you wish to install. You can also install a single file, multiple files an entire folder or multiple folders. You only thing you can't do is install files and folders at the same time.
3. Make sure that the SIMetrix command shell is visible. If it is not visible, select menu **View | Show Command Shell**.
4. Pick up the items selected in 2. above and drop them into the message window of the command shell.
5. If you installed individual files, you will see a message box asking you to confirm that you wish to continue. Just click OK. The model files are now installed.

6. If you drop folders a search will be made in those folders for SPICE models. The Add/Remove Models dialog will then be displayed as shown below:



Select the items you wish to install in the lower box and transfer them to the upper box by pressing the Add button. You can also change the order of the items in the upper box. This affects the search order when a simulation is run. Click Ok. You will see a message displayed in the command shell ‘Model library changed. Rebuilding catalogs, please wait...’. When finished the message ‘Completed’ will be displayed. The electrical model or models are now installed.

## Placing New Model on Schematic

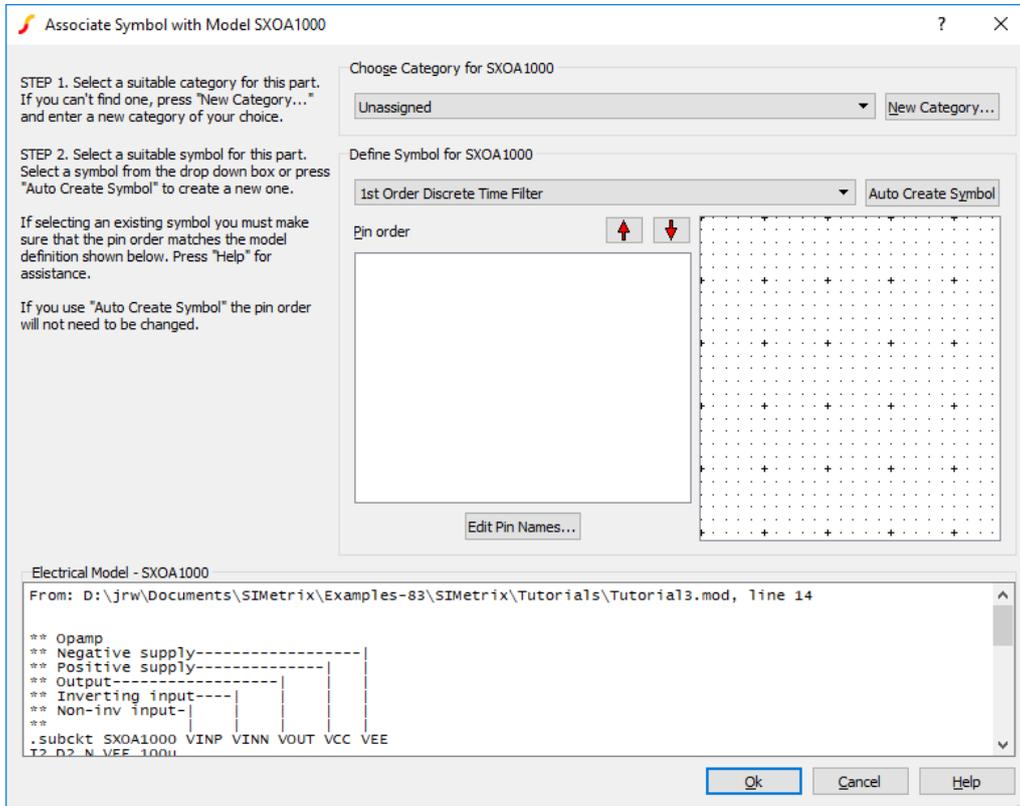
1. You can now place models installed using the model library browser. Select schematic menu **Place | From Model Library...** You will see a dialog box similar to that shown for the “[Model Library Browser](#)” on page 125.
2. On the left hand side select category ‘\* Recently Added Models \*’ or if the models were installed more than 30 days ago, select ‘\* All User Models \*’. You should see the models you have installed listed on the right hand side. Select the device you wish to place the press **Place**.
3. At this point, one of two things will happen. Either

A schematic symbol will appear (possibly after a short delay) for you to place on the schematic sheet. If so, no further action is needed after placing the symbol.

OR

If SIMetrix is unable to identify a suitable schematic symbol for the model, the associate models and symbols dialog box will open. See next step.

4. The following dialog box will be displayed:



5. Enter a suitable category for the part under **Choose Category** for *xxx* (where *xxx* is your model name). You can create a new category if desired by pressing **New Category...**
6. Using the drop down box under **Define Symbol** for *xxx*, select a suitable symbol for your model. An image of the symbol will be displayed so you can check if it is appropriate. If no suitable symbol is available, press **Auto Create Symbol** and one will automatically be created. You can edit this symbol later if required.
7. If you selected an existing symbol, you must check that the pin order matches that of the model itself. The model text is displayed under **Electrical Model - xxx**. If the pin order needs changing, use the up and down arrow keys to rearrange the pins as appropriate.
8. Click **Ok** then place symbol as usual.

Steps 4 to 8 above only need to be done once for each model.

## Note

If the message

```
Unknown file type xxx
```

is displayed when you drop a file, it means that no valid SPICE models were found in the file. It does not mean the file has the wrong extension. SIMetrix will accept any extension for model files with the exception of the extensions used for schematic or graph files (sch, sxsch and sxgph).

## 10.2.4 Removing Model Libraries

Select **File | Model Library | View/Remove Libraries...** . A dialog box similar to that shown in [“Installing Electrical Model” on page 210](#) will be displayed but with the Available Libraries box empty. Select the devices you wish to remove from the Selected Libraries box.

## 10.3 Parts Management - Configuring the Part Selector

### 10.3.1 Overview

The schematic part selector (see [“Part Selector” on page 124](#)) can be customised for your own application and preference. This can be done in two ways:

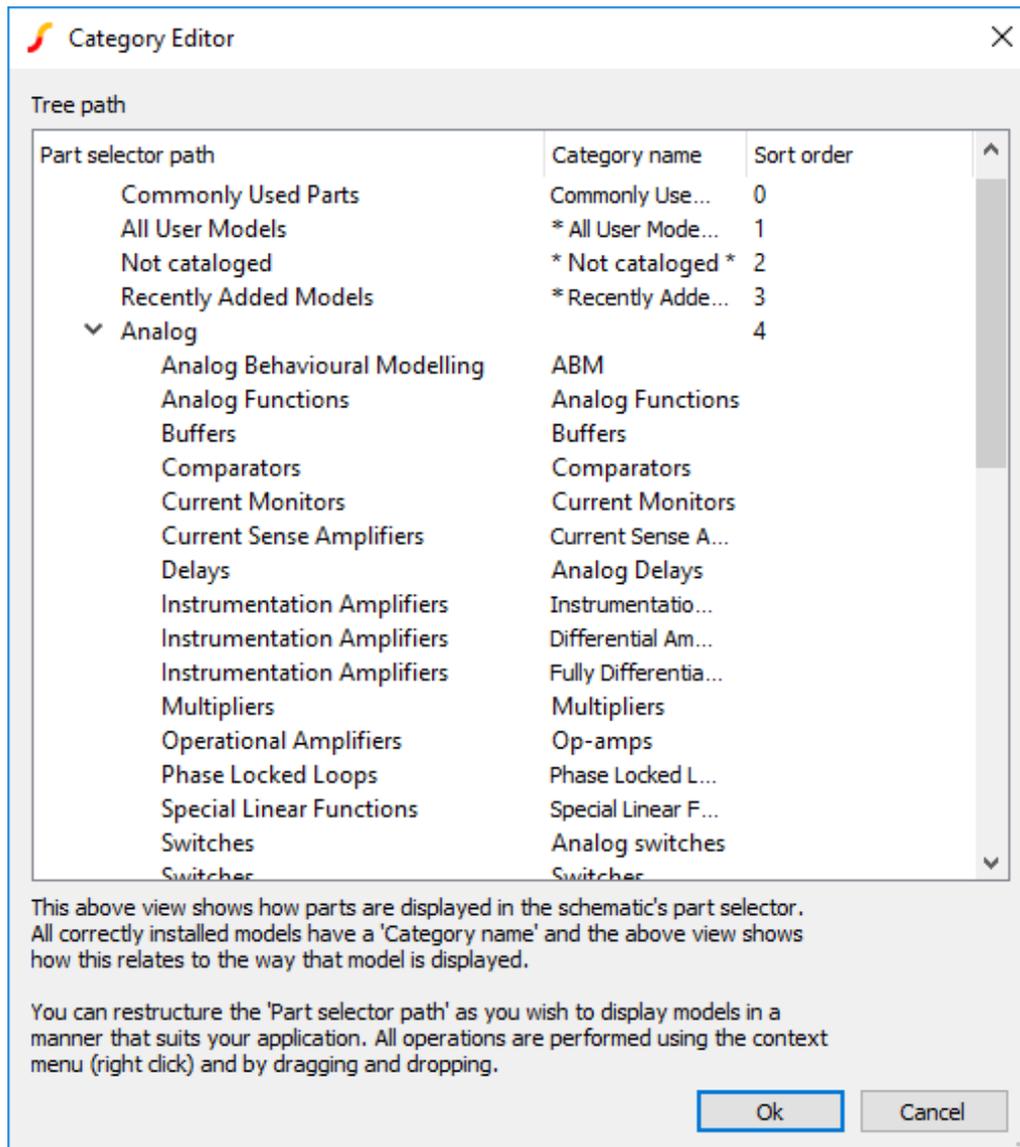
1. Category editor. This is a GUI tool that allows you to move categories of parts to different positions in the hierarchy. It can also hide them altogether.
2. Add, edit or remove individual items. There is no GUI to support this, but can be done by editing a catalog file.

### 10.3.2 Category Editor

The hierarchical structure of the part selector may be changed using the category editor.

All parts listed in the part selector are assigned category names. The category editor maps these category names to positions in the part selector hierarchy. For example, the category name **NMOS** maps to the location **Discretes** → **MOSFETs - N-channel**. So by using the category editor, it is possible to reorganise how parts are displayed. You can also remove categories from the part selector by moving them to the path called **\*\* HIDDEN \*\***.

To open the category editor, select menu **File | Model Library | Edit Part Selector Categories (SIMetrix Mode)...** or **File | Model Library | Edit Part Selector Categories (SIMPLIS Mode)...** This will open the GUI as shown below:



On the left hand side is a list of part selector paths. These are not individual parts but folders within the part selector that contain parts. On the right hand side are the category names. All parts displayed in the category editor have a category name.

With the category editor, you can:

1. Rename folder entries in the part selector. Select the entry then right click menu **Rename**.
2. Hide categories of parts that you don't use. Select the category you do not wish to show in the part selector. Drag and drop to the **\*\* HIDDEN \*\*** category path.
3. Move categories to new locations in the hierarchy. You can move categories to another existing location by simply dragging and dropping it. You can also create new folders within the category editor to move categories to. To do this, select an existing category folder or [ROOT] then right click menu **Create New Category Folder...**
4. Change order of categories. Select a category or category folder then right click menu **Edit Sort Order**. Enter a positive number - the smaller the number, the earlier in the list the item will be placed.

Items without a defined sort order will be placed below items with a sort order. Items with the same sort order will be sorted alphabetically.

5. Delete Categories and category folders. Note: you cannot delete categories and category folders that are built-in; you can only delete new categories or folders that you have created. If you want to remove items from the part selector, you can move the category to the **\*\* HIDDEN \*\*** folder. See item 2 above.

### 10.3.3 Editing the Part Selector Catalog

It is also possible to add your own parts to the part selector although there is no GUI available to do this.

To add parts to the part selector, create a file called

schematic\_generic\_parts\_tree\_journal\_simetrix.cat, for SIMetrix mode or

schematic\_generic\_parts\_tree\_journal\_simplis.cat, for SIMPLIS mode.

Each line of this file defines a single entry in the part selector. The line is a semi-colon delimited list of fields that define various aspects of the part. See the following table for a definition of each of these fields.

Field number	Function	Example
0	Name of part as it will appear in the place menu and hyperlink	DC Motor
1	Internal symbol name	dc_motor_symbol
2	unused - leave empty	
3	Category name	Motors
4	Name as it appears in the parts selector hierarchy	DC Motor
5	unused leave empty	
6	Script name - set to ps_simple	ps_simple
7	unused - leave empty	
8	Search keywords. Can add additional words to help user search. Separate multiple word with commas	electromechanical
9	Operation. Can be 'add', 'edit' or 'delete'. For creating new parts this must be 'add'	add

Using the example as shown in the above table, the line would be:

```
DC Motor;dc_motor_symbol;;Motors;DC Motor;;ps_simple;;electromechanical;add
```

You can create as many lines like the above as you wish, but it is important that each occupies a single line in the file.

The file should be placed at this location:

```
C:\Users\username\AppData\Roaming\SIMetrix Technologies\SIMetrix $nmn$ \devdb\user\
```

Where:

*username* is your login account name

*nmn* is the SIMetrix version as an integer, e.g. 800 for version 8.

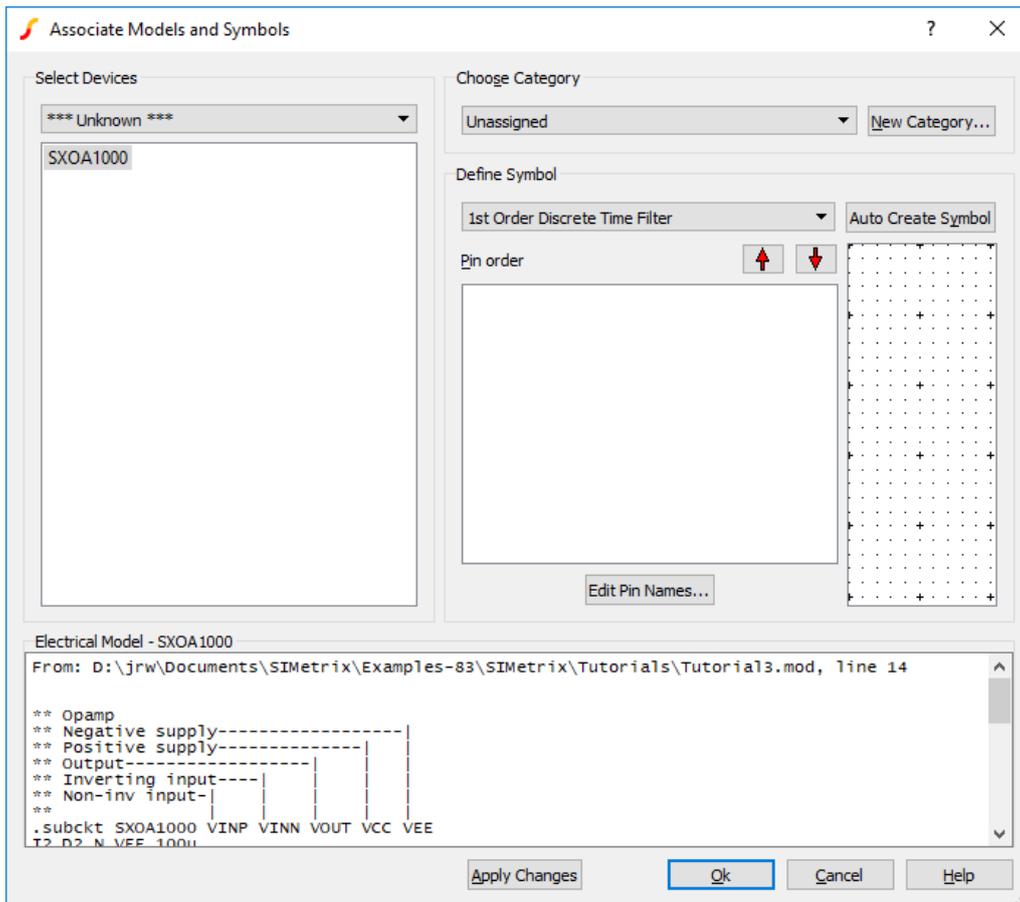
*ver* is the SIMetrix version in the form 8.00

## 10.4 Parts Management - Advanced Topics

### 10.4.1 Associating Multiple Models with Symbols

The previously described procedure for installing model libraries (see “[Full Model Installation Procedure](#)” on page 210) explains how to install the library and then, if required, associate each model individually as you place the device on the schematic.

In some situations you might wish to perform the association process in bulk, that is for many devices at once. To do this, use the **File | Model Library | Associate Models and Symbols...** menu. This is what you will see:



### Associate Model Dialog Box

In the top left hand group you select the device or devices that you wish to associate. The drop-down box at the top has a list of categories. Usually you would select a device or devices in the ‘\*\*\* Unknown \*\*\*’ category but you can also edit the association of known devices in other categories.

Once the category has been selected, a list of devices in that category will be displayed in the list box below. You should then select a device or devices to associate. To select multiple devices hold the **control** key down while selecting. Note that you will not be allowed to select multiple devices that have different numbers of pins. To help you determine what type of device it is, its electrical model is displayed in the window that covers most of the lower half of the dialog box. You must now define the Category, symbol and, if necessary, the pin order for the device. This is done using the top right hand group of controls titled

**Choose Symbol/Category.** Select an appropriate category and symbol. Note that only compatible symbols that have the same number of pins as the selected device will be shown. If an appropriate symbol is not available, you instruct SIMetrix to create one for you by pressing the **Auto Create Symbol** button. The symbol created will be functionally correct but of course its design and labelling may not be exactly what you would like. You can edit the pin names of the new symbol by pressing **Edit Pin Names...** . If other changes are required, you can edit the symbol using the graphical symbol editor at a later time.

The next list box allows the pin order to be changed. If you used the auto create symbol described in the above paragraph, you will not need to change the pin order. Even if you used an existing symbol from the drop down box you probably won't need to change the default as most devices such as opamps and MOSFETs use a de-facto standard pin order. Usually you can check the pin order from the **Electrical Model** display at the bottom of the dialog box. Many subcircuit definitions are preceded by text which identifies each connection to the sub circuit. *This must correspond exactly to the pin order of the symbol.* The names of symbol pins and the names used for the subcircuit terminations do not need to match; only the order is important. If the pin order does not match you can change it using the up and down arrow buttons. Simply select a pin in the list box then move it up or down the list. Note that the change will only apply to the device(s) you are currently editing; other devices associated with the same symbol will be unaffected.

Once you have finished selecting the category, symbol and pin mapping you *must* select the **Apply Changes** button. Your edits will be lost if you don't, but you will be warned about this before closing the box.

## 10.4.2 Embedded Association

It is possible to embed association information within the model file itself. This is useful if you wish to prepare a model to distribute to other users and wish to spare them the burden of performing the association process themselves. Models with embedded association can be installed by dropping their files in the command shell with no other action being required.

Only subcircuit devices may receive embedded association information. The information is placed in a specially formatted comment line after the `.SUBCKT` line but before the first device or command. The line is in the form:

```
*#ASSOC Category=category Symbol=symbol [Mapping=mapping]
```

<i>category</i>	Category for part. If it has spaces this must be enclosed in double quotation marks.
<i>symbol</i>	Internal symbol name to be used for part.
<i>mapping</i>	Mapping information. This changes the mapping between the subcircuit terminals and the symbol pin order. Usually, it's easiest simply to arrange the subcircuit pin order to match the symbol pin order in which case this is not required. If however there is some reason why rearranging the subcircuit pins is not desirable, you can instead specify the pin order using the mapping value.  The mapping value is a list of symbol pin numbers that match to the corresponding subcircuit terminal. So a mapping value of 2,3,1 says that the first subcircuit terminal connects to pin 2 of the symbol, the second subcircuit terminal connects to pin 3 and the third to pin 1.

## Example

```
.SUBCKT IRF530 D G S
*#ASSOC Category=NMOS Symbol=nmos_sub
```

```
...
.ENDS
```

## Priorities

It's possible that association information could be provided from multiple sources in which case the possibility of conflict arises. If this is the case the following priorities apply:

1. User supplied association (e.g. using the associate symbols and models dialog) takes precedence over embedded association.
2. Embedded association takes precedence over pre-defined association. Pre-defined association is what is stored in the all.cat catalog file supplied with SIMetrix.

### 10.4.3 Catalog Files

The data for model and symbol associations are stored in catalog files. There are three catalog files as follows:

ALL.CAT	Resides in support\devdb under the SIMetrix root directory. Stores catalog data supplied with SIMetrix. SIMetrix never modifies this file.
USER_V2.CAT	Resides in devdb/user under the application data directory (see <a href="#">“Application Data Directory” on page 444</a> ). Stores catalog data supplied by the user. Data in this file overrides data in ALL.CAT. The Associate Models dialog box writes to this file. In SIMetrix versions 5.2 and earlier this file was called USER.CAT. SIMetrix will automatically import data from USER.CAT to USER_V2.CAT if USER.CAT is present.
OUT.CAT	Resides in devdb/working under the application data directory (see <a href="#">“Application Data Directory” on page 444</a> ). This is what is actually used by the model library browser to select and place parts. It is generated by the associate model dialog box from information in ALL.CAT, USER_V2.CAT and installed models. It will also be automatically created by the model library browser if it does not already exist. You can also force it to be rebuilt at any time by selecting menu <b>File   Model Library   Re-build Catalog</b> .

## File Format

Catalog files are text files. Each line provides data about a single device in semi-colon delimited fields. The fields are as follows:

Field 1	Device name as it appears in browser. This may optionally be followed by a comma followed by the number of terminals for the model.
Field 2	Symbol name
Field 3	MODEL property - X for subcircuits, as appropriate for other devices. (This field is empty in ALL.CAT and USER_V2.CAT it is determined automatically from electrical model when OUT.CAT is built).
Field 4	Category
Field 5	Sub-category (currently not used)
Field 6	Pin mapping order
Field 7	Not used

When you select OK your edits will be written to the USER\_V2.CAT file (see above table). This is in the same format as ALL.CAT in the root folder. ALL.CAT is never modified. Also another file is updated called OUT.CAT. This is the file used by the model library browser. The process of building OUT.CAT may take a few seconds if the model library is large.

### 10.4.4 Importing Models to a Schematic

SIMetrix provides a means to automatically import all models needed for a schematic into that schematic. The models are placed in the simulator command window (opened with **F11** see “[Manual Entry of Simulator Commands](#)” on page 42). Once the models are imported to a schematic, it will no longer be necessary for SIMetrix to locate the models in the library when a simulation is run. This has the following benefits:

- It makes the schematic completely self-contained. This is useful for archiving or if you wish to pass the schematic to a third party.
- You can edit the models locally without affecting the global library.

To import models to a schematic, select the schematic menu **Simulator | Import Models...**. You will be provided with two options: **Import Direct Copy** and **Import by Reference**. The first will import the model text directly into the schematic. The second will put the model text into a file. This will be referenced in the schematic’s simulator command window using a .INC control. See *Simulator Reference Manual/Command Reference/.INC*.

## 10.5 Sundry Topics

### 10.5.1 .LIB Control

The .LIB netlist control allows the local specification of model library for a particular circuit.

Syntax:

```
.LIB pathname
```

pathname	File system path name specifying a single file or, by using a wildcard (* or ?), a group of files. If the path name contains spaces, it must be enclosed in quotation marks (").
----------	--

This control specifies a pathname to be searched for model and subcircuit libraries. Any number of .LIB controls may be specified and wild-cards (i.e. \* and ? ) may be used.

If a model or subcircuit is called up by a device line but that definition was not present in the netlist, SIMetrix will search for it in files specified using the .LIB control.

SIMetrix also supports another form of .LIB used by model files designed for Hspice ®. See the *Simulator Reference Manual/Command Reference/.LIB/Hspice Form* for details.

### 10.5.2 Drag and Drop to Schematic

You can install a model file to a schematic by picking it up in windows explorer and dropping it onto the schematic window. This will insert a .LIB control (see above) with a path to the file you dropped. This installs the model file to be local to that schematic.

### 10.5.3 Library Diagnostics

When enabled, library diagnostics display messages showing the progress of the location of device models. To enable/disable select **File | Options | General...** then **Model Library** tab.

### 10.5.4 Local Models

You can also enter a model or subcircuit definition in the schematic's **F11** window. However if you enter a model in this manner it will only be available to that schematic.

### 10.5.5 Library Indexing Mechanism

This is a technique used to speed the search for models and subcircuits. It is completely transparent and requires no action from the user. SIMetrix creates an index file for each library specification it encounters either installed globally or referenced using **.LIB**. This index files contain details of the file locations of models and subcircuit definitions referenced by the library specification. These index files can then be used for later simulation runs to speed the search for models and subcircuits. Index files are automatically rebuilt if any of the library files referenced are modified. (Modifications are detected by comparing file dates). All index files are stored in

*app\_data\_dir*/INDEXES

where *app\_data\_dir* is the location of the application data directory. See "[Application Data Directory](#)" on [page 444](#) for the location of this directory. The files are named *SXn.sidx*, where *n* is some number.

Note that if you add a new model file to a directory while SIMetrix is running, SIMetrix won't know of the new file and any relevant indexes won't be updated. In this situation, select the menu **File | Model Library | Rebuild Catalog** to update the indexes.

### 10.5.6 Duplicate Model Names

Models of some common parts are available from different sources. Sometimes these have different names, e.g LF356 and LF356/NS - the latter available from the National Semiconductor library. In some cases the model names from different sources are identical. This poses a problem as models have to be uniquely identified by their name.

SIMetrix has a built-in utility that can automatically rename models with duplicate names. The devices are renamed by adding a user specified suffix to the model name. The rename utility is not accessible via the menus but must be invoked by typing a command at the command line. Proceed as follows:

1. First ensure that all the model library files you wish to process are installed as global libraries.
2. Make backup copies of your model files. This is optional, the utility makes backups anyway.
3. Type the following at the command line. The command line is the edit box at the top of command shell. If this is not visible, select menu **View | Show Command Shell**:

```
rename_libs
```

4. A list of currently installed libraries will be displayed. Double click on any that you wish to be processed for renaming and supply a suffix. The suffix must not contain spaces and should start with a non-alphanumeric character such as '/' or '-'. Note that only models found to have duplicates will be renamed. SIMetrix will not rename unique models. If you do not supply a suffix for a library, no devices within it will be renamed.
5. Press OK. The operation can take a long time; possibly a few minutes if the library is large. On completion the message:

```
*** RENAME COMPLETE. See ???/RENAME.LOG for details
```

will be displayed in the command shell. The RENAME.LOG file will contain full details of the rename process. This includes details of all models that were renamed.

## Notes

- If the device being renamed is implemented as a subcircuit, the rename utility will copy any symbol/model association for that device with the new name.
- Devices that are used locally, i.e. within the model file itself, will be excluded from the rename procedure. These devices will not be renamed nor will they be added to the list that is searched to identify duplicate names.
- You can perform a test run which creates the log file but does not actually perform the renaming. To do this, type the command:

```
rename_libs_check
```

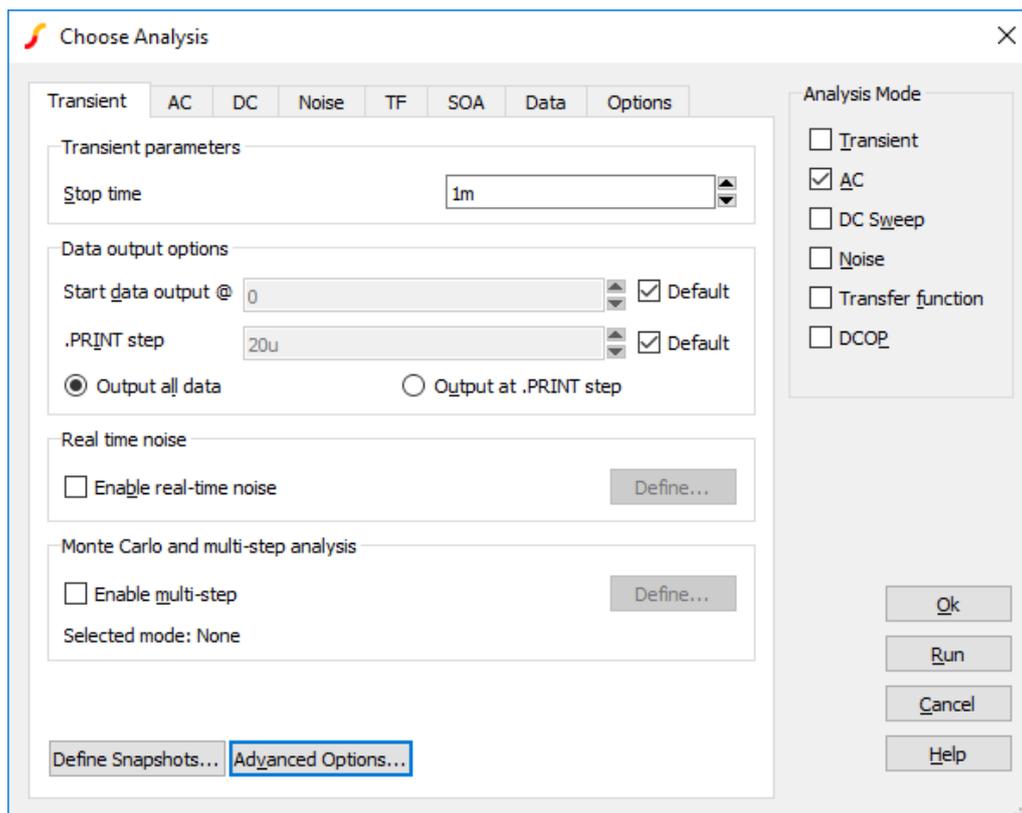
Note that messages output to the log file and to the command shell will report the renaming of models but no renaming will actually take place.

# Chapter 11

## Analysis Modes

In this chapter we describe the various analysis modes available and how to set them up from the schematic editor. There is more information on analysis modes including full details of the netlist commands to invoke them, in *Simulator Reference Manual/Command Reference*.

Most of the analyses can be setup using the Choose Analysis Dialog Box which is opened with the schematic menu **Simulator | Choose Analysis....**



You can also enter the 'raw' netlist commands in the **F11** window. The contents of this window remain synchronised with the Choose Analysis dialog box settings so you can freely switch between the two methods. The Choose Analysis dialog box does not support sensitivity and pole-zero analysis so these methods must be set up using the **F11** window.

## 11.1 Running Simulations

### 11.1.1 Overview

Once an analysis has been set up using the procedures described in this chapter, a simulation would normally be run in *synchronous* mode perhaps by selecting the **Simulator | Run Schematic** menu. In *synchronous* mode, you cannot use any part of the program while the simulation is running.

There are also other methods of running a simulation. You can run a simulation for a netlist directly and you can also run in *asynchronous* mode. These are explained in the following sections.

### 11.1.2 Starting, Pausing and Aborting Analyses

#### Starting an Analysis

To start a simulation in normal (synchronous) mode, use the Simulator | Run menu, press the **F9** key or press the **Run** button on the Choose Analysis Dialog box shown above. A dialog box will show the status of the simulation.

#### Pausing an Analysis

You can pause the simulation by selecting the **Pause** button on the simulator status dialog box. To restart select the **Resume** button (the **Pause** button changes name when simulation pauses) or the **Simulator | Resume** menu item.

When a simulation is paused, you can carry on using the program as if the simulation had completed. This includes plotting results of the simulation completed so far. If you decide you do not wish to continue the run, there is no need to explicitly abort it. You can just start a new run in the normal way. If you do this you will be asked if you would like to resume the pending run. If you answer 'No', the pending run will be automatically aborted and the new run started.

#### Aborting an Analysis

There is actually never a need to explicitly abort an analysis. If you decide you do not wish to continue a run, just pause it as described above. Pause is the same as abort except that you have the option to change your mind and restart.

Nevertheless there is an abort facility. Simply select the **Simulator | Abort** menu. When you abort a run, you will not be able to restart it.

There is just one benefit of aborting a run instead of pausing it. When an analysis is aborted, the simulator frees up the memory it needed for the run. Note that this does not happen after a run completes normally. If you need to free up simulator memory after a normal run completes, type **Reset** at the command line. (Not available with SIMetrix/SIMPLIS Elements).

#### Working while an Analysis is Running

Some operations can be performed while a simulation is running. For example, you can zoom in and out of graphs and many schematic editing operations can also be performed. However, some operations are blocked and full functionality will not be available until the simulation has completed or has been paused. If you need full GUI functionality while a simulation is running, you can run the simulation in the background. See "[Running Analyses in Asynchronous Mode](#)" on page 224.

### 11.1.3 Running Analyses in Asynchronous Mode

In *asynchronous* mode, the simulation runs in the background and you are free to carry on using the SIMetrix environment for entering schematics or viewing results from previous analyses. Because, the simulation is running in the background, it is necessary for the simulation process to be detached from the front end environment and for this reason it is not possible to use .GRAPH or fixed probes to plot simulation results during the course of the run. Also you must manually load the simulation data when the run is complete.

#### Starting an Asynchronous Run

1. Select menu **Simulator | Run Asynchronous...**. Note a simulation status box appears similar to the box used for synchronous runs but with an additional *Activity* box at the bottom. Any messages generated by the simulator will be displayed here.
2. When the simulation is complete, you must load the data manually. The name of the file to load will be displayed in the command shell when the simulation starts. Select menu **File | Data | Load Temporary Data...** to load data file. You will be able to cross probe the schematic used to run the analysis in the normal manner once this file is loaded.

#### Pausing and Aborting Asynchronous Runs

To pause, press the **Pause** button. Note that you can load the data generated so far after pausing the run as described above.

To abort a run, press the **Close** button.

### 11.1.4 Running an Analysis on a Netlist

You can run an analysis on a netlist created by hand or perhaps with a third party schematic entry program.

To run a netlist in synchronous mode, select the command shell menu **Simulator | Run Netlist...** then locate the netlist file.

To run a netlist in asynchronous mode, select the command shell menu **Simulator | Run Netlist Asynchronous...** then locate the netlist file. See [“Running Analyses in Asynchronous Mode” on page 224](#) for further information about running asynchronous analyses.

### 11.1.5 Simulation and Multi-core Processors

If your license permits it, SIMetrix will create multiple threads running on individual cores to help speed up the simulation. Please note the following in relation to multiple core operation:

1. Not all products support multiple core operation. See list below:

Product Name	Number of cores supported
SIMetrix Classic	1
SIMetrix/SIMPLIS	1
SIMetrix Pro	4
SIMetrix/SIMPLIS Pro	4
SIMetrix Elite	16

Product Name	Number of cores supported
SIMetrix/SIMPLIS Elite	16
SIMetrix/SIMPLIS Elements (free version)	1

- If you are running a Multi-step analysis such as Monte Carlo, the greatest benefit from multiple cores is achieved by using Multi-core Multi-step mode. See [“Using Multiple Cores for Multi-step Analyses”](#) on page 262.
- SIMetrix will only use a single core for small circuits even if you have multiple physical cores and a SIMetrix product that supports multiple core operation. Below a certain circuit size, multiple core operation can actually slow down the simulation.
- Speeding up simulations by employing multiple cores is analogous to speeding up human projects by allocation more people to them. That is, results vary enormously according to the circuit being simulated. Circuits that benefit the most from multiple core operation are large and typically contain many of the same type of device. This is typical for integrated circuit design. Smaller circuits that contain a variety of different types of device don’t benefit nearly as much.
- If you have a system fitted with multiple chips (referred to as physical processors), SIMetrix will only use the cores from one of the physical processors by default. For example, you may have an 8 core system with two physical processors of 4 cores each. In this situation, SIMetrix will use the 4 cores from the first physical processor and not use the second physical processor. This is a consequence of the cached hierarchical memory systems used on modern systems. You can force SIMetrix to use all available cores in all physical processors using the `mpnumthreads` option. See below.
- Many systems use hyperthreading to make a single core behave like two cores. The resulting cores are referred to as *logical* cores. The actual hardware cores are referred to as *physical* cores. SIMetrix will only use *physical* cores for multiple core operation.
- See **Help | About...** to find the specification of your system. The number that is most important is the *Number of physical cores*.
- The SIMPLIS simulator does not currently exploit the use of multiple cores for single step runs. It can, however, run multi-step analyses using multiple cores. See [“Multi-core Multi-step SIMPLIS Analyses”](#) on page 280.
- You can manually control the number of cores used for a single step simulation using the `mpnumthreads` option. Enter this line into the netlist or **F11** Window:

```
.option mpnumthreads=n
```

Where  $n$  is the number of cores you wish to use. SIMetrix will not use more than the number of physical cores fitted to your system regardless of what you set for  $n$ .

## 11.2 Transient Analysis

In this mode the simulator computes the behaviour of the circuit over a time interval specified by the stop time. Usually, the stop time is the only parameter that needs specifying but there are a number of others available.

### 11.2.1 Setting up a Transient Analysis

- Select menu **Simulator | Choose Analysis...**
- Select Transient check box on the right.

3. Select Transient tab at the top. Enter parameters as described in the following sections.

## Transient Parameters

Enter the stop time as required. Note that the simulation can be paused before the stop time is reached allowing the results obtained so far to be examined. It is also possible to restart the simulation after the stop time has been reached and continue for as long as is needed. For these reasons, it is not so important to get the stop time absolutely right. You should be aware, however, that the default values for a number of simulator parameters are chosen according to the stop time. (The minimum time step for example). You should avoid therefore entering inappropriate values for stop time.

## Data Output Options

Sometimes it is desirable to restrict the amount of data being generated by the simulator which in some situations can be very large. You can specify that data output does not begin until after some specified time and you can also specify a time interval for the data.

### Output all data/Output at .PRINT step

The simulator generates data at a variable time step according to circuit activity. If Output all data is checked, all this data is output. If Output at .PRINT step is checked, the data is output at a fixed time step regardless of the activity in the circuit. The actual interval is set by the .PRINT step. This is explained below.

If the Output at .PRINT step option is checked, the simulator is forced to perform an additional step at the required interval for the data output. The fixed time step interval data is not generated by interpolation as is the case with generic SPICE and other products derived from it.

### Start data output @

No simulation data will be output until this time is reached.

### .PRINT step

.PRINT is a simulator command that can be specified in the netlist to enable the output of tabulated data in the list file. See *Simulator Reference Manual/Command Reference/.PRINT* for details of .PRINT.

The value specified here controls the interval used for the tabulated output provided by .PRINT but the same value is also used to determine the data output interval if Output at .PRINT step is specified.

## Real Time Noise

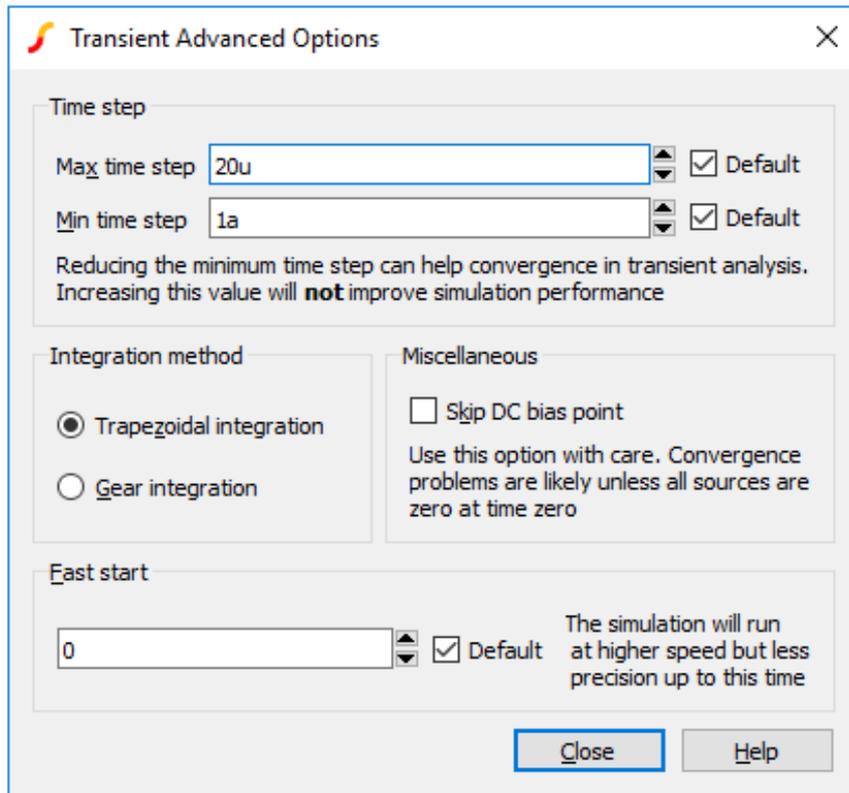
See “Real Time Noise” on page 246.

## Monte Carlo and Multi-step Analysis

See “Multi-step Analyses” on page 259

## Advanced Options

Opens a dialog as shown below



## Time Step

The options presented here set upper and lower limits to the time step used in transient analysis. It is important to note that the behaviour of the lower limit is different to the upper limit. The actual time step used is controlled by the simulator to ensure convergence and to satisfy tolerance requirements. It will not set a limit higher than the upper limit, but if it needs to set a time step below the lower limit, the simulation will abort; it will not simply hold the time step at the lower limit.

For this reason there is not usually any benefit in changing the minimum time step except perhaps to reduce it to fix a convergence problem. This is especially the case if a higher precision iteration mode has been selected (See “[Iteration Modes](#)” on page 253). It is possible that a smaller minimum time step can resolve problems that lead to a “Time step too small” error.

## Integration Method

Set this to **Gear** if you see an unexplained triangular ringing in the simulation results. Always use **Trapezoidal** for resonant circuits. A full discussion on integration methods is given in the *Simulator Reference Manual: Convergence, Accuracy and Performance/Accuracy and Integration Methods/Integration Methods - METHOD option*.

## Skip DC bias point

If checked, the simulation will start with all nodes at zero volts. Note that unless all voltage and current sources are specified to have zero output at time zero, the simulation may fail to converge if this option is specified.

## Fast start

The accuracy of the simulation will be relaxed for the period specified. This will speed up the run at the expense of precision.

This is a means of accelerating the process of finding a steady state in circuits such as oscillators and switching power supplies. Its often of little interest how the steady state is reached so precision can be relaxed while finding it.

Note that the reduced precision can also reduce the accuracy at which a steady state is found and often a settling time is required after the fast start period.

### 11.2.2 Restarting a Transient Run

After a transient analysis has run to completion, that is it has reached its stop time, it is still possible to restart the analysis to carry on from where it previously stopped. To restart a transient run:

1. Select the menu **Simulator | Restart Transient...**
2. In the **New Stop Time** box enter the time at which you wish the restarted analysis to stop. Click **Ok** to start run.

## See Also

*Simulator Reference Manual/Command Reference/.TRAN.*

### 11.2.3 Transient Snapshots

#### Overview

There is often a need to investigate a circuit at a set of circuit conditions that can only be achieved during a transient run.

For example, you might find that an amplifier circuit oscillates under some conditions but these conditions are difficult or impossible to create during the bias point calculation that usually precedes a small-signal AC analysis.

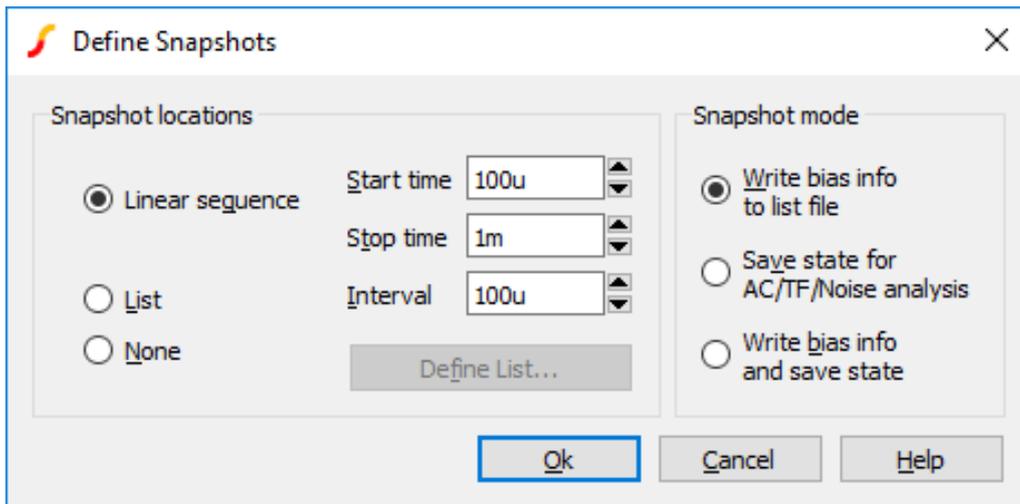
Transient snapshots provide a solution to this problem. The state of the circuit at user specified points during a transient run may be saved and subsequently used to initialise a small-signal analysis. The saved state of the circuit is called a snapshot.

Snapshots can be created at specified intervals during a transient run. They can also be created on demand at any point during a transient run by first pausing the run and then manually executing the save snapshot command. So, for example, if you find your amplifier reaches an unstable point during a transient analysis, you can stop the analysis, save a snapshot and then subsequently analyse the small signal conditions with an AC sweep.

An option is also available to save the DC operating point data to the list file at the point at which snapshots are saved.

#### Defining Snapshots Before a Run Starts

1. Select menu **Simulator | Choose Analysis...**
2. In the Transient Sheet select button **Define Snapshots...**
3. You will see the following dialog



Select either **Linear sequence** or **List** to define the time points at which the snapshots are saved. In the Snapshot mode box select one of the three options:

- **Write bias info to list file.** Instructs the simulator to write the DC operating point data to the list file. Does *not* save snapshot data.
- **Save state for AC/TF/Noise analysis.** Instructs the simulator to save snapshot data only. No bias point information will be output to the list file.
- **Write bias info and save state.** Performs both operations described in 1 and 2 above.

## Creating Snapshots on Demand

You can create a snapshot of a transient run after it has started by executing the SaveSnapShot script command. Proceed as follows:

1. Pause the current transient analysis or allow it to finish normally. You must not abort the run as this destroys all internal simulation data.
2. Type at the command line (the edit box at the top of the command shell):

```
SaveSnapShot
```

That is all that is needed. You can now start a new small signal analysis using the snapshot created.

## Applying Snapshots to a Small Signal Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select AC, TF or Noise analysis
3. Press **Define Multi-step Analysis** for the required analysis mode.
4. Select Snapshot mode.

The analysis will be repeated for all available snapshots.

## Important Note

Snapshot data can only be applied to an identical circuit to the one that created the snapshot data. So, you must make sure that any parts needed for a small-signal analysis that uses snapshot data are already present in the circuit before the transient run starts. In particular, of course, you must make sure that an AC source is present.

An error message will be output if there are any topological differences between the circuit that generated the snapshot data and the circuit that uses it. If there are only part value or model parameter differences, then the snapshot data may be accepted without error but at best the results will need careful interpretation and at worst will be completely erroneous. Generally, if you change a part that affects the DC operating point then the results will not be meaningful. If you change only an AC value, e.g. a capacitor value, then the results will probably be valid.

## How Snapshots are Stored

The snapshot data is stored in a file which has the default name of netlist.sxsnp where netlist is the name of the netlist used for the simulation. When using the schematic editor, this is usually design.net so the usual name for the snapshot file is design.sxsnp. You can override this name using the SNAPSHOTFILE OPTIONS setting although there is rarely any reason to do this.

The snapshot file is automatically deleted at the start of every transient run. The *SaveSnapShot* command always appends its data to the snapshot file so that any pre-defined snapshots are preserved.

When snapshot data is applied to a subsequent small-signal analysis, the snapshot file is read and checked that it is valid for the circuit being analysed.

## 11.3 Operating Point

To specify a DC operating point analysis check DCOP. Note that an operating point is performed automatically for all analysis modes and this is only useful if it is the only analysis specified.

Operating point analysis does not have any additional parameters so there is no tab sheet for it.

### See Also

*Simulator Reference Manual/Command Reference/.OP*

[“Viewing DC Operating Point Results” on page 373](#)

## 11.4 Sweep Modes

Each of the analysis modes DC, AC, AC Noise and Transfer Function are swept. That is they repeat a single analysis point while varying some circuit parameter. There are 7 different sweep modes that can be applied to these analyses. These modes are also used to define multi step analyses which are explained in [“Multi-step Analyses” on page 259](#). The 7 modes are:

- Device
- Temperature
- Parameter
- Model parameter
- Frequency (not applicable to DC)
- Monte Carlo

- Sensitivity/Worst-case

As well as 7 different modes there are 3 different sweep methods

- Linear
- Decade
- List

Dialog support for the List method is only available for the definition of Multi-step analyses. The simulator also offers an Octal sweep method but this is not supported by the Choose Analysis Dialog.

Each of the sweep modes is explained in more detail below.

### 11.4.1 Device Sweep

In this mode the principal value of a single device is swept. The analysis definition must specify the part reference for the device. The following types of device may be used.

Device	Value swept
Capacitor	Capacitance
Diode	Area
Voltage controlled voltage source	Gain
Current controlled current source	Gain
Voltage controlled current source	Transconductance
Current controlled voltage source	Transresistance
Current source	Current
JFET	Area
Inductor	Inductance
Bipolar Transistor	Area
Resistor	Resistance
Lossless Transmission Line	Impedance
Voltage source	Voltage
GaAs FET	Area

### 11.4.2 Temperature

Global circuit temperature is swept.

### 11.4.3 Model Parameter

Similar to device sweep except applied to a named model parameter. Both the model name and the parameter name must be specified.

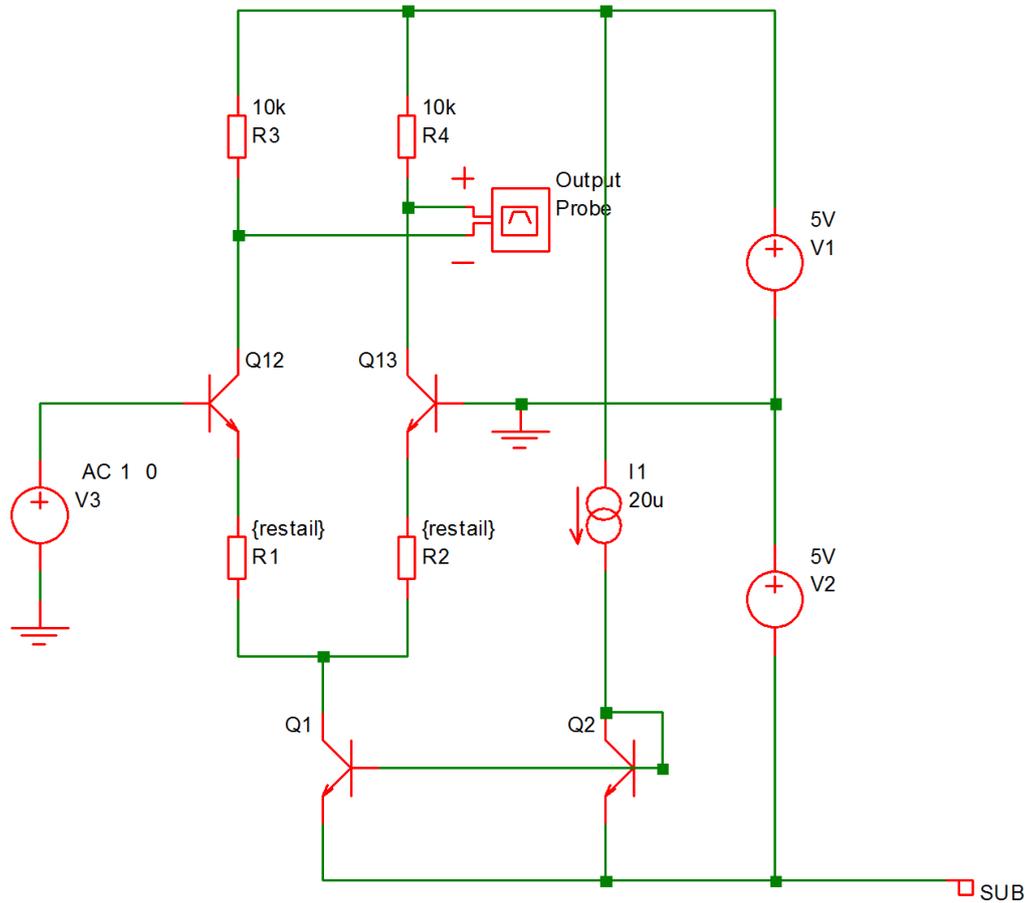
### Special Note

It is recommended that any model parameter being swept is also specified in the .MODEL parameter list. In most cases it isn't actually necessary but there are a few instances - such as for terminal resistance

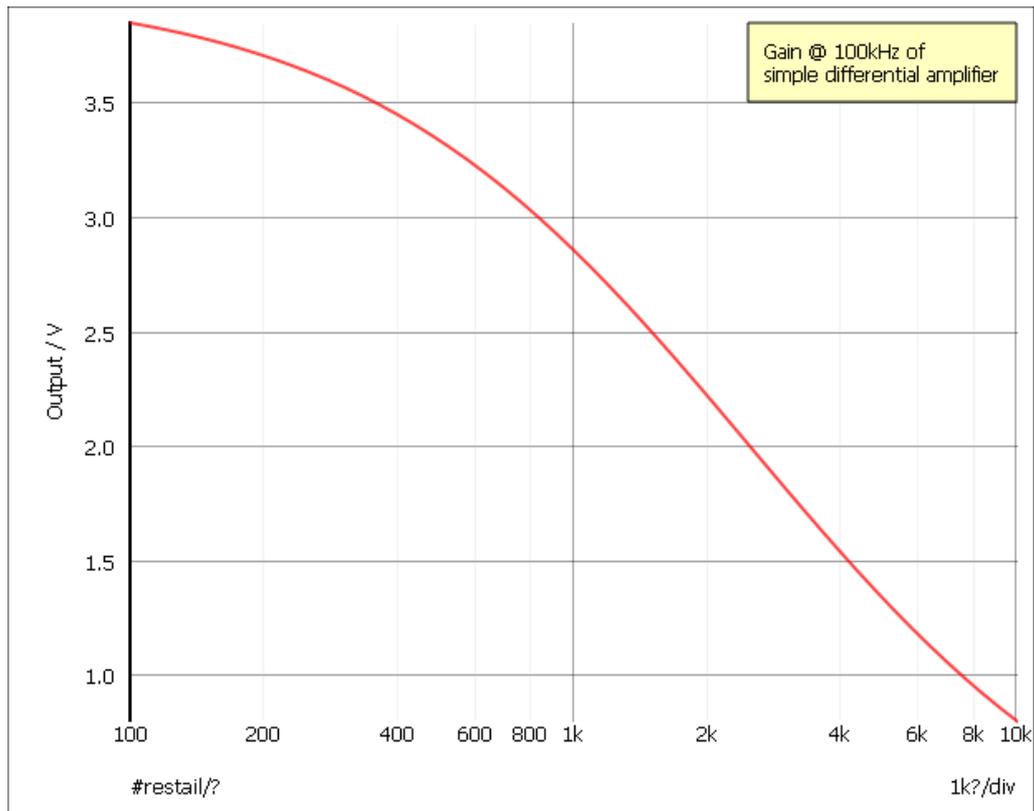
parameters - where it is necessary.

#### 11.4.4 Parameter

A user named variable that can be referenced in any number of expressions used to define model or device parameters. Here is an example. (See Examples/Sweep/AC\_Param.sxsch)



This is a simple long tailed pair. The above circuit resistors R1 and R2 have been given the values {restail}. *restail* is a parameter that is swept in an AC sweep to plot the gain of the amplifier vs tail resistance at 100kHz. Here is the result of the run:

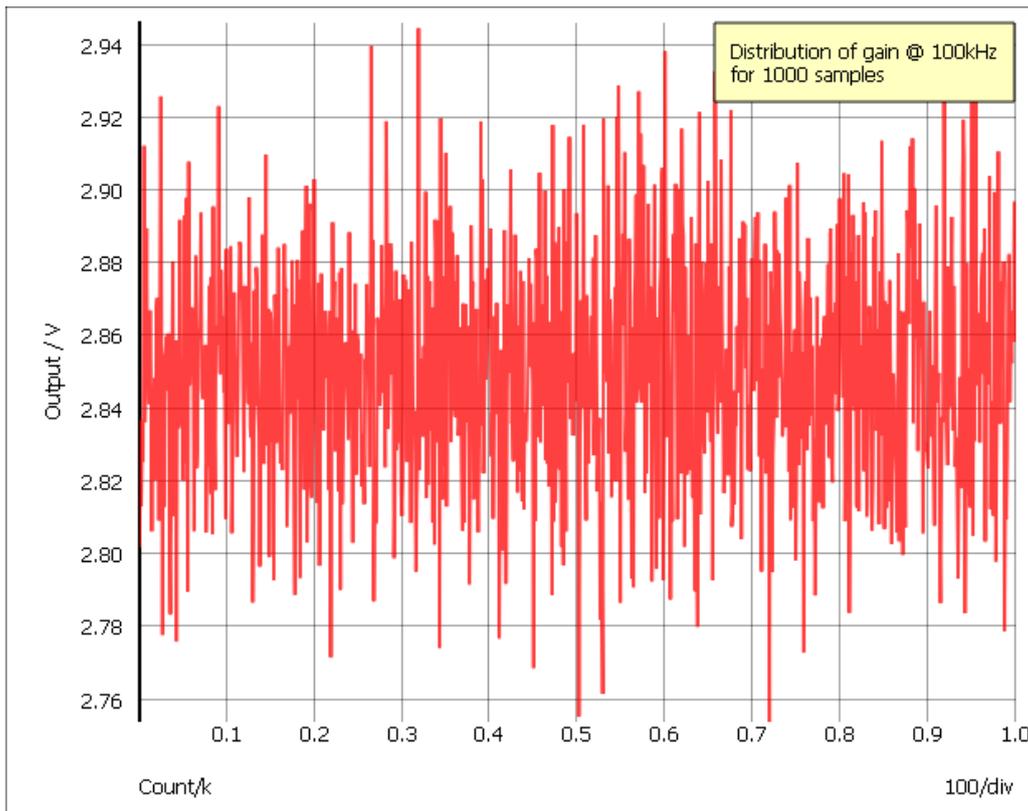


### 11.4.5 Frequency

Sweeps frequency for the small signal analysis modes namely AC, AC Noise and Transfer Function. In standard SPICE it is the only sweep mode available for AC and Noise while Transfer Function can not be swept at all.

### 11.4.6 Monte Carlo

Repeats analysis point for a specified number of times with device tolerances enabled. The following graph show the result for the same circuit as shown above but with restail=1k and with a 1000 point Monte Carlo AC sweep. This run takes a fraction of a second on any modern machine:



The graph shows the variation in gain for 1000 samples. Using the histogram feature a statistical distribution of the above can easily be plotted.

### 11.4.7 Sensitivity/Worst-case

Sensitivity and Worst-case are really two separate modes, but they are always selected together so we treat them as a single mode.

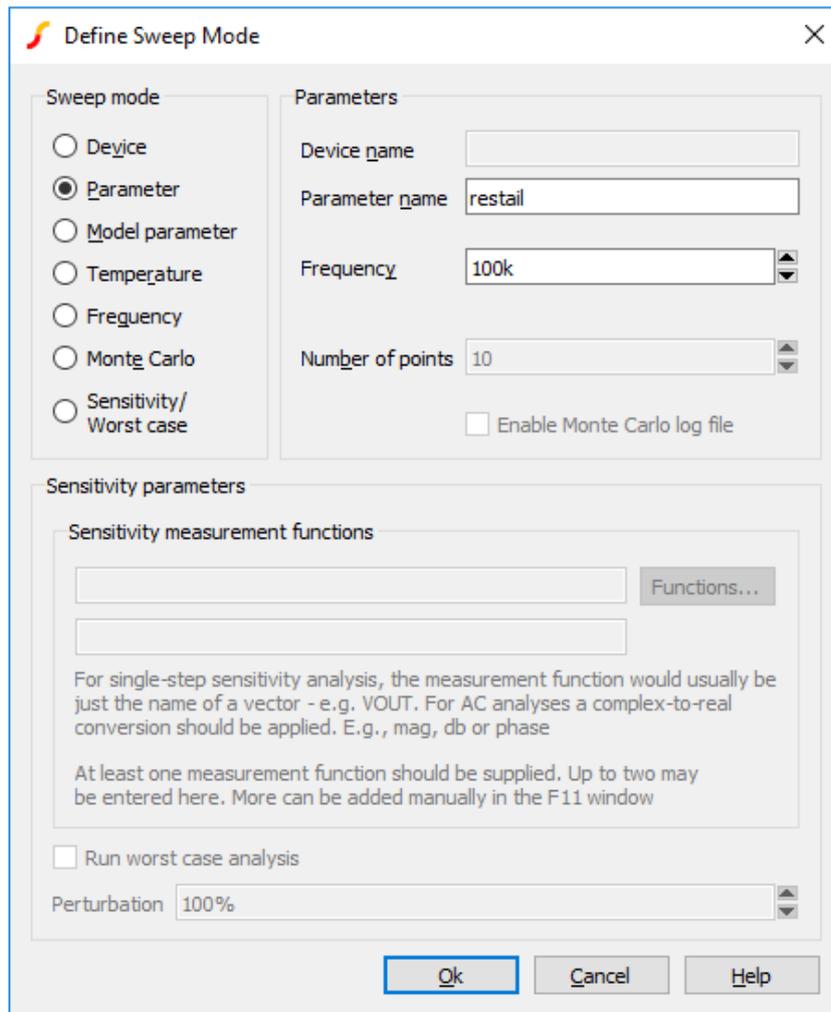
Sensitivity sweep, repeats an analysis point while varying each parameter defined by a distribution function in turn.

Worst-case sweep uses data obtained from a sensitivity sweep to identify what combination of parameter variations will yield the worst result. Two points are run with the combinations at each extreme.

For further information, see [“Setting up a Single Step Sensitivity Analysis”](#) on page 425.

### 11.4.8 Setting up a Swept Analysis

In the AC, DC, Noise or Transfer Function analysis sheets, select the Define... button in the Sweep Parameters box. This will bring up the following dialog



Select the desired mode on the left then enter the necessary parameters on the right. The parameters required vary according to the mode as follows:

Mode	Parameters
Device	Device part reference (e.g. V1) Frequency (AC, Noise and TF only)
Parameter	Parameter name Frequency (AC, Noise and TF only)
Model Parameter	Model name Model parameter name Frequency (AC, Noise and TF only)
Temperature	Frequency (AC, Noise and TF only)
Frequency (not available for DC)	None

Mode	Parameters
Monte Carlo	Number of points Frequency (AC, Noise and TF only)
Sensitivity/Worst-case	Sensitivity measurement function Frequency (AC, Noise and TF only)

## 11.5 DC Sweep

Operates in any of the sweep modes described in “[Sweep Modes](#)” on page 230, except Frequency. Repeats a DC operating point calculation for the range of circuit parameters defined by the sweep mode.

### 11.5.1 Setting up a DC sweep

1. Select menu **Simulator | Choose Analysis...**
2. Select DC sweep check box on the right.
3. Select DC tab at the top. Enter parameters as described in the following sections.

### Sweep Parameters

Start value, Stop value	Defines sweep range stop and start values.
Points per decade, Number of points	Defines sweep range. The number of points of the sweep is defined per decade for a decade sweep. For a linear sweep you must enter the total number of points.
Device/Parameter/Model Name	The device name for a device sweep, parameter name for a parameter sweep or the model name for a model parameter sweep may be entered here. It may also be entered in the sweep mode dialog opened by pressing Define....
Define...	Sets up desired sweep mode. See “ <a href="#">Setting up a Swept Analysis</a> ” on page 234.

### Monte Carlo and Multi-step Analysis

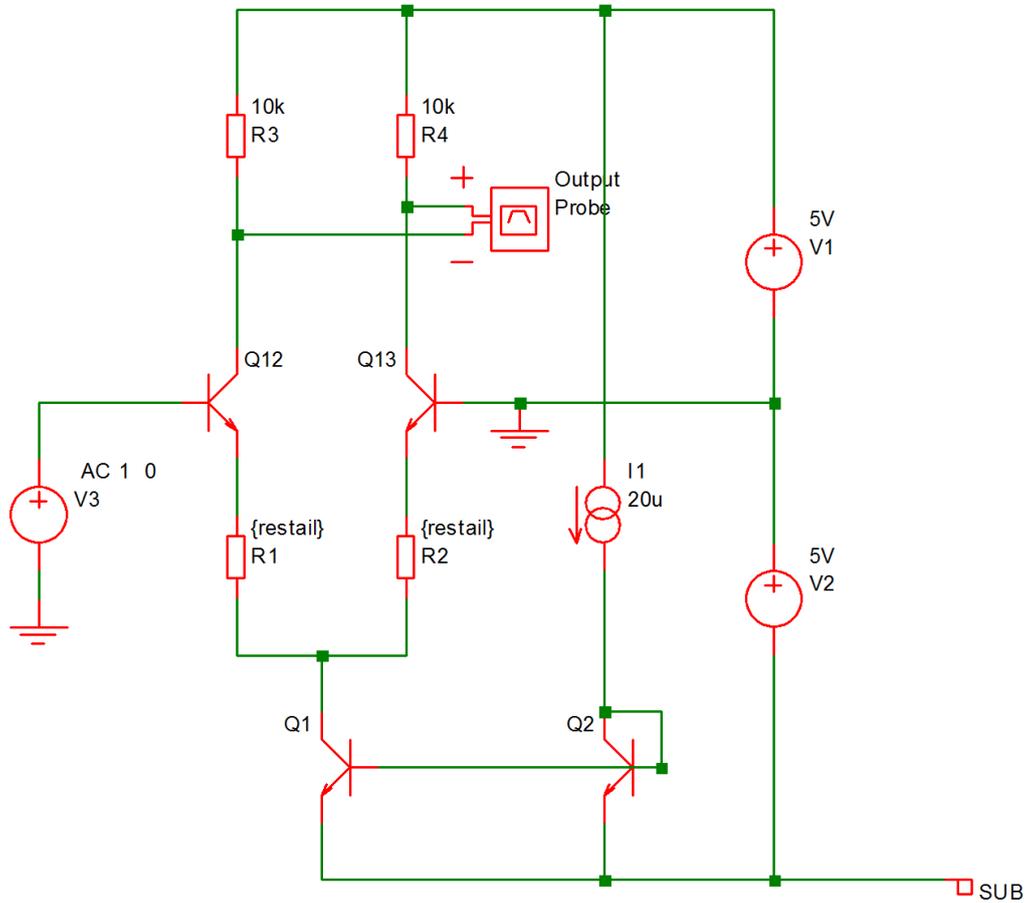
See “[Multi-step Analyses](#)” on page 259.

### See Also

*Simulator Reference Manual/Command Reference/.DC.*

## 11.5.2 Example

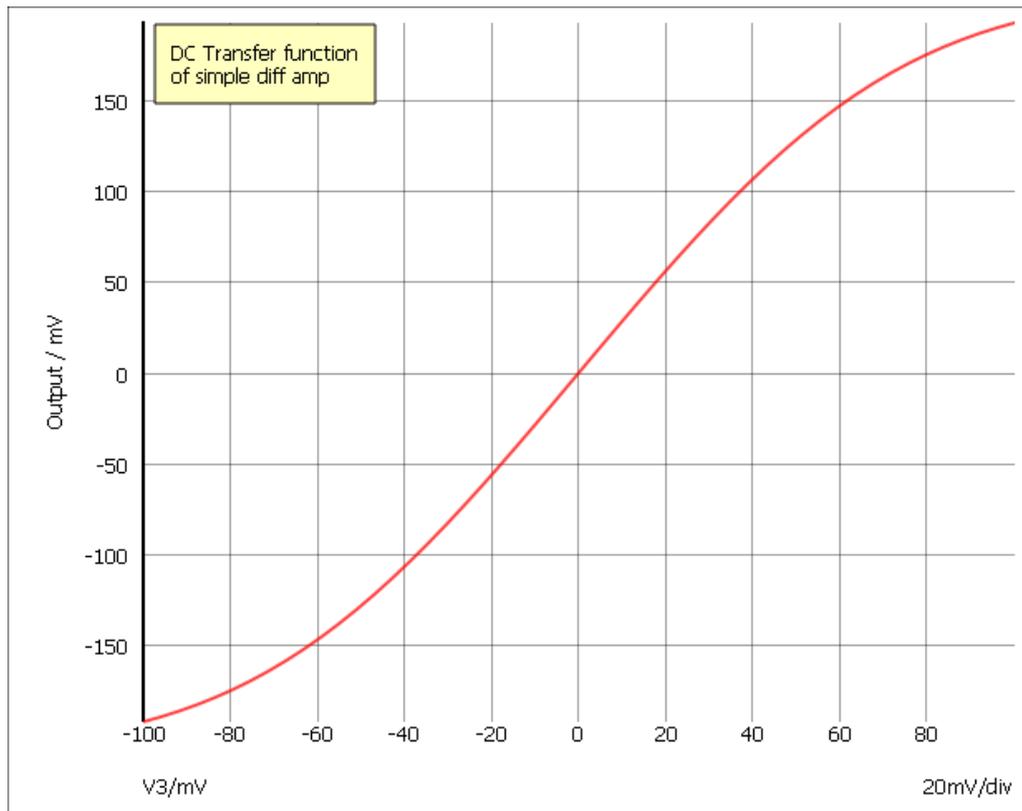
Using the example circuit (used previously):



The following is the result of a DC sweep of V3 with restail set to 1K. Analysis parameters were as follows:

Sweep mode: Device, V3

Sweep range: -0.1 to 0.1, linear sweep with 50 points.



## 11.6 AC Sweep

An AC analysis calculates the small signal response of a circuit to any number of user defined inputs. The small signal response is computed by treating the circuit as linear about its DC operating point.

Like DC, AC Noise and Transfer Function analyses, AC analysis is a swept mode and can operate in any of the 6 modes documented in [“Sweep Modes” on page 230](#). With some of these modes - e.g. sweeping a resistor value - it will be necessary for the DC operating point to be recalculated at each point while with others - such as frequency sweep - it is only necessary to calculate it at the start of the run.

For AC analysis to be meaningful there must be at least one voltage or current source on the circuit with an AC specification. To find out how to set one up see [“AC Source” on page 131](#).

### 11.6.1 Setting up an AC sweep

1. Select menu **Simulator | Choose Analysis...**
2. Select AC check box on the right.
3. Select AC tab at the top. Enter parameters as described in the following sections.

### Sweep Parameters

#### Start value, Stop value

Defines sweep range stop and start values

**Points per decade, Number of points**

Defines sweep range. The number of points of the sweep is defined per decade for a decade sweep. For a linear sweep you must enter the total number of points.

**Define...** Sets up desired sweep mode. See [“Setting up a Swept Analysis” on page 234](#).

**Monte Carlo and Multi-step Analysis**

See [“Multi-step Analyses” on page 259](#).

**Data output**

Check the Save all currents check box to enable the output of all current data including semiconductor devices. If this box is not checked the current into devices such as transistors and diodes will not be saved. In AC analysis the CPU time required to output data can be very significant relative to the solution time, so you should be aware that checking this box may slow down the simulation significantly.

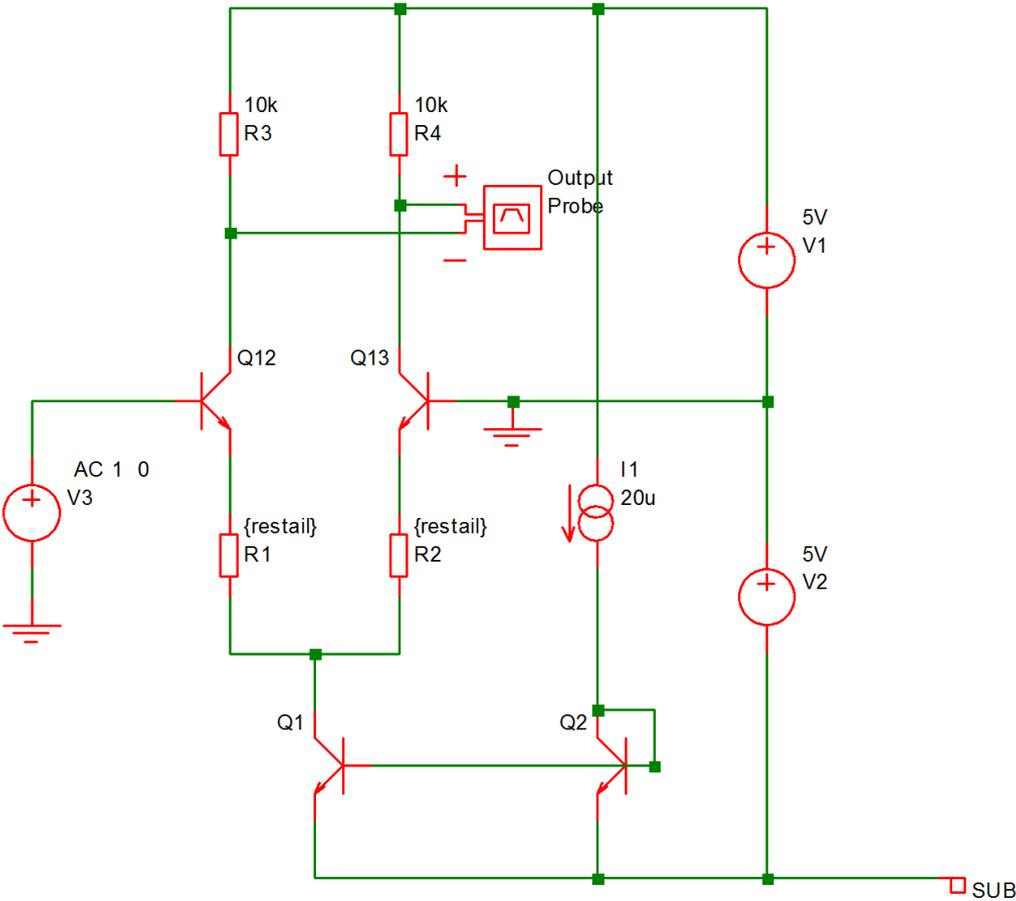
Note that this check box only affects AC analyses.

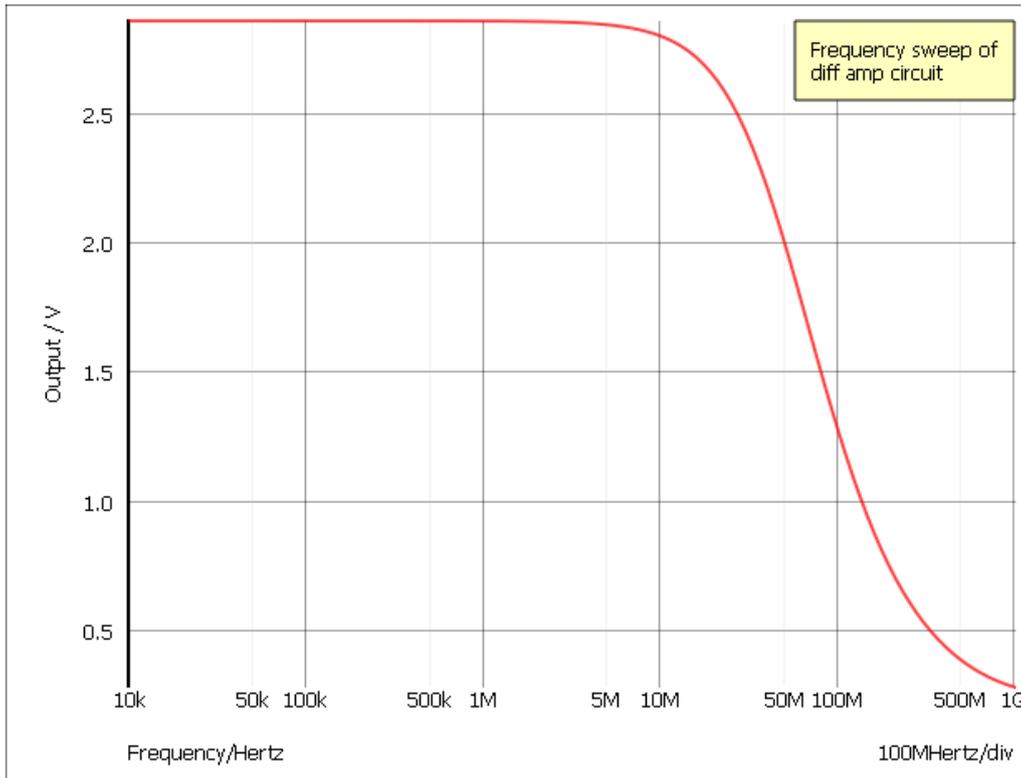
**See Also**

*Simulator Reference Manual/Command Reference/AC.*

**11.6.2 Example**

Both the examples shown in [“Sweep Modes” on page 230](#) are AC analyses. The following is a frequency sweep which is the traditional AC analysis mode. This was performed on the circuit shown below with  $\text{reftail} = 1\text{k}$ .





## 11.7 Noise Analysis

Like AC analysis, AC Noise analysis is a small signal mode. The circuit is treated as linear about its DC operating point and the contribution of all noisy devices to a designated output is computed. The total noise at that output is also calculated and optionally the noise referred back to an input source may also be computed.

Like DC, AC and Transfer Function, it is a swept mode and can be operated in any of the 6 modes described in “[Sweep Modes](#)” on page 230. With some of these modes - e.g. sweeping a resistor value - it will be necessary for the DC operating point to be recalculated at each point while with others - such as frequency sweep - it is only necessary to calculate it at the start of the run.

Note that it is not necessary to apply an AC specification to any source - including the optional input referred source - as it is with standard SPICE and many (if not all) of its derivatives.

### 11.7.1 Setting up an AC Noise Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select **Noise** check box on the right.
3. Select **Noise** tab at the top. Enter parameters as described in the following sections.

#### Sweep Parameters

Start value, Stop value

Defines sweep range stop and start values

Points per decade, Number of points	Defines sweep range. The number of points of the sweep is defined per decade for a decade sweep. For a linear sweep you must enter the total number of points.
Define...	Sets up desired sweep mode. See <a href="#">“Setting up a Swept Analysis” on page 234.</a>

## Noise Parameters

Output node	This is compulsory. It is the name of the circuit node as it appears in the netlist. Usually the schematic’s netlist generator chooses the node names but we recommend that when running a noise analysis that you assign a user defined name to your designated output node. To find out how to do this see <a href="#">“Finding and Specifying Net Names” on page 64.</a>
Reference node	Optional. Output noise is referred to this node. This is assumed to be ground if it is omitted.
Source name	Optional. Voltage or current source to which input source is referred. Enter the part reference of either a voltage or current source.

## Monte Carlo and Multi-step Analysis

See [“Multi-step Analyses” on page 259.](#)

### See Also

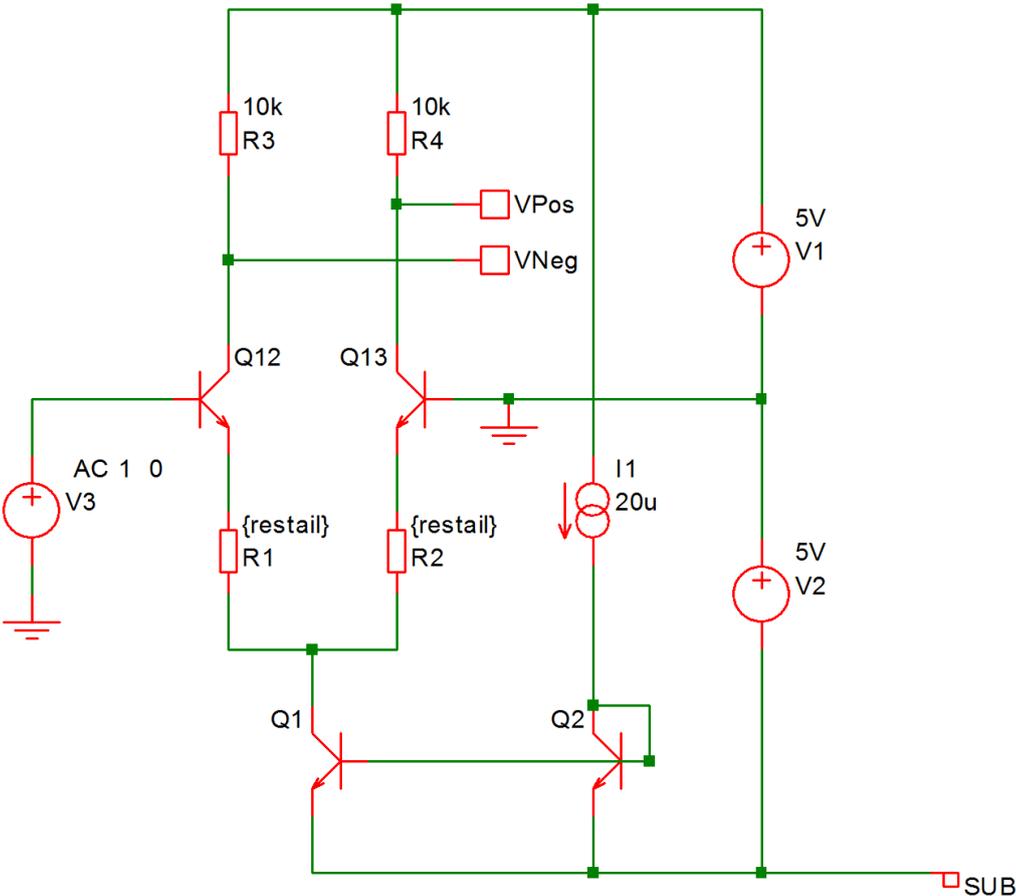
*Simulator Reference Manual/Command Reference/.NOISE.*

### 11.7.2 Plotting Results of Noise Analysis

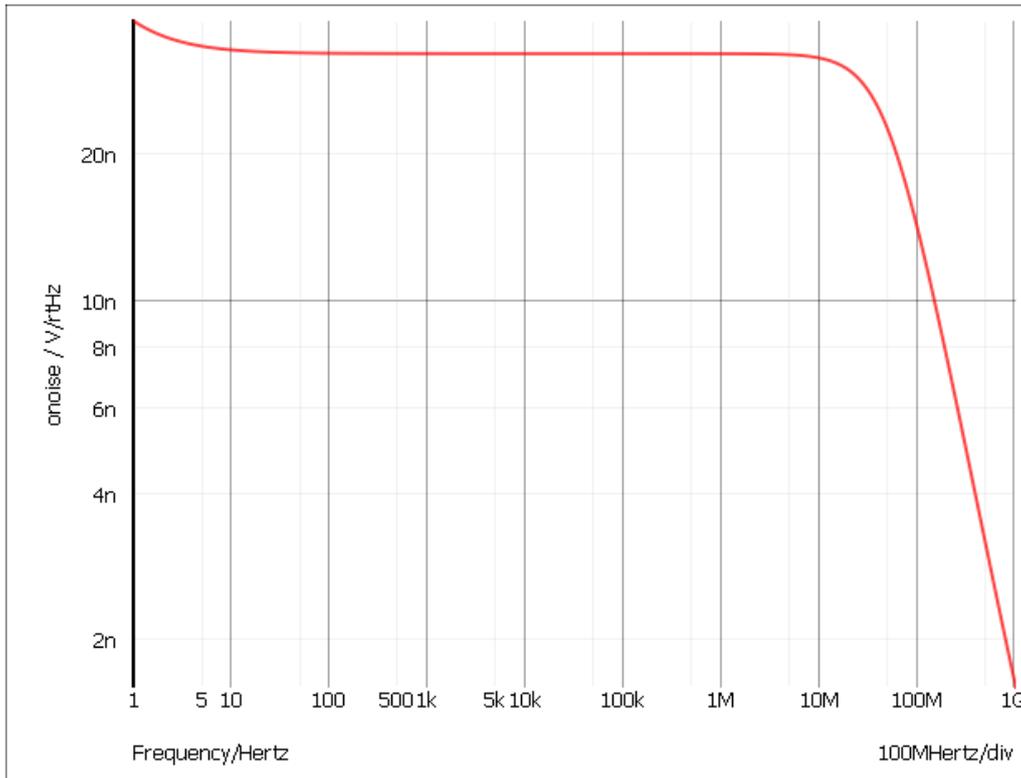
Refer to [“Plotting Noise Analysis Results” on page 324.](#)

### 11.7.3 Example 1

### Frequency Sweep



The result of a noise analysis on the above circuit using a frequency sweep



### 11.7.4 Example 2

#### Noise with a Parameter Sweep

In the following circuit we wish to find the optimum value of tail current for a source impedance of 1K. To do this we sweep the parameter *taili* which is used to set the current as well as the values for R1, R2, R3 and R4. As can be seen from the graph about  $300\mu A$  would seem to be best. The noise analysis was setup with the following parameters:

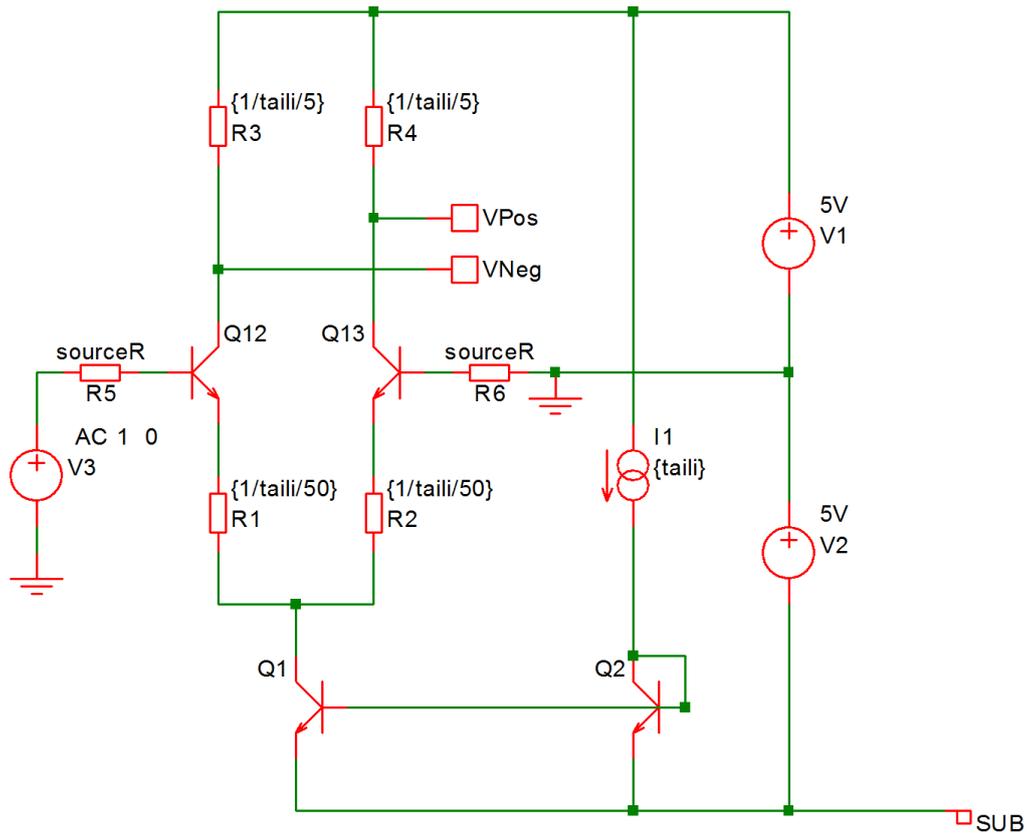
Sweep parameter *taili* from  $1\mu$  to  $10m$ , 25 points per decade

Output node: VPos

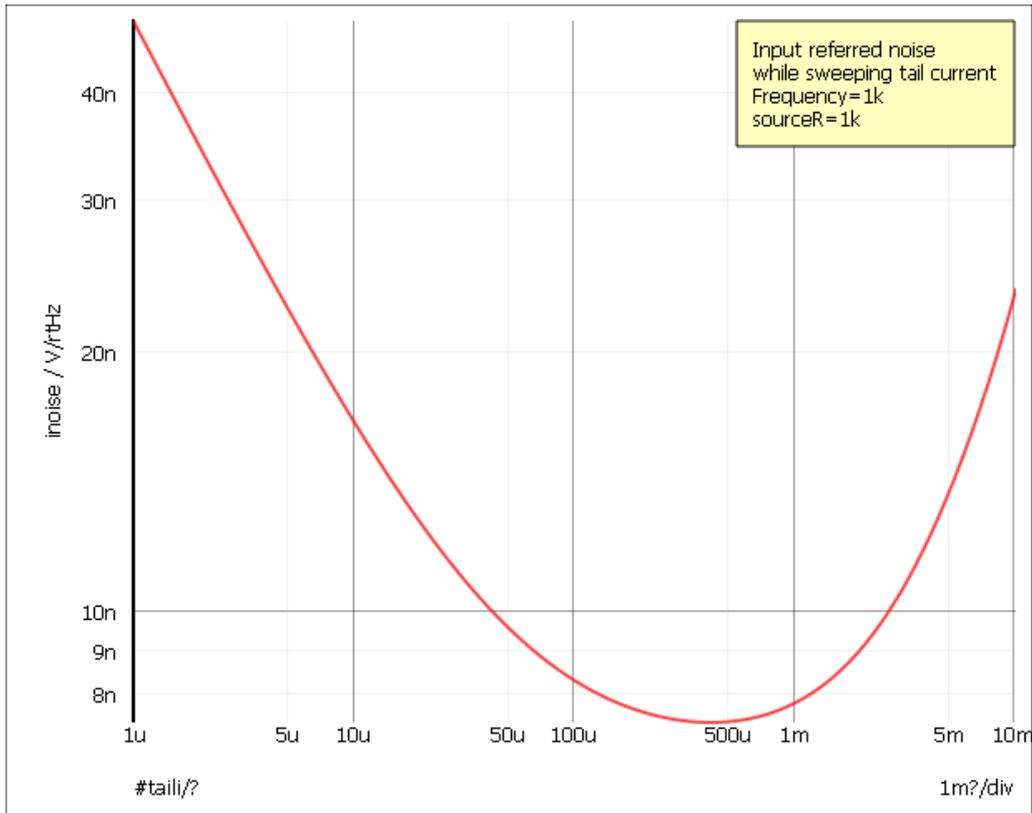
Reference node: VNeg

Input source: V3

sourceR = 1000 (set with .PARAM control)



The result:



## 11.8 Real Time Noise

This is an extension of transient analysis rather than a separate analysis mode. When activated, real time noise sources are added to all noisy devices with a magnitude and frequency distribution calculated using the same equations used for AC noise analysis. This allows noise analysis to be performed on sampled data systems and oscillators for which AC noise analysis is not appropriate.

Real time noise is not available with all versions of the product. Contact sales for details.

### 11.8.1 Setting Up a Real Time Noise Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select Transient check box on the right.
3. Select Transient tab at the top. Enter parameters as described in the following sections.
4. Set up transient analysis parameters as detailed in [“Transient Analysis” on page 225](#).
5. Check Enable real-time noise. Select Define... to set up real time noise parameters. Enter values as explained below.

#### Interval

This specifies the sampling interval of the noise generators. You should set this to a maximum of about 1/3 of the highest noise frequency of interest. Note that the interval also forces a maximum time step so short intervals can result in long simulation times.

## Start time

The time at which the noise generators are switched on. Defaults to 0.

## Stop time

The time at which the noise generators are switched off. This defaults to the stop time of the transient run.

If you think you may wish to restart the transient run after it has completed and you wish the noise generators to continue to be enabled after the restart then you must specify this time beyond the initial stop time before starting the analysis. You should avoid, however, using inappropriately large values for this stop time as this may noticeably slow the simulation and in extreme cases could cause an out of memory condition.

## RTN Mode

This affects how the noise sources are handled between noise steps. The choice is between Mode 0 and Mode 1. Mode 0 is nearly always the best mode but this can underestimate the noise in some cases. The difference between these modes is explained as follows:

Real time noise, introduces current sources across all noisy junctions. The magnitude of these sources is determined at each noise step according to the operating point of the device and a randomly generated value whose magnitude is determined from the noise equations.

The RTN mode affects how this current source is set between noise steps. This is not a problem if the operating point of the device is unchanged; the source simply ramps linearly to the next noise step. The problem occurs when the operating point changes, especially if it changes profoundly as would be the case if a transistor switches rapidly from an on-state to an off-state. In this scenario, the magnitude of the noise current would be high in the on-state but fall away to near zero in the off-state. At the same time the switch moves from a strongly conducting state to a non-conducting state.

In Mode 1, the source ramps linearly between noise steps and the operating point of the device is not considered until a new noise step is reached. This method can in some cases grossly over-estimate the noise. In Mode 0, the noise source is recalculated at each time step and adjusted according to the operating point of the device. This method tends to underestimate the noise but not by the same excessive amount that Mode 1 overestimates.

In general, we can't think of a good reason to use Mode 1 except as a confidence check. Both methods should give similar results if the noise step is small enough so a useful check is to run a circuit for a small time using each mode but with a very small noise step. The results for each should be similar.

## See Also

Real Time Noise analysis in the *Simulator Reference Manual*. This includes the results of some comparisons between AC noise and real time noise.

## 11.9 Transfer Function

Transfer function analysis is similar to AC analysis in that it performs a swept small signal analysis. However, whereas AC analysis calculates the response to all circuit nodes from a (usually) single input source, transfer function analysis calculates the individual responses from each source in the circuit to a single specified output node. This allows, for example, the series mode gain, common mode gain and power supply rejection of an amplifier to be measured in one analysis. The same measurements could be

performed using AC analysis but several of them would need to be run. Transfer function mode also calculates output impedance or admittance and, if an input source is specified, input impedance.

### 11.9.1 Setting up a Transfer Function Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select TF check box on the right.
3. Select TF tab at the top. Enter parameters as described in the following sections.

#### Sweep Parameters

Start value, Stop value	Defines sweep range stop and start values.
Points per decade, Number of points	Defines sweep range. The number of points of the sweep is defined per decade for a decade sweep. For a linear sweep you must enter the total number of points.
Define...	Sets up desired sweep mode. See <a href="#">“Setting up a Swept Analysis” on page 234.</a>

#### Transfer Function Parameters

Voltage/Current	Specify whether the output is a node voltage or device current.
Output node/Output source	This is compulsory. If voltage mode is selected it is the name of the circuit node to which the gain of all circuit sources will be calculated. It is the node name as it appears in the netlist. Usually the schematic’s netlist generator chooses the node names but we recommend that when running a transfer function analysis that you assign a user defined name to your designated output node. To find out how to do this see <a href="#">“Finding and Specifying Net Names” on page 64.</a>  If current mode is selected it is the name of a voltage source through which the output current is measured. The simulation will calculate the gain for every circuit source to this current.
Reference node	Optional and only available in voltage mode. Output voltage is referred to this node. This is assumed to be ground if it is omitted.
Source name	Optional. Input impedance to this source will be calculated if specified.

#### Monte Carlo and Multi-step Analysis

See [“Multi-step Analyses” on page 259.](#)

## See Also

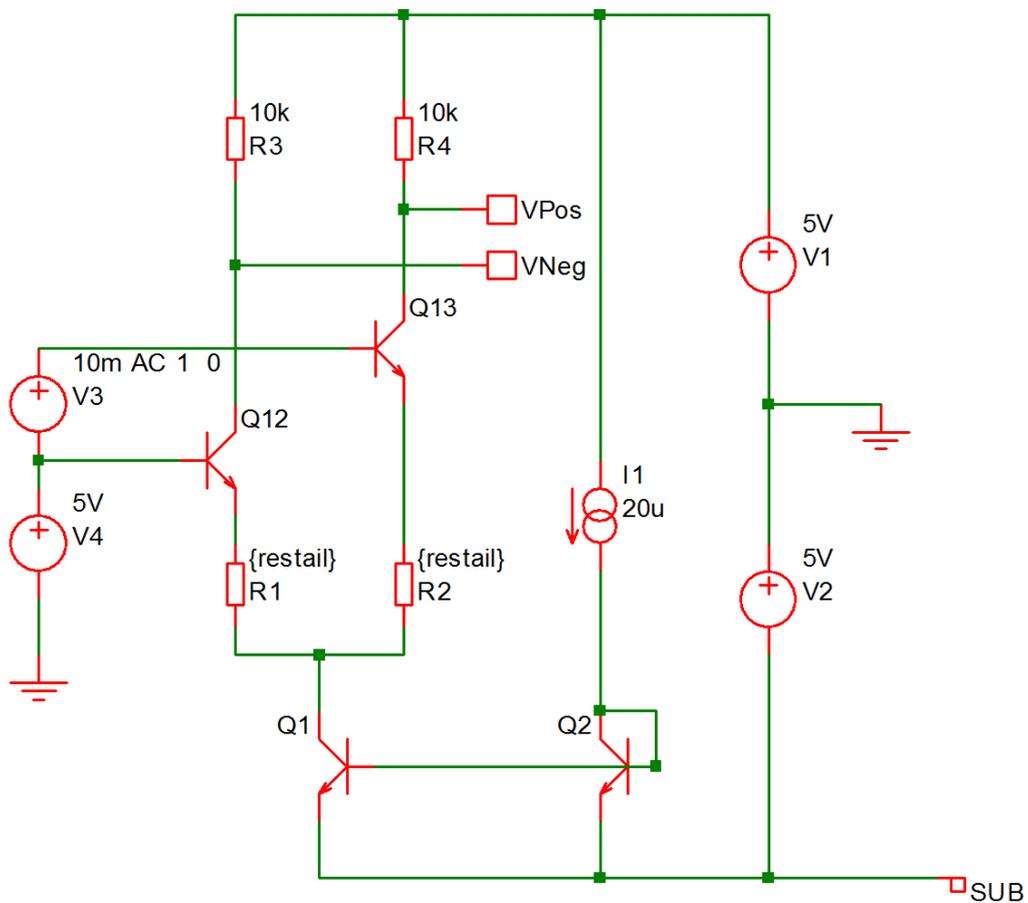
*Simulator Reference Manual/Command Reference/.TF.*

### 11.9.2 Plotting Transfer Function Analysis Results

See “Plotting Transfer Function Analysis Results” on page 325.

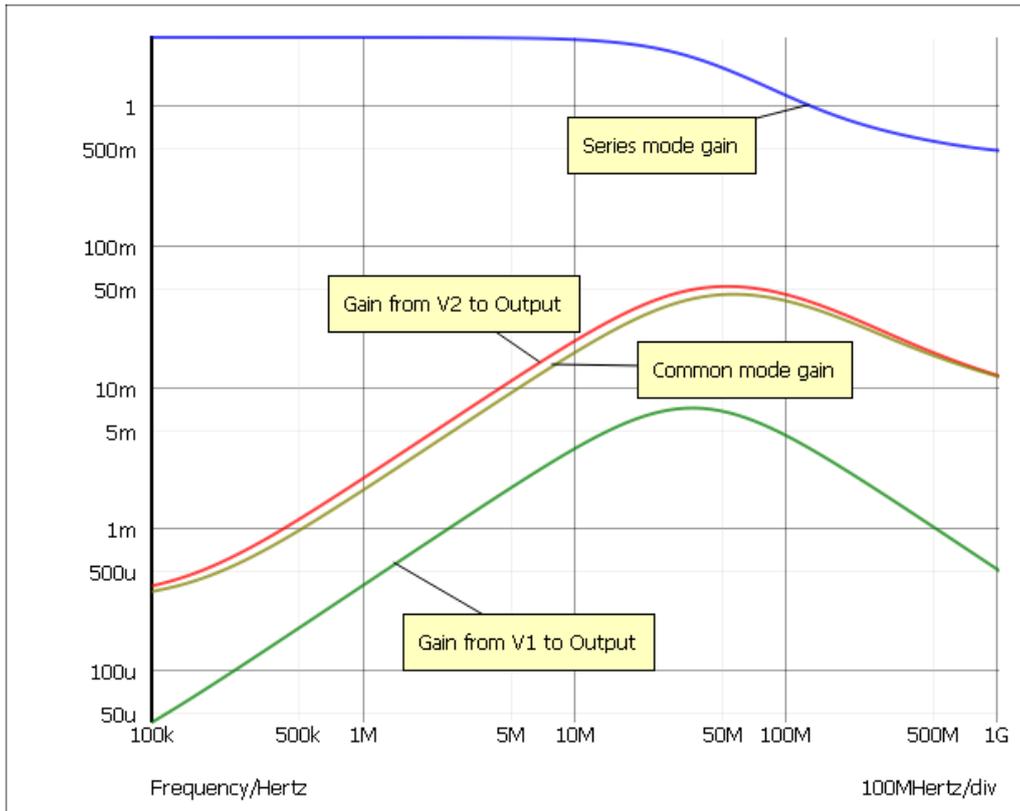
### 11.9.3 Example

Perform transfer function frequency sweep on the following circuit



Transfer function frequency sweep

The results:



All of the above waveforms were created with a single analysis.

## 11.10 DC Sensitivity

This is a stand-alone analysis mode and should not be confused with the sensitivity/worst-case sweep mode and multi-step mode.

This control instructs the simulator to perform a DC sensitivity analysis. In this analysis mode, a DC operating point is first calculated then the linearised sensitivity of the specified circuit voltage or current to every model and device parameter is evaluated. The results are written to the list file and they are also placed in a new data group. The latter allows the data to be viewed in the message window (type Display) at the command line and can also be accessed from scripts for further analysis.

This analysis mode performs a similar analysis to a sensitivity sweep applied to a DC analysis. However it differs in that it calculates the sensitivity of every parameter in the circuit, not just those with a distribution function applied. Also the way the analysis is performed is different. The sensitivity sweep perturbs each parameter and performs a full non-linear DC analysis for each case. This analysis mode uses some matrix arithmetic to calculate an approximation to the true sensitivity.

### 11.10.1 Setting up a DC Sensitivity Analysis

Place a control of the following form in the **F11** window:

```
.SENS V(nodename [,refnodename]) | I(sourcename)
```

<i>nodename</i>	Output node to which sensitivities are calculated
<i>refnodename</i>	Reference node. Ground if omitted
<i>sourcename</i>	Voltage source to measure output current to which sensitivities are calculated.

## 11.11 Convergence

Simulations sometimes fail to converge - that is the simulation aborts before completion with one of these messages:

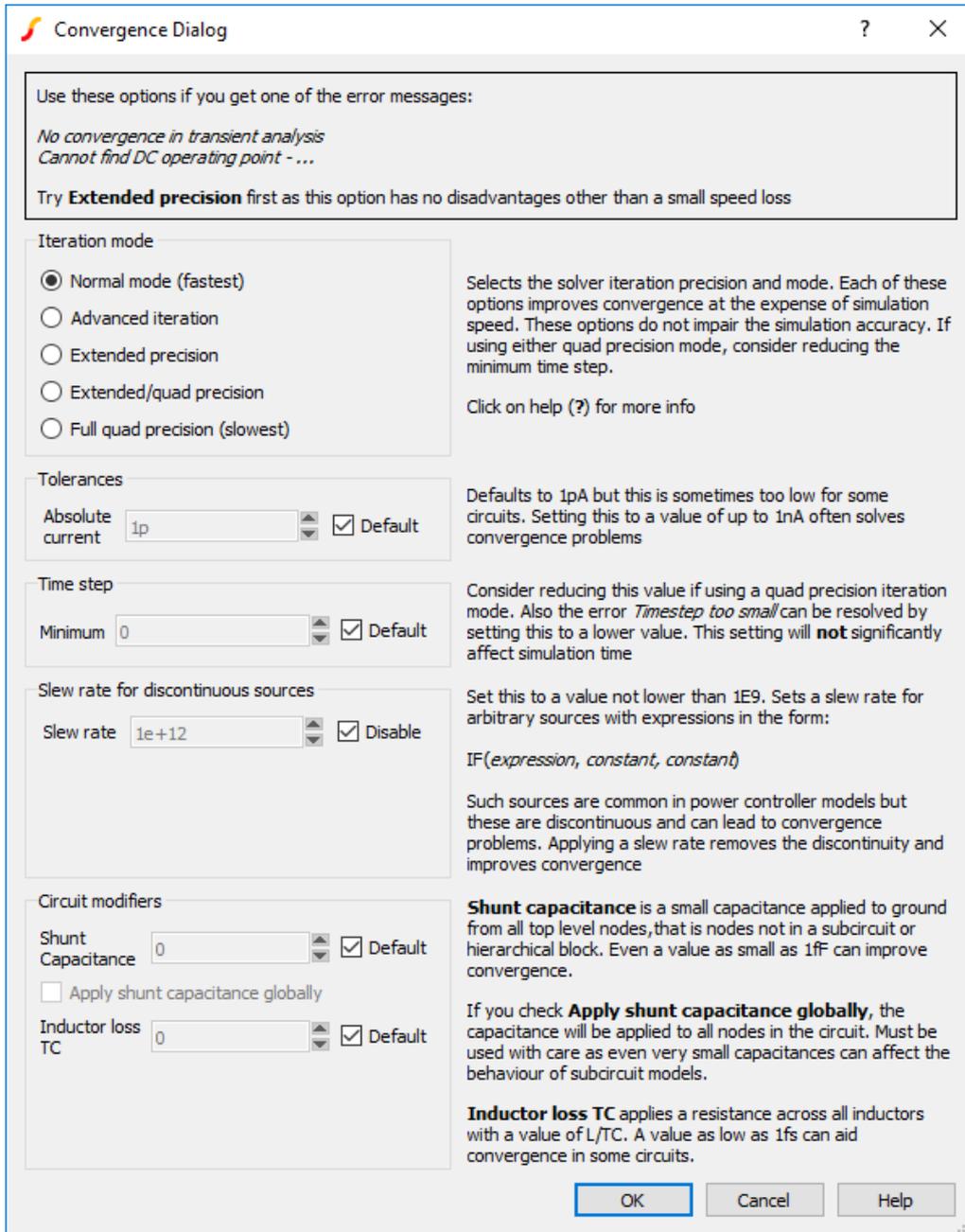
- No convergence in transient analysis
- No convergence in DC analysis
- Cannot find DC operating point - Iteration limit exceeded in pseudo transient analysis
- Cannot find DC operating point - No convergence in pseudo transient analysis

Convergence failures can be fixed by setting appropriate simulation options or by making changes to the circuit or by making changes to the models. Sometimes a combination of all three is required. See the following sections for more information:

- [“Convergence Options” on page 251](#)
- [“Convergence Failure Report” on page 254](#)
- [“Convergence Static Analysis” on page 254](#)

### 11.11.1 Convergence Options

Select menu **Simulator | Convergence Options**. You will be presented with this dialog box:



This dialog allows to set a number of simulation options that can help simulation convergence. These options sometimes have a penalty, either they slow down the simulation or in some way degrade accuracy. These are described below

Item	Description
Iteration mode	Selects solver iteration and mode. Each of these options improves convergence at the expense of simulation speed. <b>Extended precision</b> mode provides a worthwhile convergence improvement with only a modest speed loss.

Item	Description
Tolerances	Setting current tolerance to 1n sometimes fixes convergence problems. You should not increase this setting if your circuit has very small currents
Time step	The Minimum Time step has a default value of 1e-18s. Lower values sometimes help transient analysis especially if using quad precision iteration modes. Increasing this value does <b>not</b> improve simulation speed.
Slew rate for discontinuous sources	This will only have an effect if you use models that contain discontinuous sources. These are common in switching power controllers
Shunt capacitance	This is often effective especially if the <b>Apply shunt capacitance globally</b> box is checked. However, this option is effectively modifying your circuit so should always be used with care. Note that this option will only benefit transient analysis

## Iteration Mode

The iteration mode trades off simulation speed with *numerical noise*. *Numerical noise* is one of the major causes of simulation failure in transient analysis. *Numerical noise* is the noise induced as a result of uncertainty in a calculation due to the limited precision of the arithmetic and is sometimes referred to as *round-off error*. It is similar in concept to quantization noise that is prevalent in data converters. Numerical noise increases as the time step reduces due to the behaviour of reactive components and can reach levels that exceed the tolerance parameters that determine when convergence has been reached.

SIMetrix provides a range of iteration modes that reduce numerical noise by employing a higher precision in its numerical calculations. In addition SIMetrix offers an advanced iteration mode which lowers numerical noise by employing a more efficient iterative equation.

Mode	Typical numerical noise factor	Typical simulation run time factor	Advanced iteration	Precision
Normal	1 x	1x	No	Double/Extended
Advanced iteration	÷ 100	+ 5%	Yes	Double/Extended
Extended precision	÷ 10000	+ 10%-30%	Yes	Extended
Extended/quad precision	÷ 1e18	2x to 8x	Yes	Extended/Quad
Quad precision	÷ 1e18	4x to 10x	Yes	Quad

Double precision has a decimal resolution of approximately 16 digits. Extended precision has a decimal resolution of approximately 19 digits. Quad precision has a decimal resolution of approximately 34 digits

### 11.11.2 Convergence Failure Report

The convergence failure report provides details of items that caused the convergence failure. To create a convergence report, select menu **Simulator | Convergence Report**. The report is divided into the following sections:

<b>Nodes which failed to converge on the final step</b>	Lists circuit nodes whose failure to converge led to the simulation aborting. These should usually be the initial focus for investigation. These nodes will be highlighted on the schematic if at the top level of the hierarchy and this often shows the general area that is giving trouble
<b>Devices which failed to converge on the final step</b>	Devices whose failure to converge led to the simulation aborting
<b>Items which failed to converge earlier in the run</b>	The items listed here did not necessarily lead to the failure. However, devices or nodes that often fail during a run may be worth investigating. The <b>Count</b> value indicates how often the item failed.
<b>Nodes which overflowed during the run</b>	If a node voltage exceeds a preset limit, the iteration is aborted and then repeated with a smaller step. This is not necessarily a cause for concern but if the same node frequently overflows, this could be indicative of some problem
<b>Static audit of circuit</b>	Result of a static analysis of the circuit. See below for further details

### 11.11.3 Convergence Static Analysis

The convergence static analysis feature performs an analysis of the circuit to identify items that can give trouble and could lead to convergence failure. The static analysis is automatically performed as part of the convergence failure report but may also be run on its own before or after a simulation using the menu **Simulator | Convergence Static Audit**. If run before a simulation, a dummy run of the simulation will be required and you will be prompted whether you would like this to be performed. This will load the circuit into the simulator but not actually start the run.

The static analysis will identify the following potential problems:

Code	Description
IfDiscontinuity	Arbitrary source expression in the form <code>IF(expression, constant, constant)</code> . These are discontinuous as they switch abruptly from one value to another with no time definition. This can lead to convergence failure
SgnDiscontinuity	Use of Sgn (sign) function in an arbitrary source. These can be discontinuous although it depends on how they are used. If this is reported in an Infineon MOSFET or IGBT model then it can be safely ignored
StpDiscontinuity	Use of U() or Stp() functions in an arbitrary source. Can be discontinuous but depends on how they are used
FloorDiscontinuity	Use of floor function in an arbitrary source. This is discontinuous and often gives convergence problems

Code	Description
DivideByZero	A divide by zero condition was detected in an arbitrary source
DivideByVar	This means something of the form $\text{expression}/V(n1)$ or $\text{expression}/(V(n1,n2))$ or $\text{expression}/(V(n1)-V(n2))$ or $\text{expression}/I(v)$ . In each case the $V()$ or $I()$ access function can - and usually does - return zero at some point leading to a divide-by-zero error
ValueTooSmall	A model or device parameter was set to a value that is too small. An example is the N parameter for a diode model. Small values are often used in models in order to make a diode with a small voltage drop but this can lead to difficulty in some situations.
ValueTooLarge	A model or device parameter was set to a value that is too small. Large capacitors trigger this failure for example

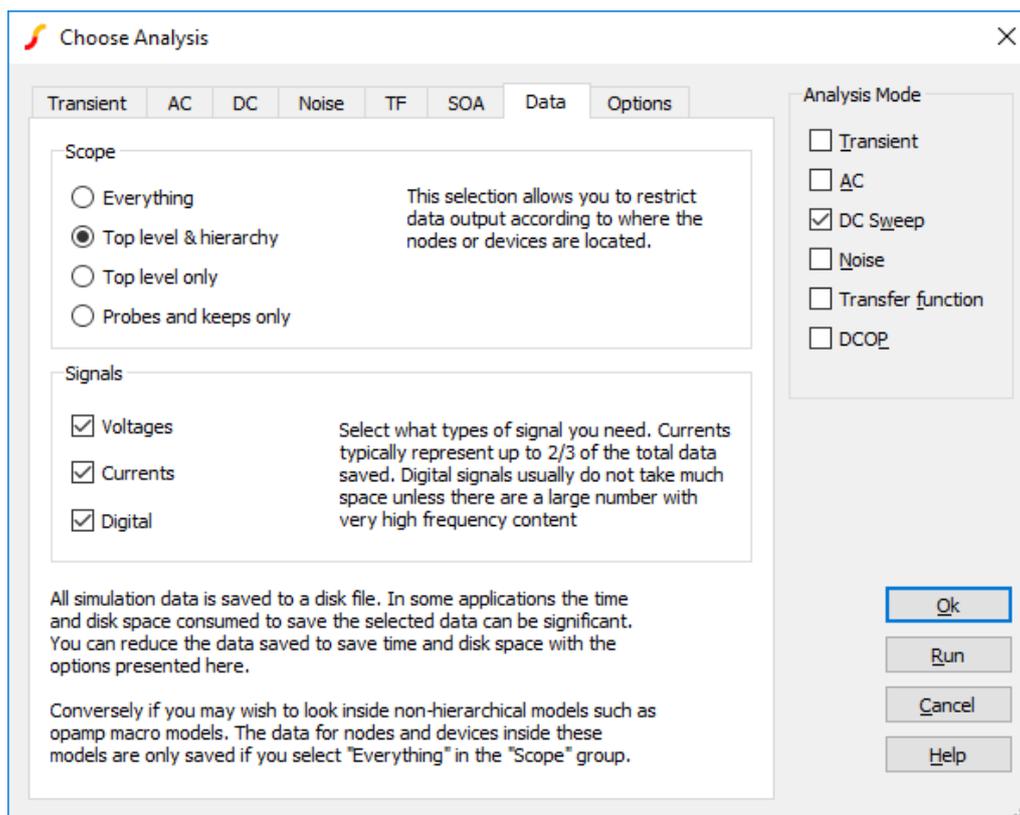
## 11.12 Data Handling and Keeps

Keeps form part of a system to limit the amount of data that is output during a simulation. For some designs the data output can be too great to fit in the available disk space. In the case of multi-core multi-step runs, it is possible for the rate at which data is generated to exceed the write speed of the disk system leading to slow run times. In these situations, the data output needs to be restricted.

By default, all voltage and currents shown in schematics are saved. This includes data from child schematics in hierarchical designs but excludes signals inside ASCII subcircuit models.

To restrict data output you can use the Data tab within the choose analysis dialog box:

1. Select menu **Simulator | Choose Analysis...**
2. Click on the Data tab. This will show the following:



### 11.12.1 Scope Options

In the following description, ASCII Subcircuits means subcircuits that are not created by the SIMetrix hierarchical schematic editor. The SIMetrix hierarchical schematic editor uses SPICE subcircuits (using .SUBCKT) as its base but decorates the subcircuit definitions with a special comment to distinguish them from subcircuits from other sources. ASCII subcircuits would typically include models for devices such as opamps along with foundry models for integrated circuit processes.

Everything	All data will be saved including signals inside ASCII subcircuits.
Top level & hierarchy	Data from the root schematic and all hierarchical child schematics will be saved. Data for fixed probes and keeps (see below) will also be saved. Data inside ASCII subcircuits will not be saved.
Top level only	Only signals from the root schematic will be saved. Data for fixed probes and keeps (see below) will also be saved.
Probes and keeps only	Only signals attached to fixed probes or keeps (see below) will be saved.

### 11.12.2 Signals Options

Signals are classed as voltages, currents or digital. Digital signals are those created by the event driven simulator and in some cases by the Verilog-HDL interface. Digital signals generally take up much less

space as data is only stored when they change state whereas for analog signals data is stored on every time step.

In most applications disabling the saving of currents will reduce the amount of data saved by a very significant amount and this may be sufficient to resolve any problems caused by large amounts of data being saved. In this case current keeps may be used to save specific currents as required.

With some or all data output inhibited the options described above, you can add keep symbols to the schematic to select specific voltages or currents to be saved.

Keep symbols use the underlying simulator statement `.KEEP`. For information on the comprehensive features of `.KEEP`, please refer to the *Simulator Reference Manual/Command Reference/.KEEP*.

### 11.12.3 To Add a Voltage Keep to a Schematic

1. From the “Part Selector” on page 124 navigate to Probes and Connectors → Keeps → Voltage Keep.
2. Place device on desired schematic net.

### 11.12.4 To Add a Current Keep to a Schematic

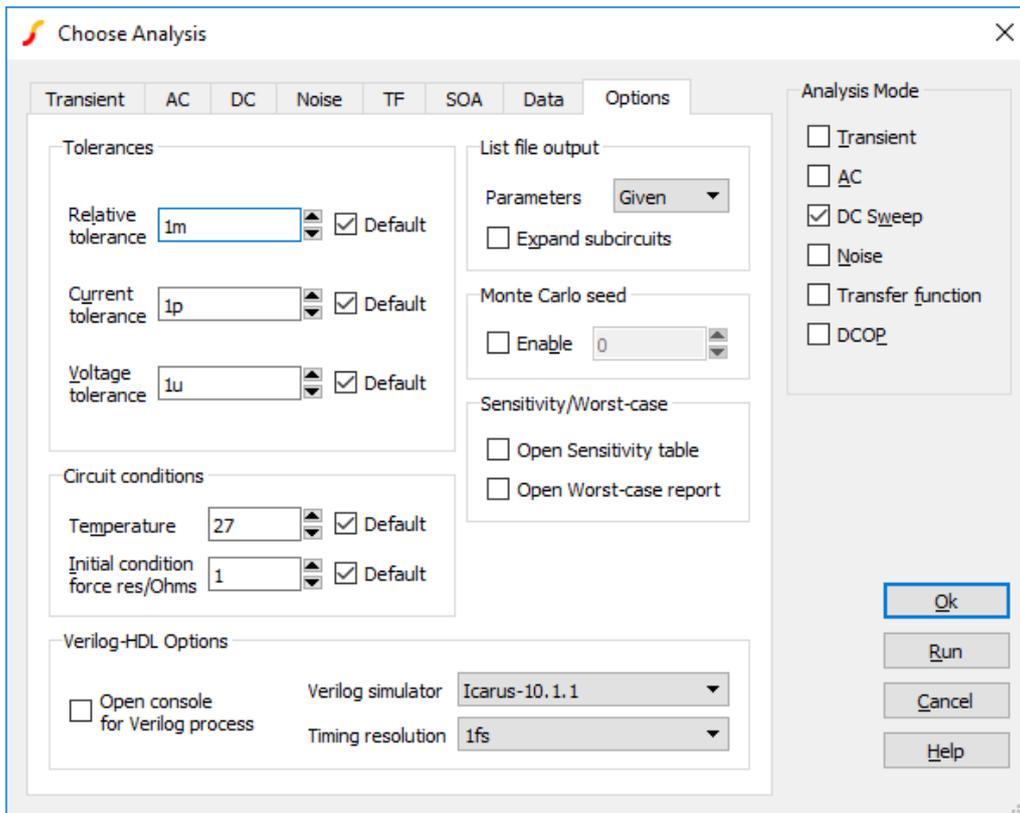
1. From the “Part Selector” on page 124 navigate to Probes and Connectors → Keeps → Current Keep.
2. Place device *directly* on a device pin.

## 11.13 Simulator Options

The simulator features a large number of option settings although, fortunately, the vast majority can be left at their default values for nearly all applications. A few option settings can be set via the Choose Analysis dialog box and these are described in the following sections. The remainder can be controlled using the simulator’s `.OPTIONS` control details of which may be found in the *Simulator Reference Manual/Command Reference/.OPTIONS*.

### 11.13.1 Setting Simulator Options

1. Select menu **Simulator | Choose Analysis...**
2. Select **Options** tab. The following will be displayed:



Note that the Verilog-HDL group will only be displayed in versions with the Verilog-HDL feature

## Tolerances

- Relative Tolerance** Controls the overall accuracy of the simulation. The default value is 0.001 and this is adequate for most applications. If you are simulating oscillator circuits it is recommended to reduce this to 0.0001 or lower.
- Increasing this value will speed up the simulation but often degrades accuracy to an unacceptable level.
- Current Tolerance** Sets the minimum tolerance for current. It may be beneficial to increase this for circuits with large currents.
- Voltage Tolerance** Sets the minimum tolerance for voltage. It may be beneficial to increase this for circuits with large voltages.

## Circuit Conditions

- Temperature** Circuit temperature in  $^{\circ}C$ .
- Initial Condition Force Resistance** Initial conditions apply a voltage to a selected node with a force resistance that defaults to  $1\Omega$ . This option allows that force resistance to be changed.

## List File Output

Expand subcircuits	If checked, the listing of expanded subcircuits will be output to the list file. This is sometime useful for diagnosing problems.
Parameters	Controls the level of model and device parameter output to the list file. Options are: <ul style="list-style-type: none"> <li>None No Output</li> <li>Brief Only values defined by an expression are output</li> <li>Given The default. Values that are explicitly defined are output</li> </ul>

## Monte Carlo Seed

Seed for pseudo random number generator used to generate random numbers for tolerances. See [“Multi-step Analyses” on page 259](#). If Enable check box is unchecked, a seed value will be chosen by the simulator.

## Verilog-HDL Options

This section will only show if Verilog-HDL simulation is available for your version of SIMetrix.

Open console for Verilog process	When the Verilog simulator runs, a console window will be displayed showing any output messages from the simulator. See <a href="#">“Open Console for Verilog Process” on page 432</a> for details.
Verilog simulator	Simulator that will be used to run Verilog-HDL. See <a href="#">“Verilog Simulator” on page 432</a> for details.
Timing resolution	Time resolution in Verilog simulator. See <a href="#">“Timing Resolution” on page 432</a> for details.

## 11.14 Multi-step Analyses

The analysis modes, Transient, AC, DC, Noise and Transfer Function can be setup to automatically repeat while varying some circuit parameter. Multi-step analyses are defined using the same 7 sweep modes used for the individual swept analyses in addition to snapshot mode. See [“Transient Snapshots” on page 228](#) for details of snapshots. The 6 modes are briefly described below. Note that Monte Carlo analysis is the subject of a whole chapter see [“Monte Carlo Analysis” on page 411](#).

- Device. Steps the principal value of a device. E.g. the resistance of a resistor, voltage of a voltage source etc. The part reference of the device must be specified.
- Model parameter. Steps the value of a single model parameter. The name of the model and the parameter name must be specified.
- Temperature. Steps global circuit temperature.
- Parameter. Steps a parameter that may be referenced in an expression.
- Frequency. Steps global frequency for AC, Noise and Transfer Function analyses.
- Monte carlo. Repeats run a specified number of times with tolerances enabled.

As well as 6 different modes there are 3 different sweep methods which can be applied to all modes except Monte Carlo. These are:

- Linear
- Decade
- List

The simulator also offers an Octal sweep method but this is not supported by the Choose Analysis Dialog.

### 11.14.1 Setting up a Multi-step Analysis

Define Transient, AC, DC, Noise or Transfer Function as required then check **Enable Multi-step** and press **Define...** button. For transient/DC analysis you will see the following dialog box. Other analysis modes will be the same except that the frequency radio button will be enabled.

**Define Multi Step Analysis**

**Sweep mode**

- Device
- Parameter
- Model parameter
- Temperature
- Frequency
- Monte Carlo
- Snapshot
- Sensitivity/Worst case

**Step parameters**

Start value: 1k  Decade

Stop value: 2k  Linear

Number of steps: 10  List

Group digital curves Define List...

Number of Cores: 1

Number of physical cores: 48

Number of cores allowed by license: 16

**Sensitivity parameters**

**Sensitivity measurement functions**

Functions...

For multi-step sensitivity analysis, the measurement functions must return a scalar value from a vector. E.g. Mean1(VOUT) returns the mean of the signal at node VOUT

At least one measurement function should be supplied. Up to two may be entered here. More can be added manually in the F11 window

Run worst case analysis

Perturbation: 100%

**Parameters**

Device name:

Parameter name:

Enable Monte Carlo log file

Ok Cancel Help

Enter parameter as described below. Only the boxes for which entries are required will be enabled. In the above example, only the Number of steps box is enabled as this is all that is required for Monte Carlo mode.

## Sweep Mode

Choice of 6 modes as described above.

## Step Parameters

Define range of values. If **Decade** is selected you must specify the number of steps per decade while if **Linear** is specified, the total number of steps must be entered. If **List** is selected, you must define a sequence of values by pressing **Define List...**

**Number of cores** On multi-core computers, the work for multi-step analyses may be split between multiple cores to speed up the simulation. See [“Using Multiple Cores for Multi-step Analyses” on page 262](#) for more information.

## Parameters

The parameters required vary according to the mode as follows:

Mode	Parameters
Device	Device name (e.g. V1)
Parameter	Parameter name
Model Parameter	Model name Model parameter name
Temperature	None
Frequency (not DC or transient)	None
Monte Carlo	None
Snapshot	See <a href="#">“Transient Snapshots” on page 228</a>

### 11.14.2 Using Multiple Cores for Multi-step Analyses

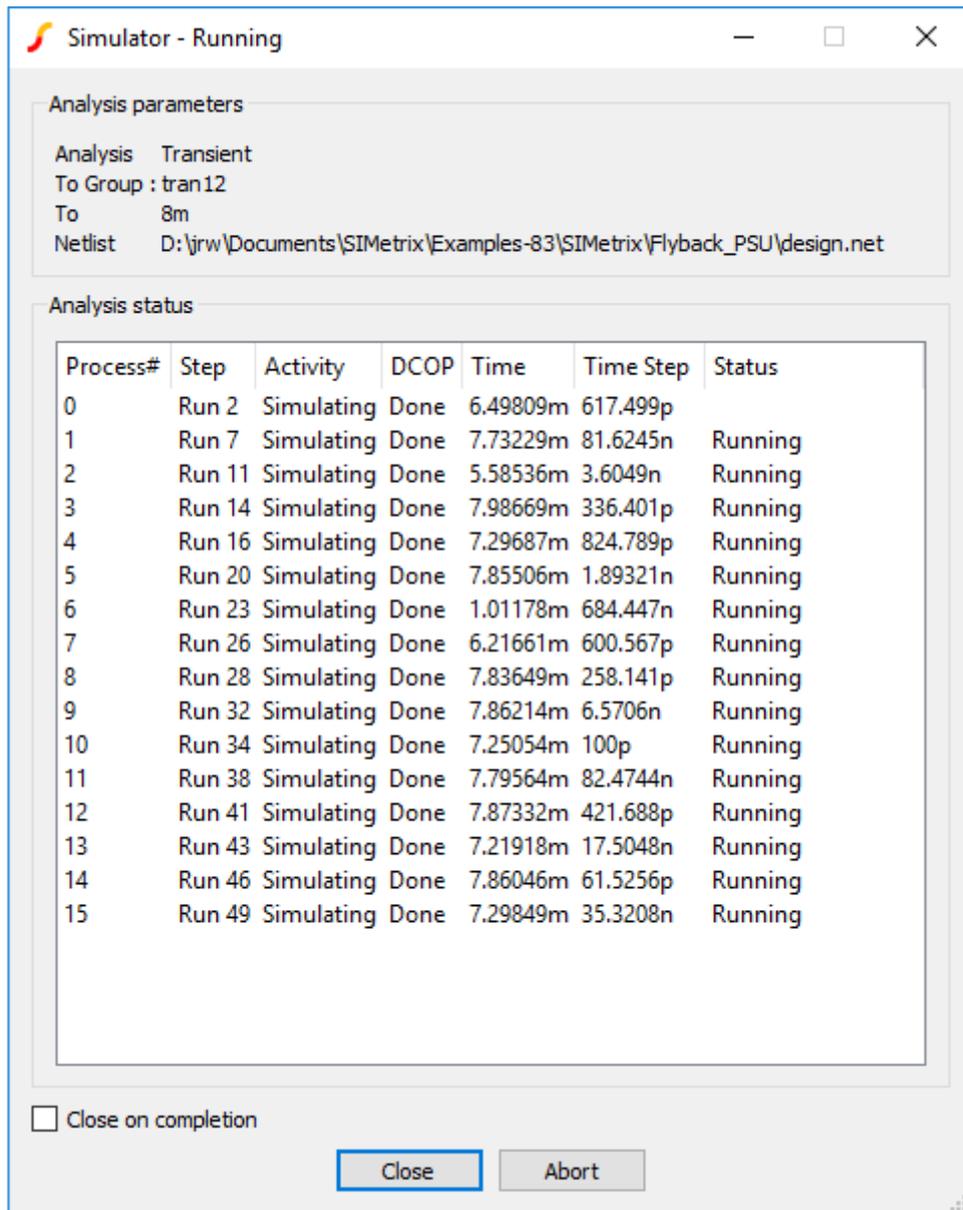
The work for multi-step analyses can be split between multiple processor cores to speed up the simulation. The maximum number cores that may be used is dependent on your license and on the number of processor cores that your system is equipped with. The speed improvement that can be achieved by this method can vary due to a number of factors but is typically of the order of 75% of the core count. E.g. if you have 4 cores, the speedup may be about 3 fold.

### Setting up a Multi-core Multi-step Simulation

Set up a multi-step simulation in the normal way but set the **Number of Cores** edit box to the number of cores you wish to use. You will not be able to set the number of cores to larger than a certain value depending on your system and your license. These values are shown below the number of cores selector.

### Running a Multi-core Multi-step Simulation

Run the simulation in the normal way. The simulator status box will show a single entry for each core being used. See below



The above shows the status box showing for a 1000 step Monte Carlo analysis using 8 cores. Each line shows the status for each of the 8 processes. Apart from sending back status information, each of the 8 processes runs completely independently and writes to its own data file. When the run is complete the data from each process will be linked to the main data file. Subsequently you can plot and analyse the results in exactly the same way as you would a single core run.

### Using Fixed Probes with Multi-core Multi-step Simulation

Fixed probes usually plot the simulation results *incrementally*, that is the plots are updated as the simulation proceeds. With Multi-core Multi-step runs, only the curves created by the primary process are plotted during the run. The remaining curves will be plotted automatically once the run is complete.

## Data Handling with Multi-core Multi-step

With multiple processes running in parallel all writing data to a single disk, it is possible in some cases for a disk write bottleneck to develop whereby the simulation appears to hang temporarily. This problem is particularly acute with large circuits where a large number of signals are being saved. Each signal being saved is allocated a buffer in main memory and that buffer is written to disk when it is full. This works well if the buffers are large enough but with large circuits and many cores the buffers will be smaller and the time taken to write them out will become dominated by disk seek times.

It is not easy to predict in a given situation whether the problem will arise. If you are running a medium to large circuit (over 2000 nodes) and are using 8 or more cores you should consider restricting the data saved. See [“Data Handling and Keeps” on page 255](#).

### 11.14.3 Example 1

Refer to circuit in [“Plotting Results of Noise Analysis, Example 2” on page 244](#). In the previous example we swept the tail current to find the optimum value to minimise noise for a 1K source resistance. Here we extend the example further so that the run is repeated for a range of source resistances. The source resistance is varied by performing a parameter step on *sourceR*. Here is what the dialog settings are for the multi-step run:

**Define Multi Step Analysis**

**Sweep mode**

- Device
- Parameter
- Model parameter
- Temperature
- Frequency
- Monte Carlo
- Snapshot
- Sensitivity/Worst case

**Step parameters**

Start value: 1k  Decade

Stop value: 100k  Linear

Steps per decade: 2  List

Group digital curves Define List...

Number of Cores: 1

Number of physical cores: 48

Number of cores allowed by license: 16

**Sensitivity parameters**

**Sensitivity measurement functions**

Functions...

For multi-step sensitivity analysis, the measurement functions must return a scalar value from a vector. E.g. Mean1(VOUT) returns the mean of the signal at node VOUT

At least one measurement function should be supplied. Up to two may be entered here. More can be added manually in the F11 window

Run worst case analysis

Perturbation: 100%

**Parameters**

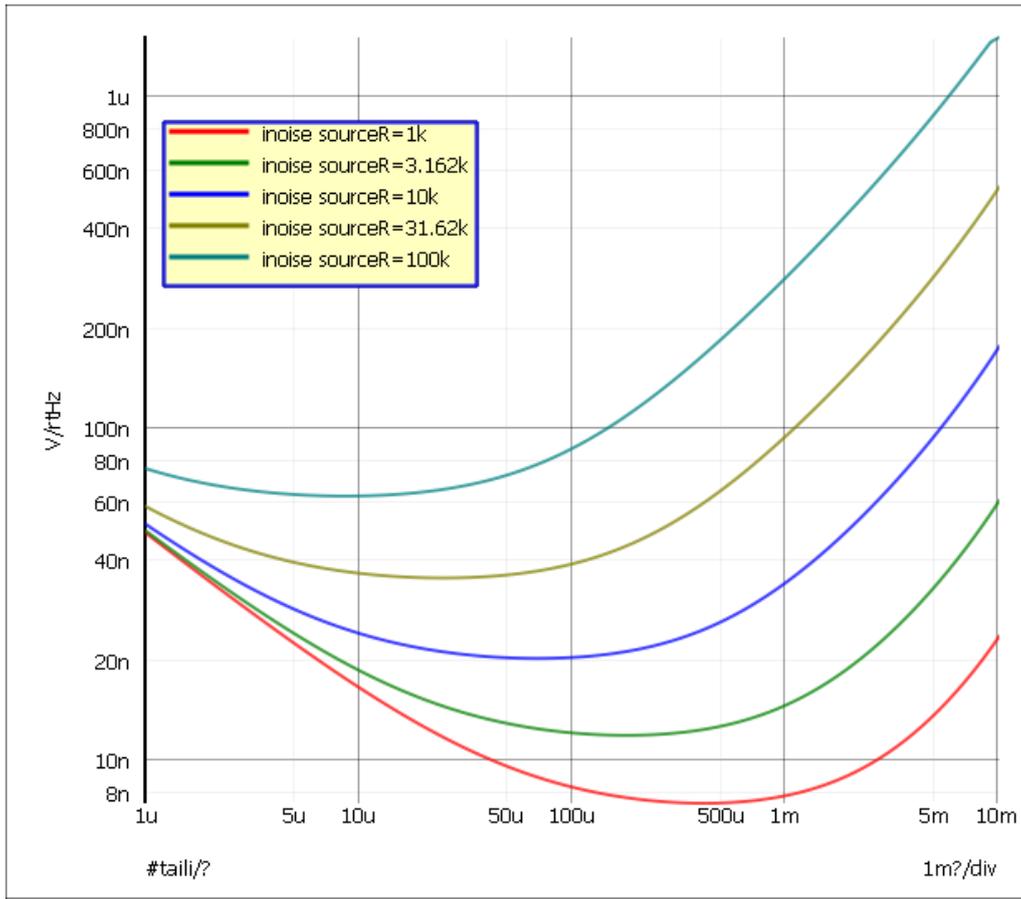
Device name:

Parameter name: sourceR

Enable Monte Carlo log file

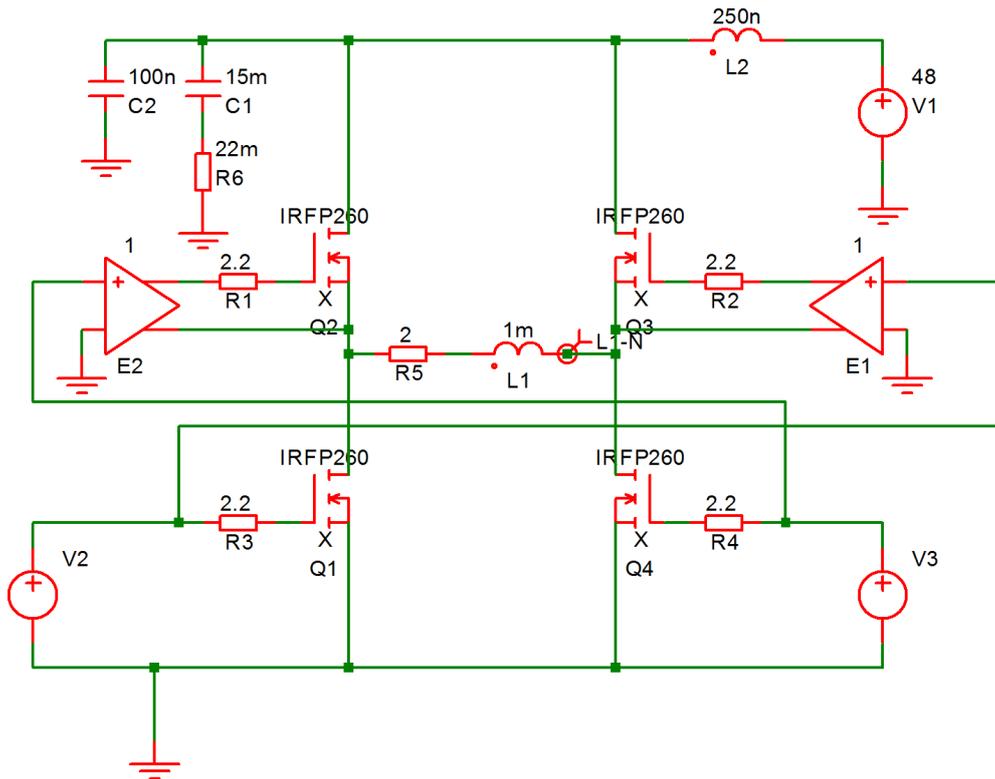
Ok Cancel Help

This does a decade sweep varying *sourceR* from 1K to 100k with 2 steps per decade. This is the result we get:



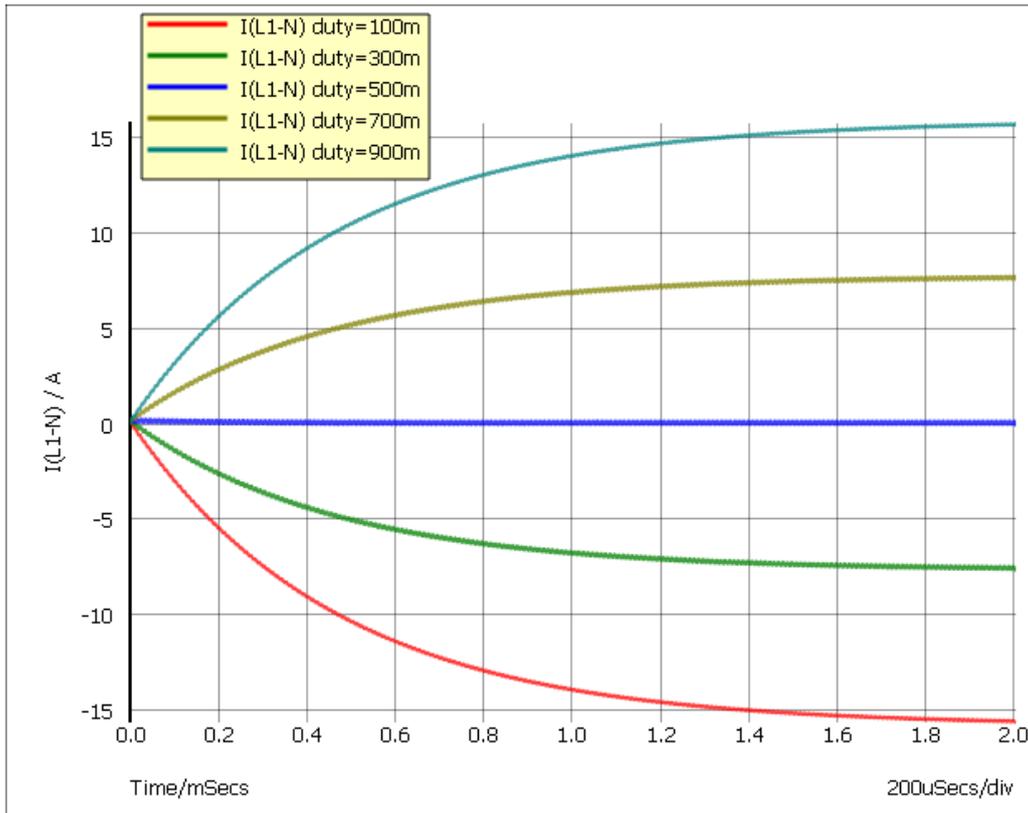
### 11.14.4 Example 2

The following circuit is a simple model of a full bridge switching amplifier used to deliver a controlled current into an inductance.



Sources V2 and V3 have been defined to be dependent on a parameter named *duty* which specifies the duty cycle of the switching waveform. See EXAMPLES/BRIDGE/BRIDGE.sxsch.

This was setup to perform a multi step analysis with the parameter *duty* stepped from 0.1 to 0.9. This is the result:



## 11.15 Safe Operating Area Testing

### 11.15.1 Overview

Safe Operating Area (SOA) is not a separate analysis mode, but a feature that can be enabled with DC or Transient analyses. With SOA testing, you can set maximum and minimum limits for any simulation quantity and the simulator will display when those limits are violated.

To use SOA testing, you must do two things:

1. Define the SOA limits for the models or devices you are using.
2. Enable and configure SOA testing.

Item 1 above is covered in detail in the *Simulator Reference Manual* - see section titled *Simulator Reference Manual/Command Reference/.SETSOA* and also the LIMIT parameter described in the section titled *Simulator Reference Manual/Command Reference/.MODEL*. Setting up simple limit tests using some simple schematic symbols is described below.

### 11.15.2 Defining Simple Limit Tests

#### Schematic Symbols

Three schematic symbols are provided that allow the definition of simple limit tests that report the following:

1. Over and under voltage on a single node.

2. Over and under current on a single device pin.
3. Over and under differential voltage on a node pair.

Use the following menus to place these devices:

**Place | Probe | Watch Voltage**

**Place | Probe | Watch Current**

**Place | Probe | Watch Differential Voltage**

Each of these symbols can be edited in the usual way. Each has three parameters that specify

1. The minimum limit. Use a large negative number (e.g. -1e100) if you don't wish to specify a minimum limit.
2. The maximum limit. Use a large positive number (e.g. 1e100) if you don't wish to specify a maximum limit.
3. A label. The default value is %REF% that will resolve to the device's part reference. You can enter any literal value instead.

## Setting Up SOA Testing

1. Select menu **Simulator | Choose Analysis...**
2. Select the SOA tab.
3. Under SOA mode choose either Summary output or Full output. In summary output mode, only the first violation for each SOA device will be reported. In full output mode, all violations are reported.
4. In Results to: choose where you would like the results reported. Note that writing results to the message window is a time consuming operation and you avoid selecting this option if you are expecting a large number of violations.

## Running Simulation

Run the simulation in the normal way. If there are any violations, the results will be reported in the location or locations specified in the Results to: section.

### 11.15.3 Advanced SOA Limit Testing

The simulator control .SETSOA allows much more sophisticated definitions for SOA limits. In particular, you can define limits for all devices belonging to a specified model. Suppose that you are using a BJT model that has a Vcb limit of 15V. While you could place a differential voltage watch device across each instance of this model, this would be time consuming and error prone. Instead, you can define a single .SETSOA control that refers to the model name of the device. The simulator will then automatically set up the limit test for every instance of that model.

You would usually enter a .SETSOA control in the schematic editor's **F11** window. See [“Manual Entry of Simulator Commands” on page 42](#) for details. Refer to *Simulator Reference Manual/Command Reference/.SETSOA* for details about .SETSOA .

It is also possible to set up an SOA specification for a model within the .MODEL control. Again, see *Simulator Reference Manual/Command Reference/.MODEL*.

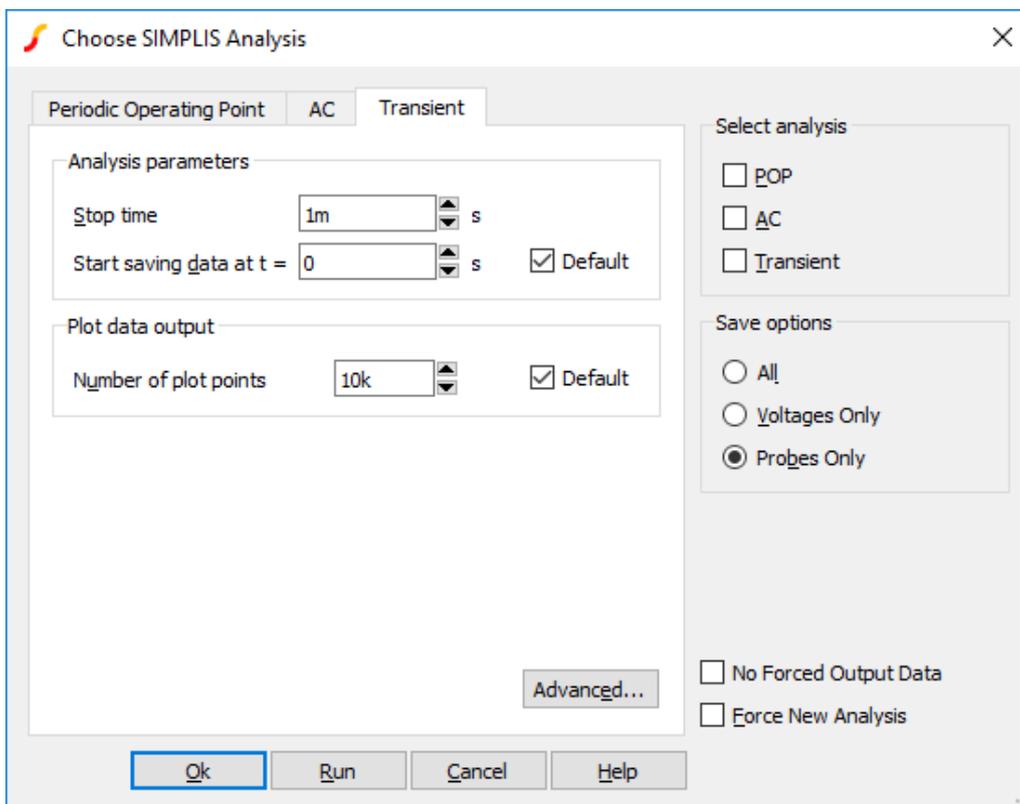
# Chapter 12

## SIMPLIS Analysis Modes

The SIMPLIS simulator is supplied with the SIMetrix/SIMPLIS product. For information on SIMPLIS see “What is SIMPLIS?” on page 3.

In this chapter we explain the analysis modes available with the SIMPLIS simulator. There is more information on SIMPLIS analysis modes including full details of the netlist commands required to invoke them, in the SIMPLIS Reference Manual.

To setup a SIMPLIS simulation, you must first set the schematic editor to SIMPLIS mode. See “Simulation Modes - SIMetrix or SIMPLIS” on page 31 for details. To set up a SIMPLIS analysis select menu **Simulator | Choose Analysis....** You will see this dialog box:



SIMPLIS offers three analysis modes namely Transient, AC and Periodic Operating Point or POP. These

analysis modes are described in detail in the SIMPLIS Reference Manual. The meaning of each of the controls is described in this chapter.

As with SIMetrix, you can also enter the raw netlist command in the **F11** window. The contents of this window remain synchronised with the Choose Analysis dialog box settings so you can freely switch between the two methods.

## 12.1 Transient Analysis

SIMPLIS transient analysis is similar to SIMetrix transient analysis.

### 12.1.1 Setting up a Transient Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select Transient check box on the right.
3. Select Transient tab at the top. Enter parameters as described in the following sections.

#### Analysis Parameters

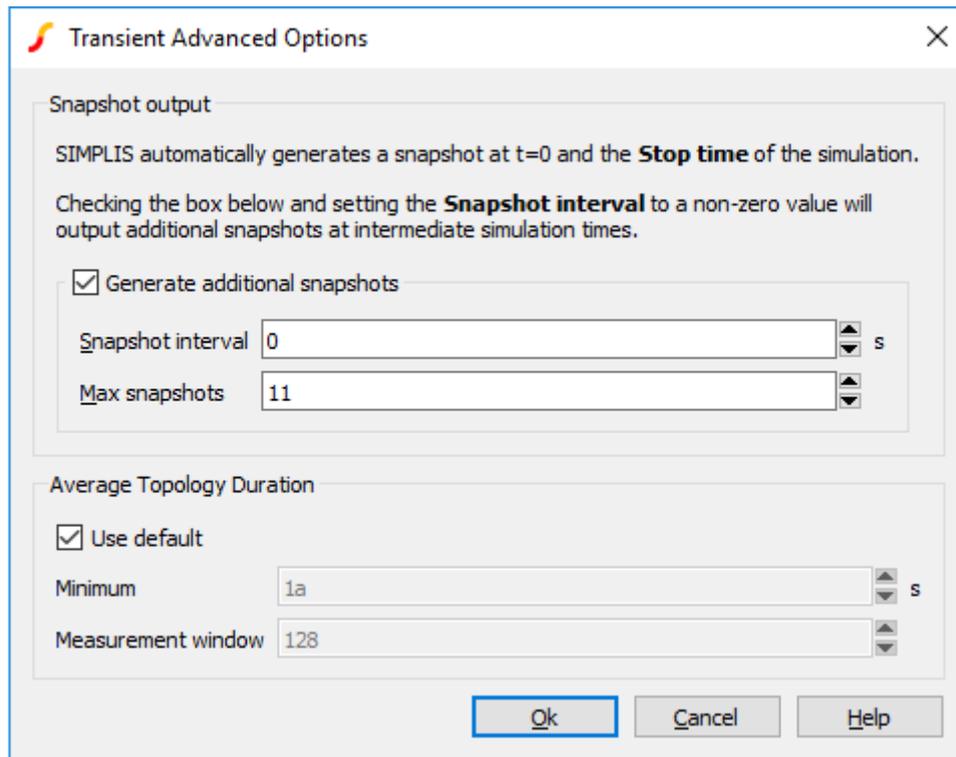
Stop Time	The finish time of a transient analysis.
Start Saving Data @	The data required to create plots will start being output at this time. (Start plotting data @ under Plot data output has a similar function but is subtly different. See below for details)

#### Plot data output

Number of Plot Points	The total number of points to be generated. These will be evenly spaced within the start and stop times.
-----------------------	--

#### Advanced...

Pressing the Advanced button opens the following dialog



## Snapshot output

SIMPLIS has the ability to save its internal state in order to allow a run to be repeated from a certain time point. This allows a run to be continued from where it previously left off. (Similar to SIMetrix transient restart facility). The internal saved states are known as *snapshots*.

SIMPLIS always saves a snapshot at the end of every run so if you start a new run of the same circuit with a start time (Start saving data at t= ) equal to the stop time of the previous run, SIMPLIS will not need to rerun the start and instead will load the snapshot state. SIMPLIS will do this automatically.

The entries in this dialog section allow you to specify the saving of snapshots at other times as well as the end of a run. This might be useful if you wanted to restart a run at some before the end of the previous run.

- |                        |  |
|------------------------|--|
| Enable snapshot output | Check this box to enable saving of snapshot data. (Snapshots are always saved at the end of a run)                           |
| Snapshot interval      | This is the minimum duration between snapshots.  |
| Max snapshots          | This is the maximum number of snapshots that will be saved. This setting overrides Snapshot interval if there is a conflict. |

## Average Topology Duration

SIMPLIS calculates the average time it spends in each topology over a defined number of topologies. If this value falls below a minimum value the simulation aborts. The entries in the Average Topology Duration group define the parameters for this feature as follows:

Minimum	If the average time falls below this threshold the simulation will abort
Measurement window	Number of windows over which the average time will be calculated

The purpose of this is to resolve problems with the simulation apparently getting 'stuck' in situations where there are unexpected very high speed oscillations. If this happens you may wish to increase the minimum time or reduce the measurement window as appropriate.

## 12.2 Periodic Operating Point (POP)

Periodic Operating Point (POP) finds a steady state operating point of switched systems that are periodically driven or self-oscillating. The predominant application of this analysis mode is to rapidly find the settled condition of a switching power supply without having to simulate the entire power up sequence. This dramatically speeds up the analysis of design's behaviour under different load conditions.

For further details of POP analysis see the SIMPLIS Reference Manual.

### 12.2.1 Setting up a POP Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select POP check box on the right.
3. Select POP tab at the top:

Periodic Operating Point    AC    Transient

POP Trigger source

Use "POP Trigger" schematic device  
(Commonly Used Parts->POP Trigger)

Custom POP Trigger gate    POP Trigger Schematic Device

Trigger condition

**f** Rising edge (logic low to logic high)

**⌋** Falling edge (logic high to logic low)

Timing

Maximum period    1u    s

Cycles before launching POP    5    Cycles

Advanced...

Enter parameters as described in the following sections.

## POP Parameters

Triggering - Use "POP Trigger" Schematic Device

POP analysis requires a trigger signal to indicate the start of each periodic cycle. The best way to define this is using a special schematic part. To place this select menu **Place | Analog Functions | POP Trigger**. You should check this box if you are using this part.

Trigger gate

If you do not use the schematic POP trigger device (see above) you must specify a suitable part in this edit box. Enter the full part reference of the device.

Trigger Condition

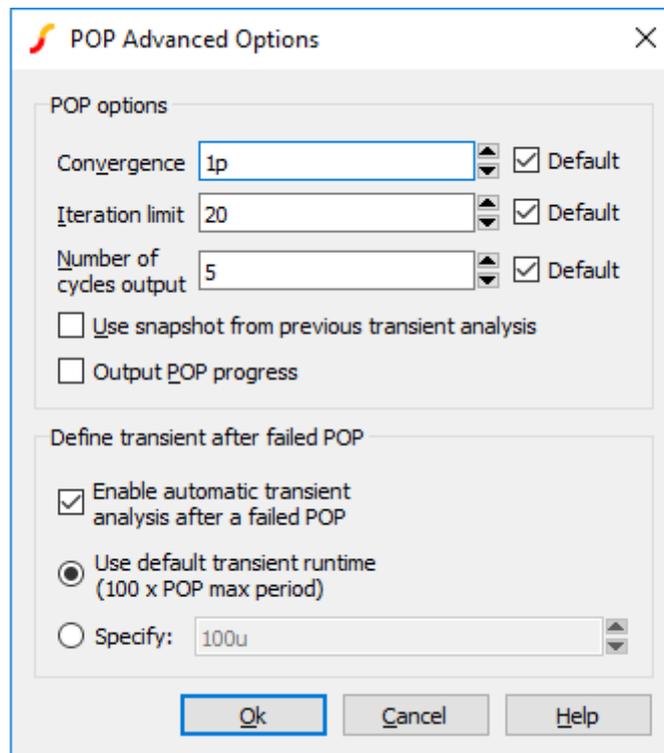
The polarity of the trigger edge.

## Conditions

Max. period	You should set this to a value that is larger than the expected period of your circuit's switching cycle. During each run SIMPLIS expects to see valid trigger conditions. However, if there is a fault in the design of the circuit or a fault in the definition of the trigger conditions, it is possible that none will be detected. The Max. period prevents SIMPLIS from carrying on indefinitely in such an event.
Cycles before launching POP	SIMPLIS will run a number of switching cycles in a normal transient analysis before starting the periodic point algorithm. This can speed up convergence of POP or in some cases make the difference between POP converging and not converging. If you find POP does not converge, increasing this value can sometimes help.

## Advanced - POP Options

Press the Advanced... button for more POP options:



Convergence	Sets the convergence criteria for the periodic operating point analysis. The convergence criteria is satisfied when the relative change in each state variable, between the start and end of a switching cycle, is less than this parameter.
Iteration limit	Sets the maximum number of iterations for the periodic operating point analysis.

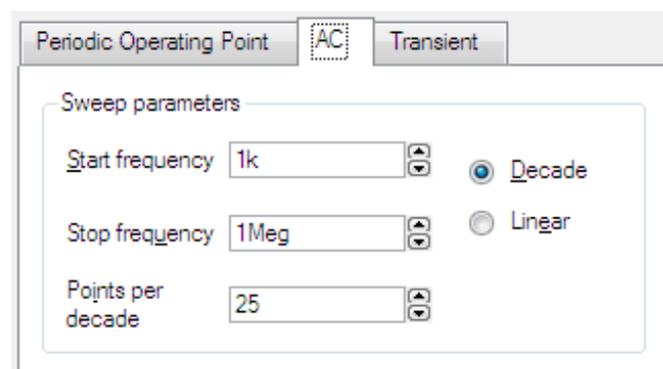
Number of cycles output	After a successful POP analysis, and if there is no transient analysis specified, SIMPLIS will generate the steady-state time-domain waveforms for an integral number switching cycles. This option sets the number of cycles.
Use snapshot from previous transient analysis	If checked, POP is instructed to take advantage of the last data point of a previous transient simulation, assuming the circuit and the initial conditions remained the same between the two simulation runs.
Output POP progress	If checked the progress of the POP solution will be output to the data file for plotting etc. This is useful for debugging.
Enable automatic transient analysis after a failed POP	If POP fails, a transient analysis automatically follows. This is to help diagnose the cause of POP failure but is also useful in some cases where a subsequent transient may settle sufficiently to perform a study of load transient behaviour. For further details, refer to the <i>SIMPLIS Reference Manual</i> . See <i>SIMPLIS Reference Manual/Simplis-POP/statements Relating to POP Analysis/.POP/Statement for POP Analysis/Behaviour of POP Analysis after POP Convergence Failure</i> .
Use default transient runtime/Specify	Run time after failed POP. See above.

## 12.3 AC Analysis

AC is a small signal frequency domain analysis mode applied to a switching circuit. Please refer to the SIMPLIS Reference Manual for full details of this analysis mode. Note that AC analysis requires a POP analysis (see above) to be also defined.

### 12.3.1 Setting up an AC Analysis

1. Select menu **Simulator | Choose Analysis...**
2. Select AC check box on the right. Note that the POP check box is automatically checked when AC is checked.
3. Select AC tab at the top:



Enter parameters as described in the following sections.

## AC Sweep Parameters

Start frequency	Enter the start frequency for the AC sweep
Stop frequency	Enter the stop frequency for the AC sweep
Points per decade/Number of points	If a decade sweep is selected enter the number of points required for each decade. If a linear sweep is selected enter the total number of points for the analysis.
Decade/Linear	Select type of sweep.

## 12.4 SIMPLIS Options

### Save options

All	If selected, all voltages and currents will be saved.
Voltages only	If selected, only node voltages will be saved
Probes only	If selected only voltages and currents that are explicitly probed will be output.

### Other Options

Force New Analysis	This tells SIMPLIS to ignore any state information that it may have stored and which could be used to speed up the run. For example, any stored snapshots (see above) will not be used if this is selected.
No Forced Output Data	<p>If checked, SIMPLIS will not force a data point before and after every switching instant.</p> <p>Under most circumstances, this option should remain turned OFF. For very long simulations that generate extremely large data sets, the waveform viewer may be slow responding to user commands. In such cases, turning ON the NO_FORCED_DATA option will reduce the number of simulation data points displayed in the waveform viewer during each switching cycle. For long simulations that involve many switching instants in one switching cycle this reduction can be significant. Enabling this option in no way degrades the accuracy of the SIMPLIS solution, but it can potentially reduce the fidelity of the displayed waveforms within each switching cycle.</p>

## 12.5 Multi-step and Monte Carlo Analyses

### 12.5.1 Overview

The SIMetrix environment provides a facility to run automatic multiple SIMPLIS analyses. Two modes are available namely parameter step and Monte Carlo.

In parameter step mode, the run is repeated while setting a parameter value at each step. The parameter may be used within any expression to describe a device or model value.

In Monte Carlo mode runs are simply repeated the specified number of times with random distribution functions enabled. Distribution functions return unity in normal analysis modes but in Monte Carlo mode

they return a random number according to a specified tolerance and distribution. Any model or device parameter may be defined in terms of such functions allowing an analysis of manufacturing yields to be performed.

## 12.5.2 Comparison Between SIMetrix and SIMPLIS

The multi-step analysis modes offered in SIMetrix simulation mode achieve the same end result as the SIMPLIS multi-step modes but their method of implementation is quite different.

SIMetrix multi-step analyses are implemented within the simulator while the SIMPLIS multi-step analyses are implemented by the front end using the scripting language. The different approaches trade off speed with flexibility. The approach used for SIMPLIS is more flexible while that used for SIMetrix is faster.

## 12.5.3 Setting up a SIMPLIS Multi-step Parameter Analysis

### An Example

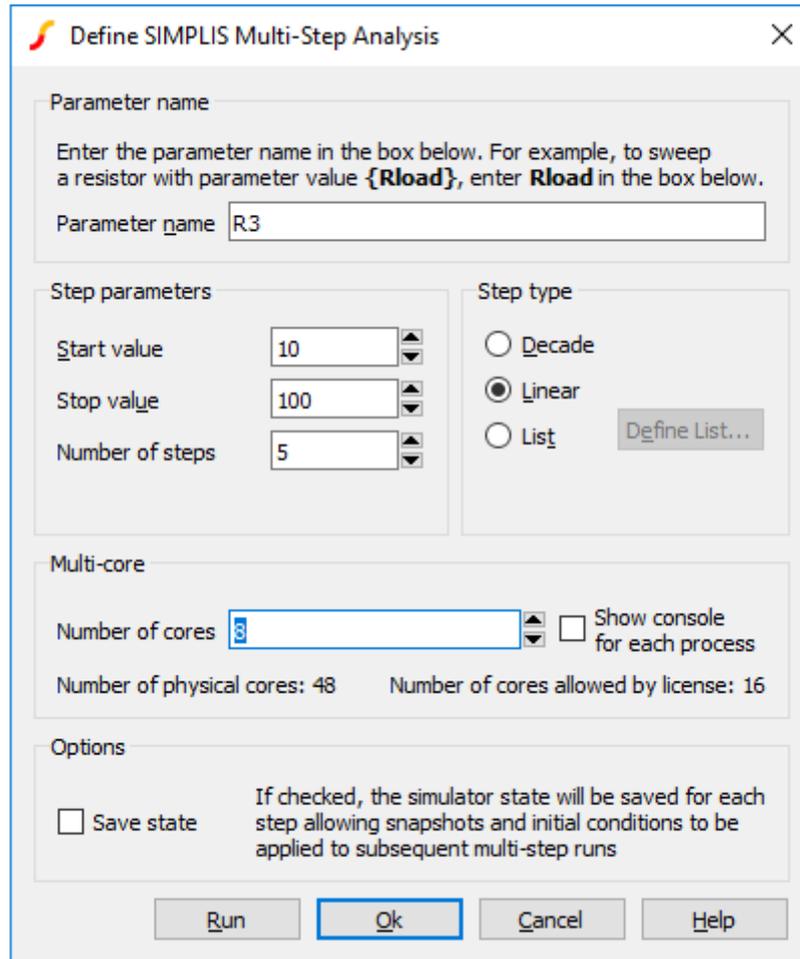
We will begin with an example and will use one of the supplied example schematics. First open the schematic Examples/SIMPLIS/Manual\_Examples/Example1/example1.sxsch. We will set up the system to repeat the analysis three times while varying R3. Proceed as follows:

1. First we must define R3's value in terms of an expression relating to a parameter. To do this, select R3 then press **shift-F7**. Enter the following:

```
{r3_value}
```

r3\_value is an arbitrary parameter name. You could also use 'R3'.

2. Select menu Simulator | Select Multi-step...
3. Enter 'r3\_value' for Parameter Name and set Start value to 20, Stop value to 100 and Number of steps to 3. This should be what you see:



#### 4. Press Run.

The analysis will be repeated three times for values of r3\_value of 20, 60 and 100. The resistor value R3 is defined in terms of r3\_value so in effect we are stepping R3 through that range.

In most cases you will probably want to step just one part in a similar manner as described above. But you can also use the parameter value to define any number of part or model values.

If you now run a normal single analysis, you will find that SIMPLIS reports an error as it is unable to resolve the value for R3. This can be overcome by specifying the value using a .VAR control. Add this line:

```
.VAR r3_value=100
```

to the **F11** window. This line defines the value of R3 when a normal single step analysis is run.

## Options

The above example illustrates a linear multi-step parameter run. You can also define a decade (logarithmic) run and also a list based run that selects parameter values from a list. To set up a list run, select the List radio button, then press Define List... Enter the values for the list using the dialog box.

## 12.5.4 Setting Up a SIMPLIS Monte Carlo Analysis

### An Example

To set up a Monte Carlo analysis, you must first define part tolerances. This is done by defining each value as an expression using one of the functions Gauss(), Unif or WC(). Here is another example. Open the same example circuit as above then make the following changes:

1. Select R3, press shift-F7 then enter the value

```
{ 100*GAUSS ( 0 . 05 ) }
```

2. Select C2, press shift-F7 then enter the value

```
{ 100u*GAUSS ( 0 . 2 ) }
```

3. Delete the fixed probes on the V1 input and on R1. (This is just to prevent too many unnecessary curves being plotted)

The above will give R3 a 5% tolerance and C2 a 20% tolerance with a 3 Sigma Gaussian distribution. Now set up the Monte Carlo run:

1. Select menu **Monte Carlo | Setup Monte Carlo...**
2. Enter the desired number of steps in **Number of steps**. To demonstrate the concepts, 10 will be sufficient, but usually a Monte Carlo run would be a minimum of around 30 steps
3. Press Run

You should see a series of curves build up as the run progresses.

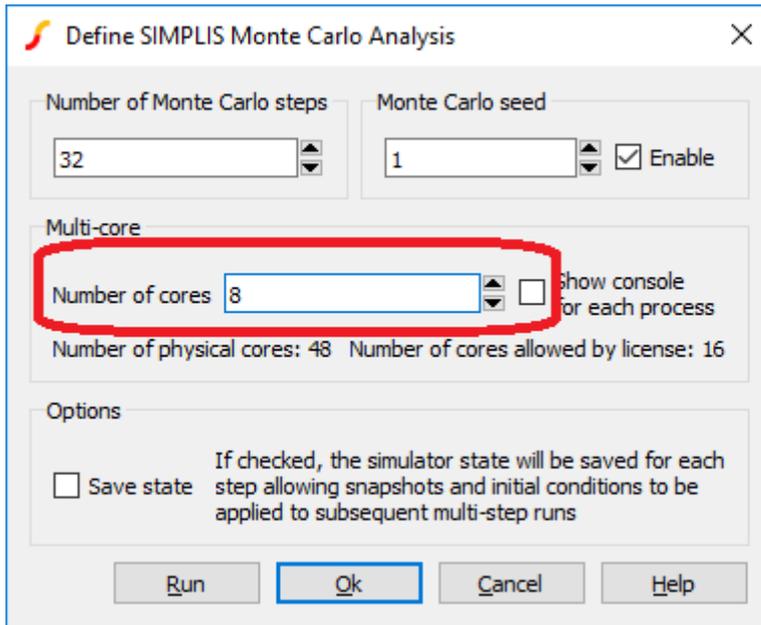
## 12.5.5 Multi-core Multi-step SIMPLIS Analyses

If your system and your license permit, you can specify multiple processor cores to be employed for multi-step analyses. This can substantially speed up the run. For example, suppose you have a 4 core system and wish to run a 100 step Monte Carlo analysis. The 100 steps may be split across the 4 cores each doing 25 steps each. In most cases this improve the run time by at least a factor of 3.

In this mode of operation, multiple SIMPLIS processes run independently with each one creating its own data file.

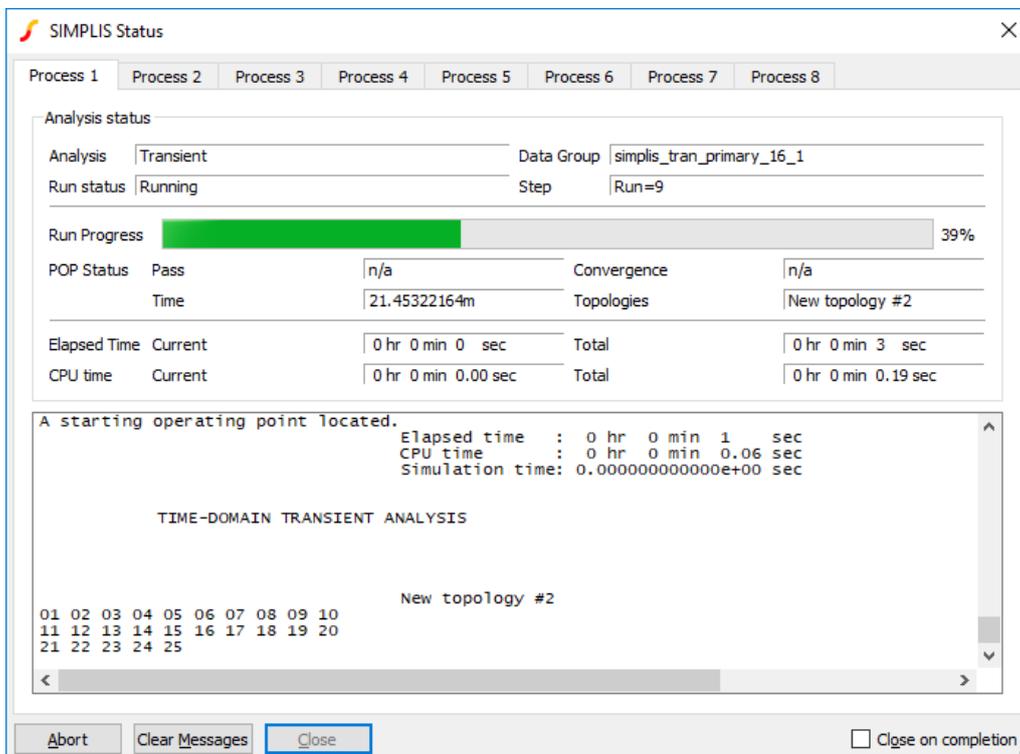
### To setup a Multi-core Multi-step SIMPLIS Analysis

Setup the multi-step analysis as normal. In the Define SIMPLIS Multi Step Analysis dialog box set the **Number of cores** to the desired value as shown below. Note that the maximum value you can set is determined by your license and system. For allowed number of cores for each product type, see [“Simulation and Multi-core Processors” on page 224](#).



### Running a SIMPLIS multi-core multi-step analysis

Start the run in the usual way for a multi-step run. You will see the usual SIMPLIS status box but with the difference that there will be multiple tabs, one for each process running as shown below:



You can watch the progress of each process using the tabs at the top of the status box.

## Using Fixed Probes with SIMPLIS Multi-core Multi-step Analyses

Fixed probes usually update the waveform viewer incrementally. That is, the display is updated while the simulation proceeds. This does not happen for multi-core multi-step runs. Instead, the graphs will be updated when the run is complete.

### 12.5.6 Tolerances and Distribution Functions

#### Distribution Functions

Tolerances are defined using distribution functions. For SIMPLIS Monte Carlo there are just three functions available. These are defined below.

Function Name	Description
GAUSS(tol)	Returns a random with a mean of 1.0 and a standard deviation of tol/3. Random values have a Gaussian or Normal distribution.
UNIF(tol)	Returns a random value in the range 1.0 +/- tol with a uniform distribution
WC(tol)	Returns either 1.0-tol or 1.0+tol chosen at random.

The 'L' and 'E' suffix functions available in SIMetrix Monte Carlo analysis are not available for SIMPLIS operation.

#### Lot and Device Tolerances

No special provision has been made to implement so called 'Lot' tolerances which model tolerances that track. However, it is nevertheless possible to implement Lot tolerances by defining a parameter as a random variable. Suppose for example that you have a resistor network consisting of 4 resistors of 1k with an absolute tolerance of 2% but the resistors match to within 0.2%. The absolute tolerance is the 'lot' tolerance. This is how it can be implemented:

1. Assign a random variable using the .VAR preprocessor control. (You cannot use .PARAM in SIMPLIS simulations). E.g.:

```
.VAR rv1 = {UNIF(0.02)}
```

2. Give each resistor in the network a value of:

```
{1K * rv1 * UNIF(0.002)}
```

rv1 will be updated on each Monte Carlo step but will always have the same value in each place where it is used.

### 12.5.7 Monte Carlo Log File

When a Monte Carlo analysis is run a log file is created which details which device values were changed and what they were changed to. Two versions of the log file are created; one is plain text and is called simplis\_mclog.log and the other is in HTML format and called simplis\_mclog.html.

To view the HTML log file, select menu **Monte Carlo | View Log File**

An example is shown below:

**SIMPLIS Monte Carlo Log**

Schematic: example1  
Date: 12/02/2019 11:07

Run	Reference	Actual	Nominal	Deviation
Run: 1 Seed=1	C2	9.91362317516603e-05	0.0001	-0.86376825%
	R3	105.338621663128	100	5.3386217%
Run: 2 Seed=16807	C2	0.000101253591776561	0.0001	1.2535918%
	R3	98.1929051586053	100	-1.8070948%
Run: 3 Seed=282475249	C2	9.91362317516603e-05	0.0001	-0.86376825%
	R3	105.338621663128	100	5.3386217%
Run: 4 Seed=1622647863	C2	0.000104087778144502	0.0001	4.0877781%
	R3	99.2827070313065	100	-0.71729297%
Run: 5 Seed=947787489	C2	0.000100723388598558	0.0001	0.72338860%
	R3	100.392538941292	100	0.39253894%
Run: 6 Seed=1578110407	C2	9.90199720258228e-05	0.0001	-0.98002797%
	R3	103.125444801103	100	3.1254448%
Run: 7 Seed=1878557649	C2	9.9786860466105e-05	0.0001	-0.21313953%
	R3	99.8988103835267	100	-0.10118962%

Ok

## 12.5.8 Monte Carlo Seed Values

### Introduction

The random values used in Monte Carlo analysis are generated using a pseudo-random number generator. This generates a sequence of numbers that appear to be random and have the statistical properties of random numbers. However, the sequence used is in fact repeatable by setting a defined seed value. The sequence generated will always be the same as long as the same starting seed value is used.

This makes it possible to repeat a specific Monte Carlo step as long as the seed value is known. SIMetrix/SIMPLIS explicitly sets a random seed value at the start of each Monte Carlo run and records this in a file. A facility exists to explicitly set the seed value instead of using a random seed and this makes it possible to repeat a specific step. This is useful in cases where one or more Monte Carlo steps show unexplained behaviour and further investigation is needed.

This is also useful if you wish to find the values of the parts used for a particular Monte Carlo step.

### Finding the Seed Value

To find the seed value of a particular step, follow this procedure:

1. Run normal Monte Carlo without seed definition if you have not done so already.
2. Find the run number of the result whose seed you require (so that you can repeat that step). You can find the run number from a plotted waveform from the Monte Carlo run. Plot a result as normal then position the small cursor on the curve of interest. The run number will be displayed in the status bar on the right.

3. Select menu **Monte Carlo | View Log File** to show the log file. The seed value is shown against the run number. See “[Monte Carlo Log File](#)” on page 282

## Using the Seed Value

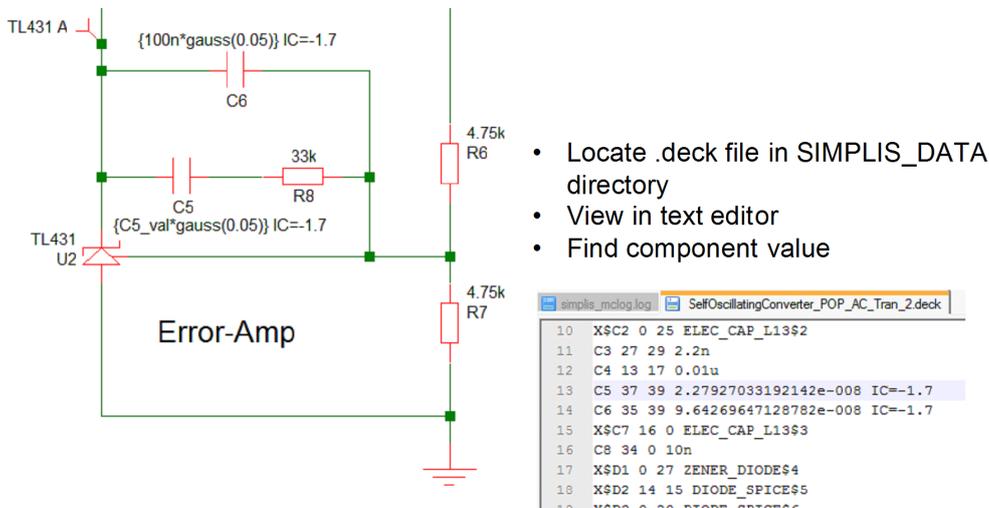
You can use the seed value to repeat the run associated with it. Follow this procedure:

1. Select menu **Simulator | Setup Multi-step**. In the Monte Carlo Seed group check the Enable box and enter the seed value obtained from step 3 in the above section. Set the Number of steps to 1.
2. Run the simulation as normal.

## Finding Part Values Used

The part values used for a particular step can be found by studying the .deck file that is created for every SIMPLIS run. To do this, follow the procedure for finding and using the seed value as described above. Then after the single run is complete locate the file with the .deck extension. This will be located in the SIMPLIS\_DATA folder and will have a name the same as the top level schematic but with the extension .deck. So if your schematic is called DC-DC-Converter.sxsch, the deck file will be called DC-DC-Converter.deck and will be located in the SIMPLIS\_DATA folder.

The .deck file is an ASCII file that list the parts in the design with their connections and values. Its format is similar to a SPICE netlist. The picture below illustrates the process of matching a part in the schematic to an entry in the .deck file:



## 12.6 Initial Condition Back-annotation

### 12.6.1 Overview

On each run, SIMPLIS generates a file called the initial condition file. This contains a sequence of SIMPLIS netlist commands that initialises a circuit to the state achieved at the end of the run. This allows a new run to continue from where a previous run completed.

The initial condition file can be applied by including it in the netlist for a new run and in some instances this may be the most convenient method. However, it is also possible to annotate the schematic with the initial condition information. This has some advantages:

1. The initial conditions back annotated to top level capacitors and inductors will also be recognised in SIMetrix simulation mode.
2. Back annotated initial conditions are attached to schematic instances and will be faithfully reproduced if, for example, a schematic block is copied and pasted to another schematic.

Please read all of the sections below on back-annotation and ensure you correctly understand all the issues involved.

### 12.6.2 How to Back-annotate a Schematic

Simply select menu **Simulator | Initial Conditions | Back-annotate**. You will notice a second or two of activity in the schematic and then the operation is complete.

You should note that SIMetrix/SIMPLIS does not distinguish between initial conditions that are back-annotated and initial conditions that are applied manually. After running the back-annotation algorithm, you will not be able to restore the initial condition value to those set before. You can, however, use Undo in the normal way and in fact the back-annotation operation will be reversed with a single Undo operation.

### 12.6.3 Disable/Enable Initial Conditions

To disable initial conditions select menu **Simulator | Initial Conditions | Disable**. Note that this will disable all initial conditions defined at the top level, not just ones that are back-annotated. To re-enable use the menu **Simulator | Initial Conditions | Enable**.

### 12.6.4 Back-annotation Errors

If you get the error message `"The following instances have initial condition values but do not support back annotation"` it means that the `SIMPLIS_TEMPLATE` property is protected for the instances listed. To fix the problem remove the protection on this property. You will need to open the symbol in the symbol editor to do this.

In order to apply back-annotation in a generic fashion, SIMetrix needs to modify the `SIMPLIS_TEMPLATE` property, but cannot do so if it protected hence the error message. You shouldn't get this error with any standard symbols from the SIMetrix v5 library or later, but you may get it with your own symbols or symbols from an earlier library.

### 12.6.5 Editing Back-annotated Initial Conditions

How you change the value of an back-annotated initial condition depends on the device. If the device already has a user editable initial condition, then simply use the standard method. With capacitors and

inductors, this is simply done using **F7** or the Edit Part... menu. With some other devices, the initial condition value may be found in the Edit Additional Parameters menu.

For devices that do not have user editable initial conditions, you should use the Edit Additional Parameters menu. This applies to most subcircuit models and to all hierarchical blocks.

### 12.6.6 How Does it Work?

The initial condition file specifies the value of initial conditions for each device that requires them. This information must then be applied to each schematic instance in an appropriate manner. Two basic approaches are used to apply the initial condition values depending on the device namely the specialised method and the generic method.

In the specialised method, a special script is called which edits one or more properties of the schematic instance. With a capacitor for example, the VALUE property is edited so that the IC parameter is specified or modified. Something similar is done for inductors. This action is done using a special script specified by the INIT\_SCRIPT property. In the case of the capacitor, the script 'ic\_reactive' is called. The advantage of the specialised method is that the device can be modified in a manner that is consistent with its existing user interface. Capacitors already have user editable initial conditions and the application of back-annotated initial conditions is compatible with this.

The disadvantage of the specialised method is that a method of applying the back annotated value needs to be developed for every different type of device. This would not be acceptable for most users who develop their own symbols. The generic method overcomes this difficulty. The generic method modifies the properties so that additional netlist lines are created containing the .INIT simulator command that defines the initial conditions. To achieve this the SIMPLIS\_TEMPLATE property needs to be modified and as long as this isn't protected, the generic method will always work.

### 12.6.7 Hierarchical Blocks and Subcircuits

All back-annotated initial conditions are applied at the top level and no child schematics or subcircuits will be modified.

This introduces a potential problem in that once back-annotated initial conditions are applied, you will no longer be able to modify individual initial conditions within a hierarchical block. You will only be able to edit them on the top level device using the **Edit Additional Parameters... menu**.

You will be able to use initial conditions defined within a hierarchy or subcircuit if you first disable top level initial conditions using the Initial Conditions | Disable menu. This will of course disable all initial conditions specified at the top level.

To disable initial conditions for a single hierarchical block, use the Edit Properties menu to set the USEIC property to 0. Note that the Enable and Disable menus will reset this property.

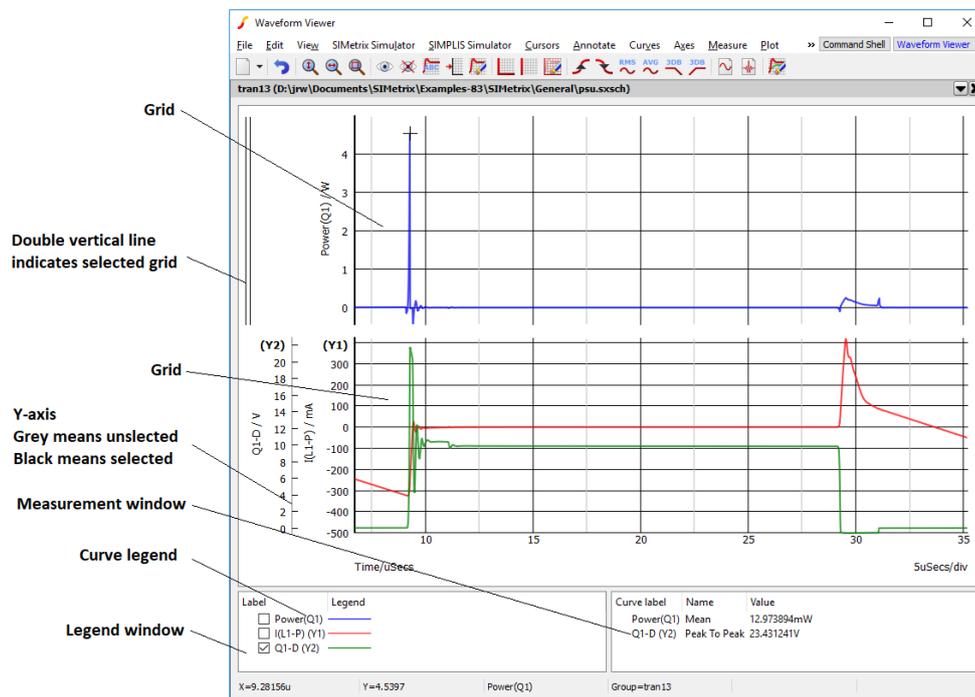
# Chapter 13

## Graphs, Probes and Data Analysis

The basics of how to create graphs of your circuit's signals were explained in "Getting Started" on page 31. This chapter provides a full reference on all aspects of probing and creating graphs.

### 13.1 Elements of the Graph Window

#### 13.1.1 Main Window

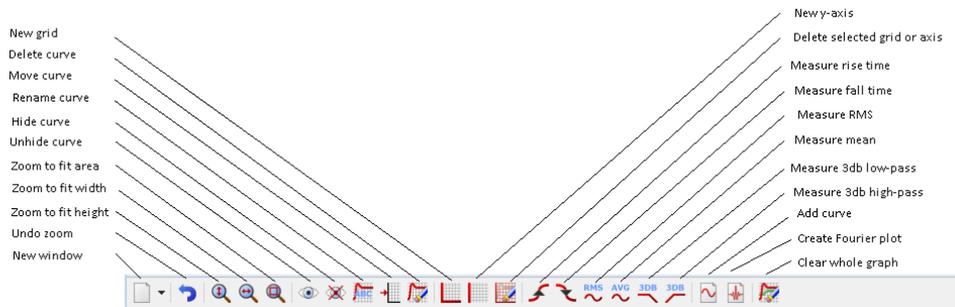


#### 13.1.2 Windows and Tabbed Sheets

Normally new graphs are created within the same window as a tabbed sheet. A row of tabs will appear at the top of the graph window allowing you select which graph you wish to view. You can also create a new

graph window using the menu **File | New Graph** . This will create an empty window to which you may add new graphs.

### 13.1.3 Graph Toolbar



The above shows the function of each of the buttons on the graph toolbar. These are referred to in the following sections.

## 13.2 Probes: Fixed vs. Random

Much of this section and some of the next have already been covered in [“Plotting Simulation Results” on page 43](#). It is repeated here for convenience.

SIMetrix provides two approaches to creating plots of simulated results from a schematic.

The first approach is to fix voltage or current probes to the schematic before or during a run. SIMetrix will then generate graphs of the selected voltages and/or currents automatically. Normally the graphs for fixed probes are opened and updated while the simulator is running. The probes have a wide range of options which allow you to specify - for example - how the graphs are organised and when and how often they are updated. These probes are known as *fixed probes*.

The second approach is to randomly probe the circuit after the run is complete. (You can also do this during a run by pausing first). With this approach, the graph will be created as you point the probe but will not be updated on a new run. These probes are known as *random probes*.

You do not need to make any decisions on how you wish to probe your circuit before starting the run. You can enter a circuit without any fixed probes, run it, then randomly probe afterwards. Alternatively, you can place a single fixed probe on an obvious point of interest, then randomly probe to investigate the detailed behaviour of your circuit. Note that you can add fixed probes after a run has started but the run must be paused first.

There are many types of fixed probe to suit a range of applications. The random probing method allows you to plot anything you like including device power, FFTs, arbitrary expressions of simulation results and X-Y plots such as Nyquist diagrams. The arbitrary fixed probe provides the same flexibility for fixed probes. See [“Arbitrary Fixed Probe” on page 319](#) for details.

## 13.3 Fixed Probes

There are many types of fixed probe as described in the following table

Probe Type	Description	To Place
Voltage	Single ended voltage. Hint: If you place the probe immediately on an existing schematic wire, it will automatically be given a meaningful name related to what it is connected to	Menu: <b>Probe   Place Fixed Voltage Probe...</b> Hot key: B
Current	Device pin current. A single terminal device to place over a device pin	Menu: <b>Probe   Place Fixed Current Probe...</b> Hot key: U
Inline current	In line current. This is a two terminal device that probes the current flowing through it.	Menu: <b>Probe   Place Inline Current Probe...</b>
Differential voltage	Probe voltage between two points	Menu: <b>Probe   Place Fixed Diff. Voltage Probe...</b>
dB	Probes db value of signal voltage. Only useful in AC analysis	Menu: <b>Probe AC/Noise   Fixed dB Probe...</b>
Phase	Probes phase of signal voltage. Only useful in AC analysis	Menu: <b>Probe AC/Noise   Fixed phase Probe...</b>
Bode plot with Measurements	Plots db and phase of vout/vin. Connect to the input and output of a circuit to plot its gain and phase. Includes optional measurements for phas margin, gain margin and gain crossover frequency	Menu: <b>Probe AC/Noise   Bode Plot Probe - with Measurements</b>
Bode plot - basic version	Plots db and phase of vout/vin. Connect to the input and output of a circuit to plot its gain and phase.	Menu: <b>Probe AC/Noise   Bode Plot Probe - Simple</b>
Bus plot	Plots bus signal in 'logic analyser' style	Menu: <b>Probe   Place Fixed Bus Probe</b>
Power plot	Plots the power in a device. The power probe must be attached to a single pin of the device. It doesn't matter which pin - the power plotted in the instantaneous power in the <i>whole</i> device	Menu: <b>Probe   Place Fixed Power Probe...</b>
Per Cycle	Plot a graph based on a calculation performed for each complete cycle of a repetitive waveform. See <a href="#">"Per Cycle Probes"</a> on page 296	Available from Parts Selector
X-Y	Plot a graph of one signal on the y-axis with respect to a second signal on the x-axis. See <a href="#">"X-Y Probes"</a> on page 296	Menu: <b>Probe   Place XY Probe</b>
Fourier Probe	Create a Fourier plot. See <a href="#">"Fourier Analysis"</a> on page 316	Menu: <b>Probe   Place Fourier Voltage Probe</b>

Probe Type	Description	To Place
Arbitrary Probe	Create a custom probe with any number of voltage or current inputs related by a user-entered expression. See “Custom Fixed Probe” on page 319	Menu: <b>Probe   Create and Place Arbitrary Fixed Probe</b>

These probes are schematic symbols with special properties. When you place a fixed probe on the schematic, the probed value at the point where you place the probe will be plotted each time you run the simulation.

Current probes and power probes must be placed directly over a part pin. They will have no function if they are not and a warning message will be displayed.

### 13.3.1 Fixed Probe Options

All probe types have a large number of options allowing you to customise how you want the graph plotted. For many applications the default settings are satisfactory. In this section, the full details of available probe options are described. Select the probe and press **F7** or menu **Edit Part...** The following dialog will be displayed for voltage, current, power, db and phase probes:

The elements of each tabbed sheet are explained below.

## Probe Options Sheet

Curve Label	Text displayed by the probe on the schematic and also used to label the resulting curves
History	<p>Controls the number of curves from previous runs that will be displayed. If set to 1, only the latest result will be shown. If set to a value greater than one, the current result and a number of previous results equal to the history depth less one will be shown. The old results will be shown in broken lines.</p> <p>If <b>default</b> is checked, the number of curves will be based on the values specified in the general options.</p>

To set the history depth value in general options, follow these steps:

1. From the menu bar, select **File | Options | General...**, and then select the Graph/Probe/Data Analysis tab
2. In the Fixed probe global options section, enter a value in the Default history depth field, and then click Ok

If **Separate curves** is checked the history depth setting is ignored. In this case a new curve is created for each new run.

See [“Curve History” on page 294](#) for more information



- Display order** Enter a string to control the grid display order. The value is arbitrary and will not be displayed. To force the curve to be placed above other curves that don't use this value, prefix the name with '!'. The '!' character has a low ASCII value. Conversely, use ' ' to force curve to be displayed after other curves.
- Colour** Check **Use default** to minimise duplicate colours on the same graph by allowing the colour to be chosen automatically. Alternatively uncheck, this box and then press **Edit...** to select a colour of your choice. In this case the trace always has the same colour.
- Plot on completion only**
- If checked, incremental plotting is disabled; that is, the plot does not appear until the run is complete.
  - If not checked, the curve updates regularly during the course of the run.
- Note: For some expressions, it is not possible to incrementally plot the data; if this is the case incremental plotting is disabled and this check box is ignored.

## Axis Scales Sheet

The screenshot shows the 'Edit Probe' dialog box with the 'Axis Scales' tab selected. The dialog is divided into two main sections: 'X-Axis' and 'Y-Axis'. Each section contains three radio button options for the scale type: 'Lin', 'Log', and 'Auto'. In both sections, 'Auto' is selected. Below these options are three radio button options for the scale mode: 'No change', 'Auto scale', and 'Defined'. In both sections, 'No change' is selected. At the bottom of each section are two spin boxes for 'Min' and 'Max' values, both set to 0 and 1 respectively. The 'Ok' button at the bottom center is highlighted with a blue border.

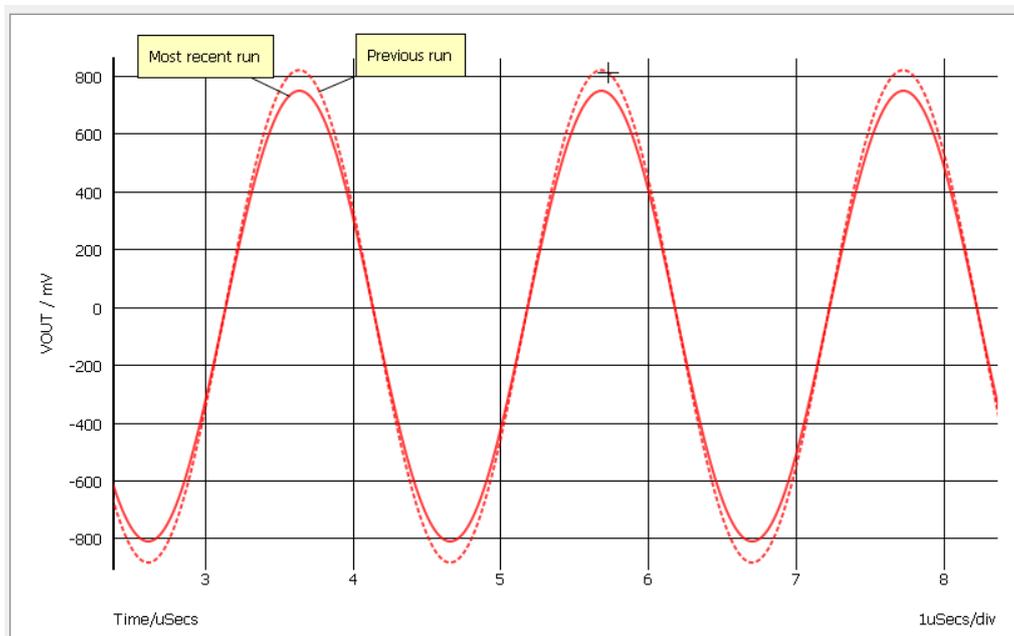
Parameter	Description
Lin/Log/Auto	Specify whether you want X-Axis to be linear or logarithmic. <ul style="list-style-type: none"> <li>• If you select Auto, the axis (X or Y) will be set to log if the x values are logarithmically spaced.</li> <li>• For the Y-axis, it is also necessary that the curve values are positive for a log axis to be selected.</li> </ul>
No Change Auto scale Defined	Controls how the axis limits are defined <ul style="list-style-type: none"> <li>• If you select No change, the axis scales remain as they are; this is relevant only if you are adding to an existing graph.</li> <li>• If you select Auto scale, the axis scales are based on the minimum and maximum values of the curve.</li> <li>• If you select Defined, enter values in the Min and Max fields.</li> </ul>

## Axis Labels Sheet

This sheet has four edit boxes allowing you to specify, x and y axis labels as well as their units. If any box is left blank, a default value will be used or will remain unchanged if the axis already has a defined label.

### 13.3.2 Curve History

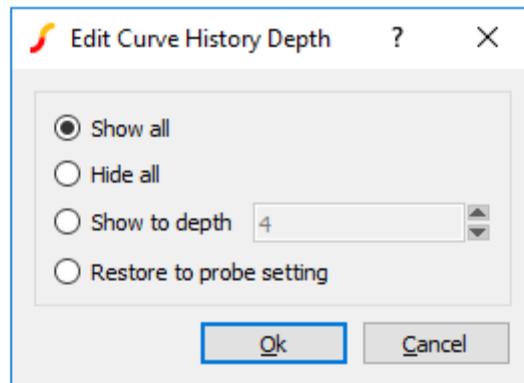
Curves created by fixed probes store their history which can be viewed at any time. The picture below shows how the current result and one previous result are displayed.



By default, the waveform viewer will *store* the current result and three previous results. How many previous results are actually *displayed* may be changed in a number of ways:

1. The history depth setting in the probe dialog

2. Menus **Curve | History | Set Depth - All Curves** and **Curve | History | Set Depth - Selected Curves**, set the displayed history depth after the curve has already been plotted. These menus display this dialog box:



Show all	Show full history (usually up to 4 deep)
Hide all	Hide all history leaving just one curve on display
Show to depth	Show history up to specified depth. For example, if you set a depth of 2, the latest result and one previous result will be displayed
Restore to probe setting	Sets the depth that was defined in the fixed probe that created the curve. Note that this sets the depth to whatever the fixed probe setting was at the time the curve was created, not the current fixed probe setting

The stored history depth (not the displayed history depth) may be changed using the `DataHistoryDepth` option setting. This has a default value of 4 meaning that the current result and three previous results will be stored. Note that in some circumstances the depth will be restricted to preserve memory.

### 13.3.3 Fixed Bus Probe Options

Select device and press **F7** in the usual manner. A dialog box will show similar to that shown in “[Bus Probe Options](#)” on page 329. But you will notice an additional tabbed sheet titled **Probe Options**. This allows you to select an axis type and graph in a similar manner to that described above for fixed voltage and current probes.

### 13.3.4 Using Fixed Probes in Hierarchical Designs

Fixed probes may successfully be used in hierarchical designs. If placed in a child schematic, a plot will be produced *for all instances* of that child and the labels for each curve will be prefixed with the child reference.

### 13.3.5 Adding Fixed Probes After a Run has Started

When you add a fixed single ended voltage or current probe after a run has started, the graph of the probed point opens soon after resuming the simulation. To do this:

1. Pause simulation.

2. Place a probe on the circuit in the normal way.
3. Resume simulation

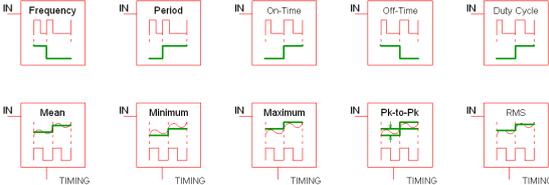
### 13.3.6 Per-cycle Probes

Per Cycle Probes first calculate a new curve based on a built-in formula and then plot the curve on a graph. The input can be one of the following:

- Single-ended voltage
- Differential voltage
- Current
- Difference of two currents

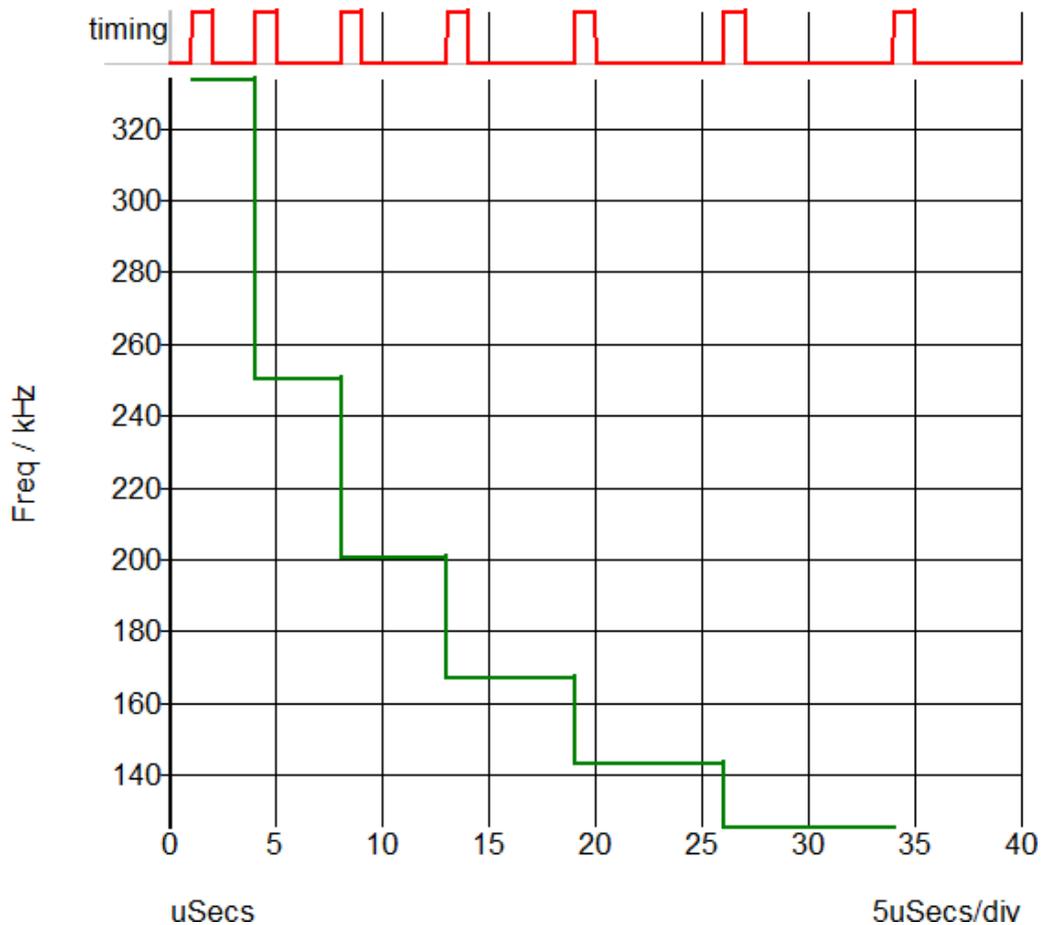
As with the other probes, the curve can be output to any graph and axis. Per Cycle Probes are compatible with both simulators.

The Per-cycle Probe specification is shown in the following table:

Per-cycle Probes	
Model Name:	Per Cycle Probes
Simulator:	The device is compatible with the SIMetrix and SIMPLIS simulators
SIMetrix Parts Selector Location:	<b>Probes   Per Cycle Voltage Probes Probes   Per Cycle Diff Voltage Probes Probes   Per Cycle Current Probes Probes   Per Cycle Diff Current Probes</b>
SIMPLIS Parts Selector Location:	<b>Commonly Used Parts   Per Cycle Voltage Probes Commonly Used Parts   Per Cycle Diff Voltage Probes Commonly Used Parts   Per Cycle Current Probes Commonly Used Parts   Per Cycle Diff Current Probes</b>
Symbol Library:	None the symbol is automatically generated when placed 
Symbols:	
Multiple Selections:	Multiple probes can be selected and edited simultaneously.

### Per-Cycle Timing Measurements

These advanced probes allow you to visualize subtle behaviour which is not otherwise easy to see. One example is to visualize how the switching frequency varies over the simulation time window. In this case, the probe first finds all edges which match the trigger conditions and then calculates a new curve with the frequency measurements on the vertical axis and the input time vector on the horizontal axis. An example waveform showing a pulse train and the per cycle frequency of the curve is shown below:

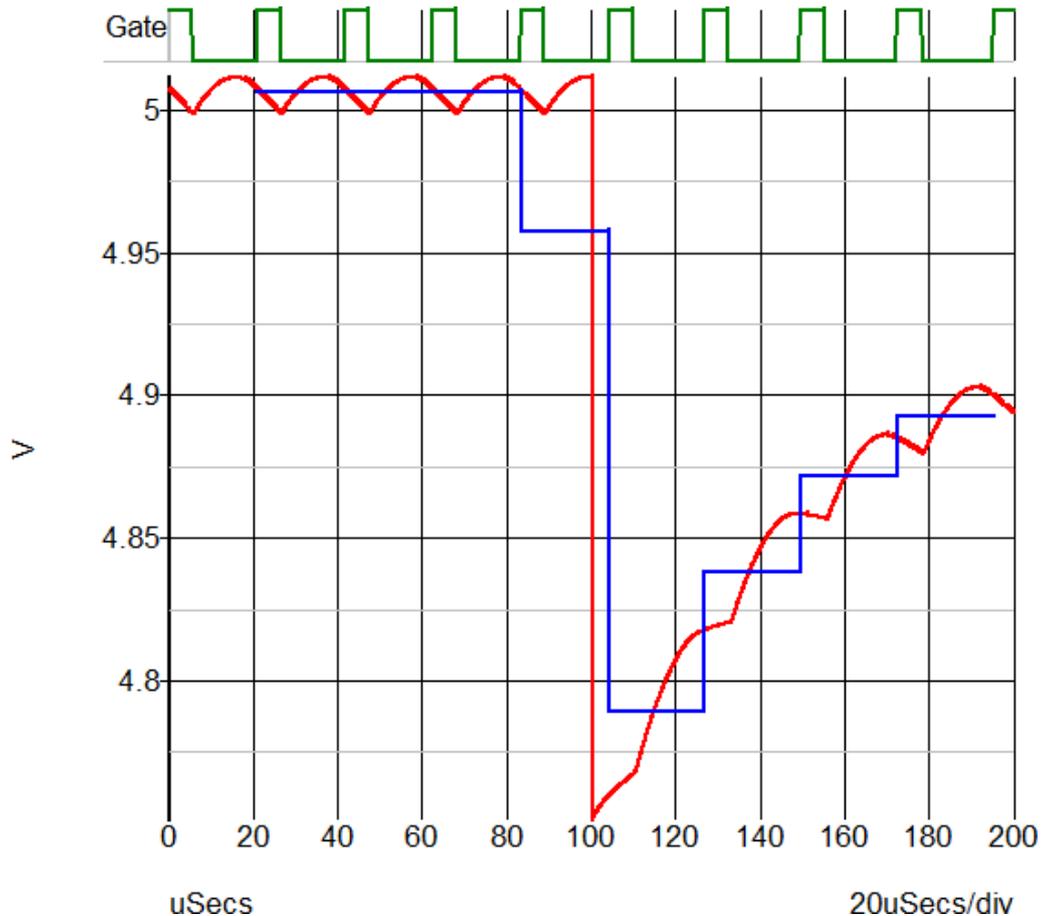


The following timing measurements are supported on a per-cycle basis:

- Frequency
- Period
- Duty Cycle
- On Time
- Off Time

### Per-Cycle Amplitude Measurements

The per-cycle measurement can also be applied to amplitude of an input voltage or current. For example, you can plot the mean value of a switching power converter output with the mean value being calculated on a per-cycle basis. This example is show below - the mean value of a converter output voltage is plotted in a per-cycle manner. The gate voltage is used to determine the timing edge information.



The following amplitude measurements are supported:

- Mean
- Maximum
- Minimum
- RMS
- Peak to Peak

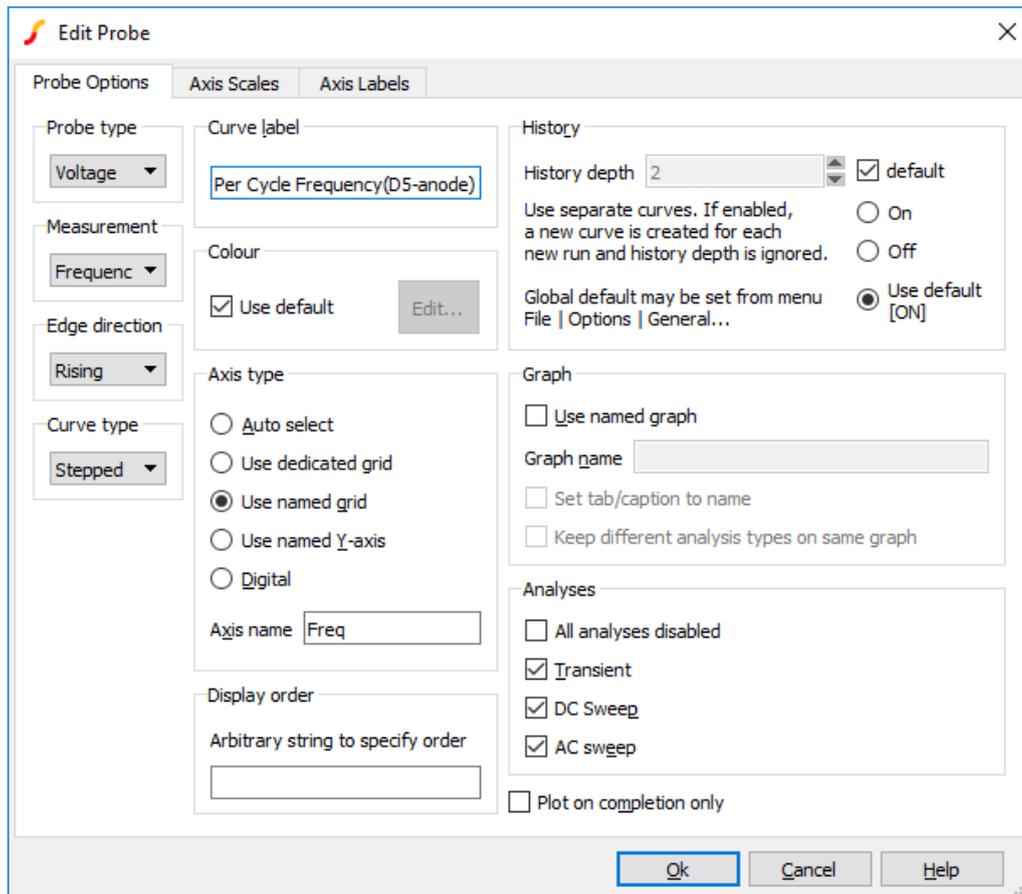
### Editing the Per Cycle Probe

To configure the Per Cycle Probe, follow these steps:

1. Double click the symbol on the schematic to open the editing dialog to the Probe Options tab.
2. Make the appropriate changes on the three tabs as explained below.

### Per Cycle Probe Options

The following tab allows you to define general probe options, which are explained in the table below.



Parameter	Description
Measurement	Frequency, Period, On-Time, Off-Time, Duty Cycle, Mean Maximum, Minimum, Peak-to-Peak, or RMS
Edge Direction	Rising, Falling or Both
Curve type	Stepped or Smooth
Curve Label	Text displayed by the probe on the schematic and also used to label the resulting curves

## History

Controls the number of curves from previous runs that will be displayed. If set to 1, only the latest result will be shown. If set to a value greater than one, the current result and a number of previous results equal to the history depth less one will be shown. The old results will be shown in broken lines.

If **default** is checked, the number of curves will be based on the values specified in the general options.

To set the history depth value in general options, follow these steps:

1. From the menu bar, select **File | Options | General...**, and then select the Graph/Probe/Data Analysis tab
2. In the Fixed probe global options section, enter a value in the Default history depth field, and then click Ok

If **Separate curves** is checked the history depth setting is ignored. In this case a new curve is created for each new run.

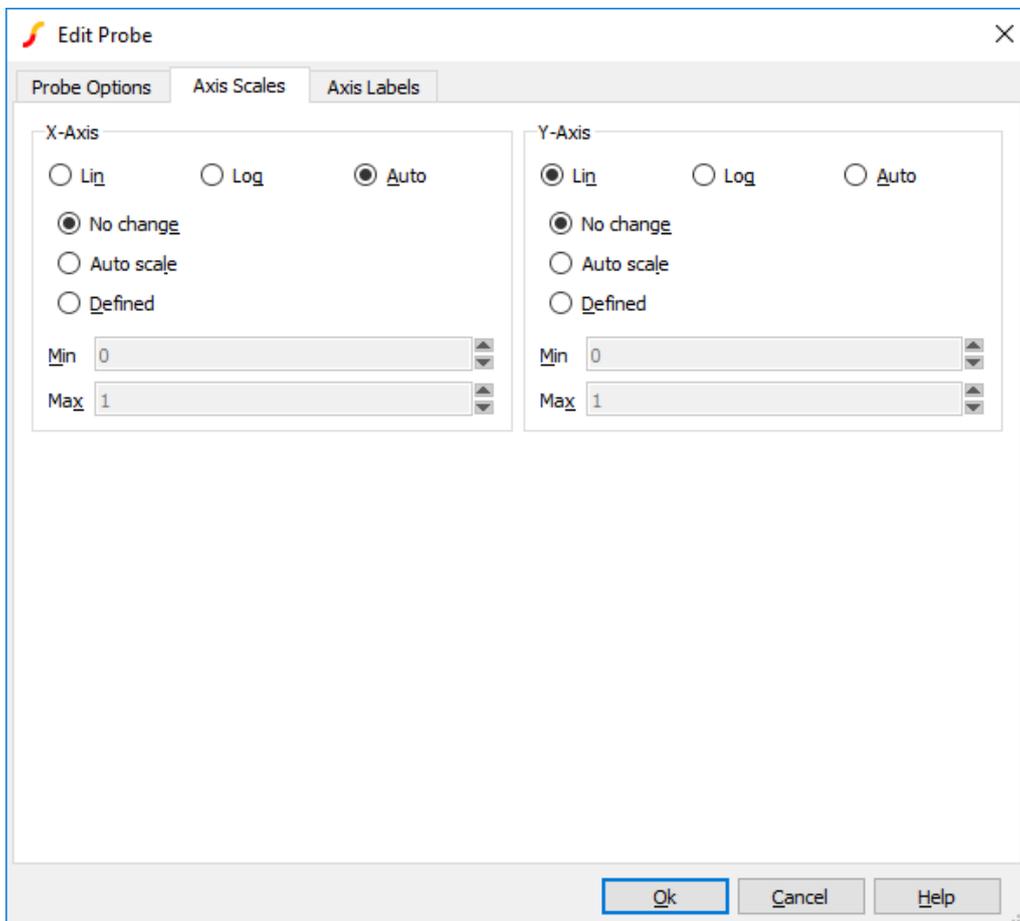
See [“Curve History” on page 294](#) for more information

Axis type	<p>Specifies how the grid and y-axis are chosen for the plotted data.</p> <p><b>Auto Select</b>      Chooses the most suitable grid and y-axis based on the physical units of the plotted data. So voltages will be plotted on the same y-axis as other voltages and currents will be plotted on the same y-axis as other currents. Separate grids will be used if the x-axis are incompatible. For example, if an AC analysis and transient analysis are performed in the same run and the same graph sheet is forced (See <a href="#">Use named graph</a> below) the results for the two analyses will automatically be placed on independent grids.</p> <p style="padding-left: 100px;">If the signal is digital, a digital axis (see below) will be used for this probe.</p> <p><b>Use dedicated grid</b>      The results for this probe will be placed on its own grid which will not be shared with any other probe</p> <p><b>Use named grid</b>      Similar to above but grid will be shared with other probes that specify the same grid name. See diagram in section <a href="#">“Graph Layout - Multiple Y-Axis Graphs”</a> on page 341.</p> <p><b>Use named Y-axis</b>      Uses a separate y-axis. The y-axis may be shared with other probes that specify the same axis name. This name is not used as a label for display purposes but simply as a means of identification.</p> <p style="padding-left: 100px;">Axes can be labelled using the <a href="#">Axis Labels</a> sheet as explained below.</p> <p><b>Digital</b>      Use a digital axis. Digital axes are placed at the top of the window and are stacked. Each one may take only a single curve. As the name suggests, these are intended for digital traces but can be used for analog signals if required.</p>
Graph	<p>Check <b>Use Separate Graph</b> to create a new graph sheet for the probe. You may also supply a graph name, in which case other probes using the same name will be directed to the same graph. The graph name may also be used to force curves to the same graph when the default action would be to create a new graph. This will happen for example for a new analysis type. The results from a Transient analysis, would usually be placed on a different graph sheet to the results from an AC analysis. Specifying this option with a name will override this.</p> <p>Check the <b>Set tab/caption to name</b> box to show the graph name in the caption bar and tab of the associated graph sheet.</p>
Analyses	<p>Specifies for which analyses the probe is enabled.</p> <p>Note: Other analysis modes such as noise and sensitivity are not included because these do not support schematic cross probing of current or voltage. For SIMPLIS mode, a Periodic Operating Point (POP) option is also available.</p>

Display order	Enter a string to control the grid display order. The value is arbitrary and will not be displayed. To force the curve to be placed above other curves that do not use this value, prefix the name with '!'. The '!' character has a low ASCII value. Conversely, use ' ' to force curve to be displayed after other curves.
Colour	Check Use default to minimise duplicate colours on the same graph by allowing the colour to be chosen automatically. Alternatively uncheck, this box and then press Edit... to select a colour of your choice. In this case the trace always has the same colour.
Plot on completion only	This option is not available with per-cycle probes and the setting of the check box will be ignored.

## Axis Scales

The following tab allows you to define the scale for the X-axis and for the Y-axis as explained in the table below.



Parameter	Description
-----------	-------------

- |                                    |  |
|------------------------------------|--|
| Lin/Log/Auto                       | Specify whether you want X-Axis to be linear or logarithmic. <ul style="list-style-type: none"> <li>• If you select Auto, the axis (X or Y) will be set to log if the x values are logarithmically spaced.</li> <li>• For the Y-axis, it is also necessary that the curve values are positive for a log axis to be selected.</li> </ul>  |
| No Change<br>Auto scale<br>Defined | Controls how the axis limits are defined <ul style="list-style-type: none"> <li>• If you select No change, the axis scales remain as they are; this is relevant only if you are adding to an existing graph.</li> <li>• If you select Auto scale, the axis scales are based on the minimum and maximum values of the curve.</li> <li>• If you select Defined, enter values in the Min and Max fields.</li> </ul> |

## Axis Labels

To specify axis labels and units, click the **Axis Labels** tab, and enter values as needed.

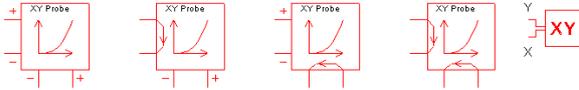
**Note:** If any box is left blank, a default value is used or remains unchanged if the axis already has a defined label.

The screenshot shows the 'Edit Probe' dialog box with the 'Axis Labels' tab selected. The dialog contains the following elements:

- Probe Options** | **Axis Scales** | **Axis Labels**
- X-Axis**
  - Axis label:
  - Axis units:
- Y-Axis**
  - Axis label:
  - Axis units:
- Leave boxes blank to specify default
- Buttons: **Ok**, **Cancel**, **Help**

### 13.3.7 XY Probes

XY Probes plot two curves against each other in an XY plot. Each input can be either differential voltage or inline current, and the curves can be output to any graph and axis. XY probes are available in both SIMetrix and SIMPLIS. The XY Probe specification is shown in the following table:

XY Probes	
Model Name:	XY Probe
Simulator:	This device is compatible with the SIMetrix and SIMPLIS simulators.
Part Selector Location:	<b>Probes   XY Probes</b>
Symbol Library:	connection.sxslb
Symbols:	
Multiple Selections:	Multiple probes can be selected and edited simultaneously.

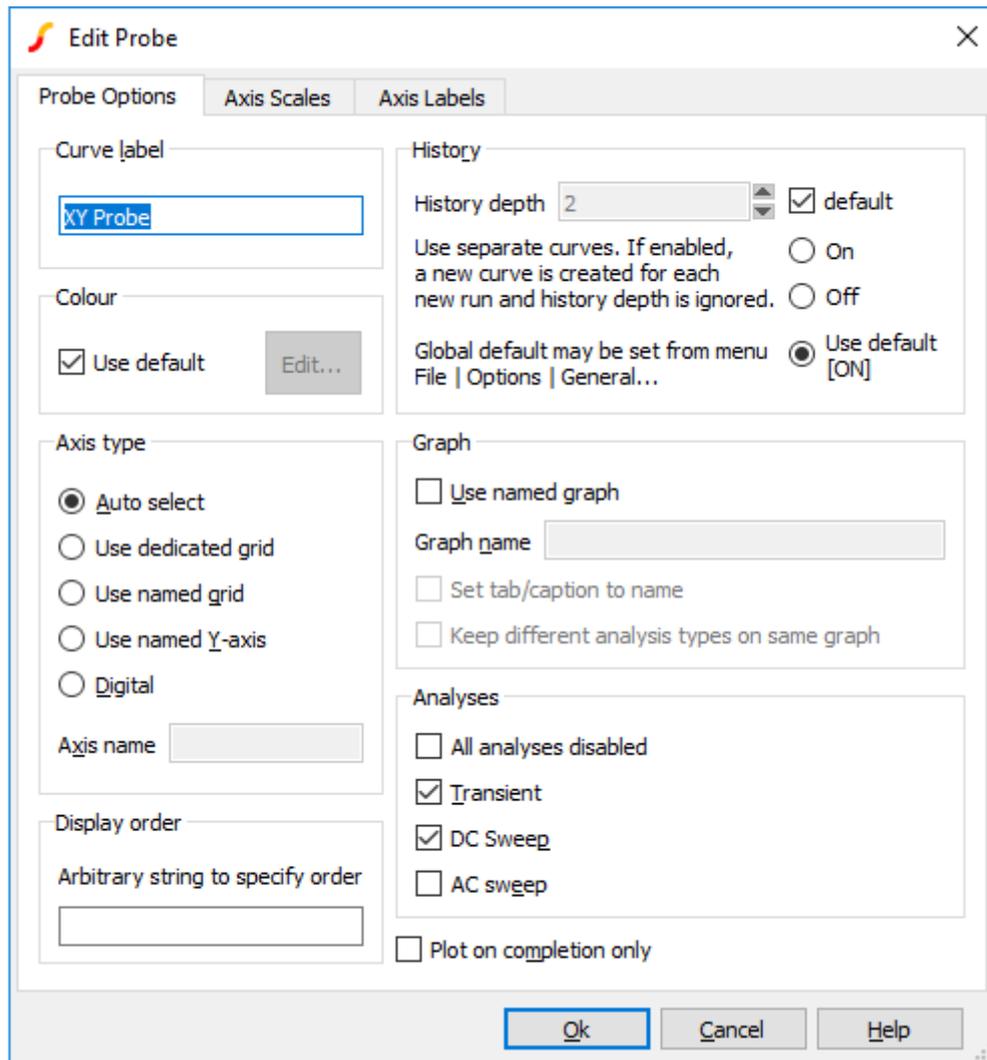
### Editing the XY Probe

To configure the XY Probe, follow these steps:

1. Double click the symbol on the schematic to open the editing dialog to the Probe Options tab.
2. Make the appropriate changes on the three tabs as explained below.

### XY Probe Options

The following tab allows you to define general probe options, which are explained in the table below.



**Curve Label**      Text displayed by the probe on the schematic and also used to label the resulting curves

## History

Controls the number of curves from previous runs that will be displayed. If set to 1, only the latest result will be shown. If set to a value greater than one, the current result and a number of previous results equal to the history depth less one will be shown. The old results will be shown in broken lines.

If **default** is checked, the number of curves will be based on the values specified in the general options.

To set the history depth value in general options, follow these steps:

1. From the menu bar, select **File | Options | General...**, and then select the Graph/Probe/Data Analysis tab
2. In the Fixed probe global options section, enter a value in the Default history depth field, and then click Ok

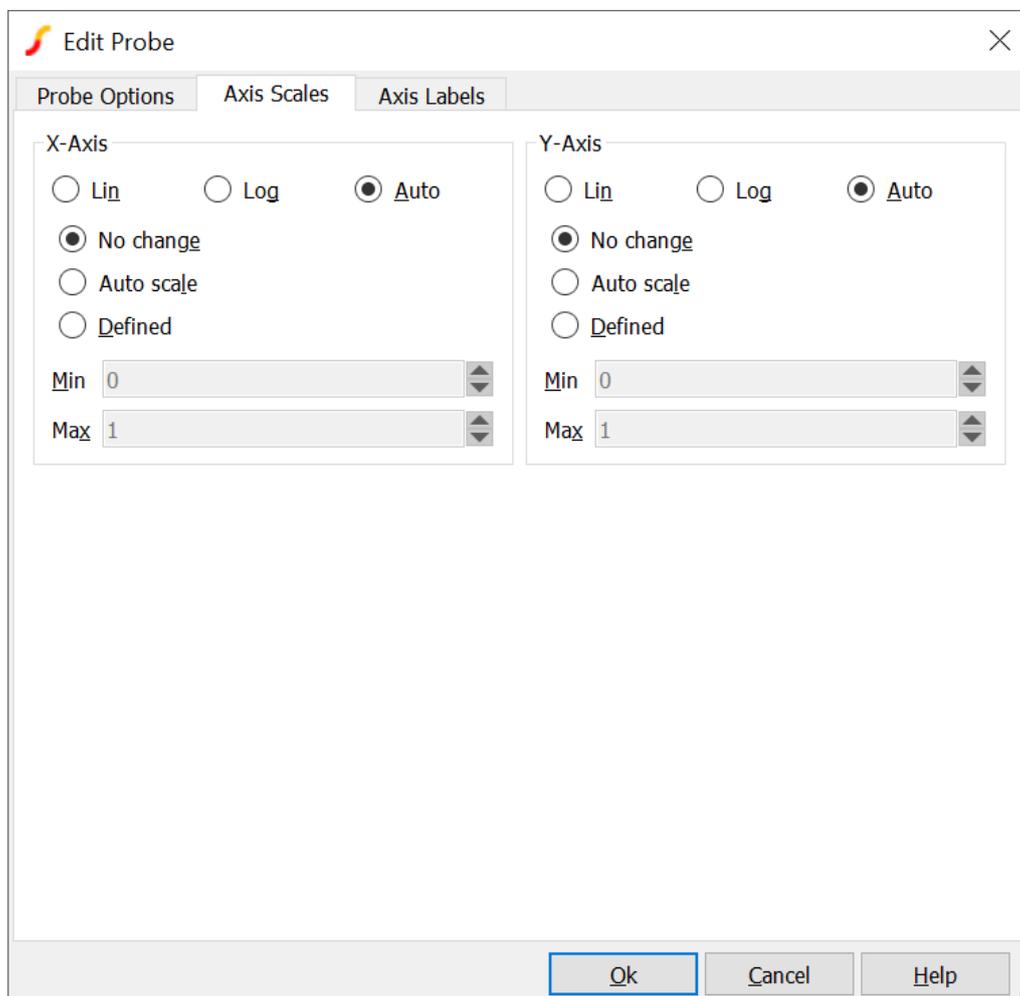
If **Separate curves** is checked the history depth setting is ignored. In this case a new curve is created for each new run.

See [“Curve History” on page 294](#) for more information

Axis type	<p>Specifies how the grid and y-axis are chosen for the plotted data.</p> <p><b>Auto Select</b>      Chooses the most suitable grid and y-axis based on the physical units of the plotted data. So voltages will be plotted on the same y-axis as other voltages and currents will be plotted on the same y-axis as other currents. Separate grids will be used if the x-axis are incompatible. For example, if an AC analysis and transient analysis are performed in the same run and the same graph sheet is forced (See <a href="#">Use named graph</a> below) the results for the two analyses will automatically be placed on independent grids.</p> <p style="padding-left: 100px;">If the signal is digital, a digital axis (see below) will be used for this probe.</p> <p><b>Use dedicated grid</b>      The results for this probe will be placed on its own grid which will not be shared with any other probe</p> <p><b>Use named grid</b>      Similar to above but grid will be shared with other probes that specify the same grid name. See diagram in section <a href="#">“Graph Layout - Multiple Y-Axis Graphs”</a> on page 341.</p> <p><b>Use named Y-axis</b>      Uses a separate y-axis. The y-axis may be shared with other probes that specify the same axis name. This name is not used as a label for display purposes but simply as a means of identification.</p> <p style="padding-left: 100px;">Axes can be labelled using the <a href="#">Axis Labels</a> sheet as explained below.</p> <p><b>Digital</b>      Use a digital axis. Digital axes are placed at the top of the window and are stacked. Each one may take only a single curve. As the name suggests, these are intended for digital traces but can be used for analog signals if required.</p>
Graph	<p>Check <b>Use Separate Graph</b> to create a new graph sheet for the probe. You may also supply a graph name, in which case other probes using the same name will be directed to the same graph. The graph name may also be used to force curves to the same graph when the default action would be to create a new graph. This will happen for example for a new analysis type. The results from a Transient analysis, would usually be placed on a different graph sheet to the results from an AC analysis. Specifying this option with a name will override this.</p> <p>Check the <b>Set tab/caption to name</b> box to show the graph name in the caption bar and tab of the associated graph sheet.</p>
Analyses	<p>Specifies for which analyses the probe is enabled.</p> <p>Note: Other analysis modes such as noise and sensitivity are not included because these do not support schematic cross probing of current or voltage. For SIMPLIS mode, a Periodic Operating Point (POP) option is also available.</p>

Display order	Enter a string to control the grid display order. The value is arbitrary and will not be displayed. To force the curve to be placed above other curves that don't use this value, prefix the name with '!'. The '!' character has a low ASCII value. Conversely, use ' ' to force curve to be displayed after other curves.
Colour	Check <b>Use default</b> to minimise duplicate colours on the same graph by allowing the colour to be chosen automatically. Alternatively uncheck, this box and then press <b>Edit...</b> to select a colour of your choice. In this case the trace always has the same colour.
Plot on completion only	This option is not available with XY Probes and the check box setting will be ignored.

## Axis Scales



Lin/Log/Auto

Specify whether you want X-Axis to be linear or logarithmic.

- If you select Auto, the axis (X or Y) will be set to log if the x values are logarithmically spaced.
- For the Y-axis, it is also necessary that the curve values are positive for a log axis to be selected.

No Change  
Auto scale  
Defined

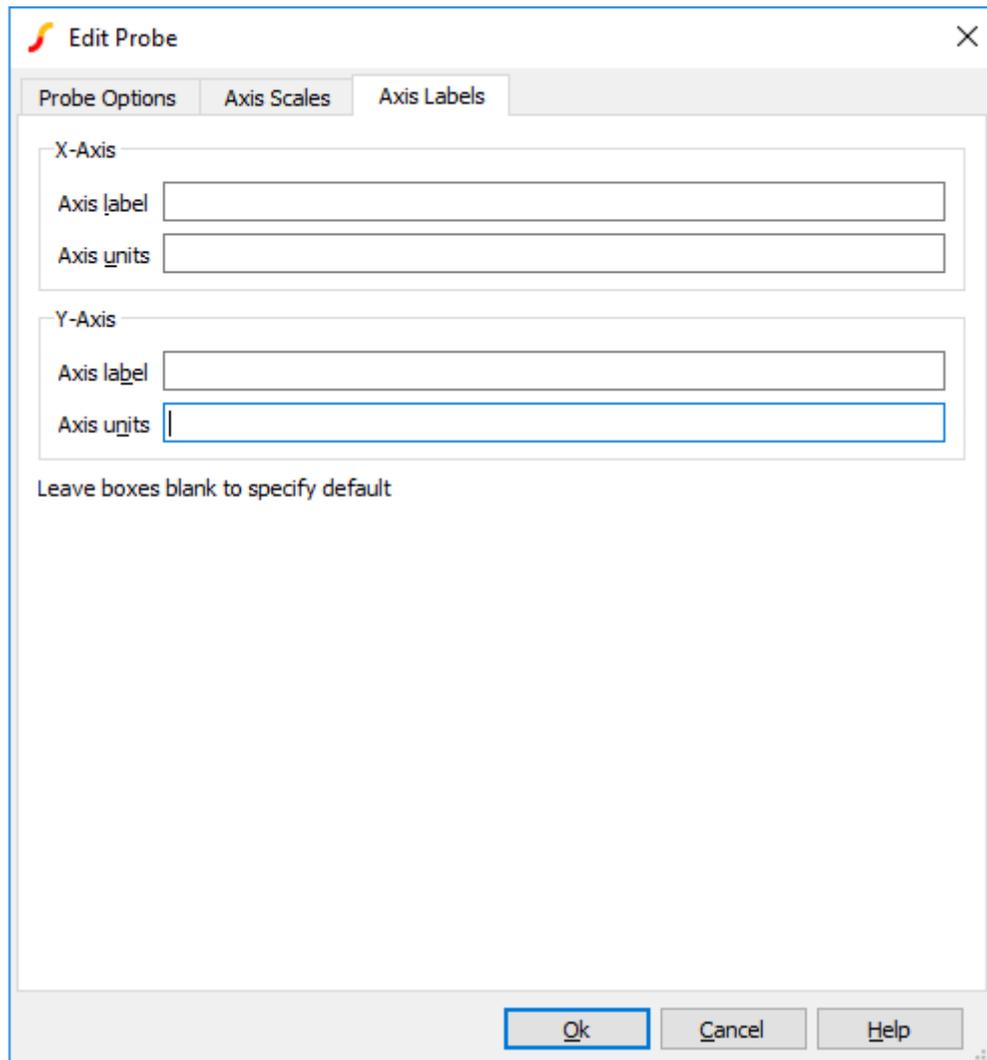
Controls how the axis limits are defined

- If you select No change, the axis scales remain as they are; this is relevant only if you are adding to an existing graph.
- If you select Auto scale, the axis scales are based on the minimum and maximum values of the curve.
- If you select Defined, enter values in the Min and Max fields.

## Axis Labels

To specify axis labels and units, click the **Axis Labels** tab, and enter values as needed.

**Note:** If any box is left blank, a default value is used or remains unchanged if the axis already has a defined label.



### 13.3.8 Bode Plot Probe with Measurements

The Bode Plot Probe with Measurements generates plots for the gain and phase of the ratio of two voltages. The probe can be configured to plot only the gain or phase, or both gain and phase.

Summary - Bode Plot Probe with Measurements	
Model Name	Bode Plot Probe
Simulator	This device is compatible with both SIMetrix and SIMPLIS simulators
Schematic Menu	<b>Probe AC/Noise   Bode Plot Probe - with Measurements</b>
Parts Selector	<b>Probes   Bode Plot Probe - with Measurements</b>
Symbol Library	Connections
Model File	none
Symbol	

---

**Summary - Bode Plot Probe with Measurements**


---

Multiple Selections	If multiple Bode plot probes are selected before editing, all properties except the curve labels can be simultaneously changed for all probes. The curve label properties will remain unchanged for all selected probes
Usage	This schematic probe symbol plots the magnitude and phase of the ratio of two complex voltages, OUT/IN. The magnitude can be plotted in db or volts/volts; the horizontal axis scale is determined by the simulation sweep type. If the simulation is swept in log frequency steps, the horizontal axis will automatically be log scaled

## Editing the Bode Plot Probe with Measurements

To configure the Bode Plot Probe with Measurements, follow these steps:

1. Double click the symbol on the schematic to open the editing dialog to the Parameters tab.
2. Make the appropriate changes to the fields described in the table below the image.

---

**Options - Bode Plot Probe with Measurements**


---

Use separate graph	Check this option in order to supply an alias for each output graph tab
--------------------	---

<b>Options - Bode Plot Probe with Measurements</b>	
Graph name	Curves with the same graph name property will be output on the same graph tab. Graph name be safely ignored unless multiple Bode plot probes are used. Use separate graph must be checked to supply this name. The name entered will be used as a label for the tab and the caption bar if the Set tab/caption to name box is checked
Disable gain/phase	Check this box to disable both gain and phase graphs
History	<p>Controls the number of curves from previous runs that will be displayed. If set to 1, only the latest result will be shown. If set to a value greater than one, the current result and a number of previous results equal to the history depth less one will be shown. The old results will be shown in broken lines.</p> <p>If <b>default</b> is checked, history depth will be based on the values specified in the general options.</p> <p>To set the history depth value in general options, follow these steps:</p> <ol style="list-style-type: none"> <li>1. From the menu bar, select <b>File   Options   General...</b>, and then select the Graph/Probe/Data Analysis tab</li> <li>2. In the Fixed probe global options section, enter a value in the Default history depth field, and then click Ok</li> </ol> <p>If <b>Separate curves</b> is checked the history depth setting is ignored. In this case a new curve is created for each new run.</p> <p>See <a href="#">“Curve History” on page 294</a> for more information</p>
Curve label	Sets the name of the curve. <b>Note:</b> This field appears in both the Gain and Phase groups on the dialog
Y axis label	Sets the Y axis label for the individual gain and phase axes. When multiple curves from multiple probes are placed on the same axis, the axis label properties must be identical or the axis label is blank. If this field is left blank, the axis label appears with the name specified in the Curve label field. Note: This field appears in both the Gain and Phase groups on the dialog
Vertical scale	<p>Allows you to select the function to perform on the simulation data.</p> <ul style="list-style-type: none"> <li>• Select <b>db</b> to perform the built-in db() vector function.</li> <li>• Select <b>Linear</b> to perform the mag() function</li> </ul>
Curve	<p>Selects the Phase curve offset</p> <ul style="list-style-type: none"> <li>• Select <b>Phase</b> for no offset.</li> <li>• Select <b>Phase - 180 degrees</b> to subtract 180 degrees before plotting the curve.</li> </ul>

<b>Options - Bode Plot Probe with Measurements</b>	
Vertical axis	<p>This group has two options: <b>Vertical limits</b></p> <ul style="list-style-type: none"> <li>• Check to have the vertical limits automatically set to fit the curve data.</li> <li>• Uncheck to enter minimum and maximum values for the limits.</li> </ul> <p>Both minimum and maximum axis limits must be set or auto-calculated as a group. <b>Grid spacing:</b></p> <ul style="list-style-type: none"> <li>• Check to auto calculate the grid spacing.</li> <li>• Uncheck to set the grid spacing independent of the axis limits</li> </ul> <p>Note: This field appears in both the <b>Gain</b> and <b>Phase</b> groups on the dialog.</p>
Colour	<p>Defines colours for the curves:</p> <ul style="list-style-type: none"> <li>• Check <b>Use default</b> in both the <b>Gain</b> and <b>Phase</b> groups to generate the curves with the same default colour, which is automatically selected from a sequence of colours</li> <li>• Leave <b>Use default</b> unchecked and click <b>Edit...</b> to open a colour selector dialog box to specify fixed colours for the curves</li> </ul> <p>The change the default sequence of colours from the command shell menu, select <b>File   Options   Colour</b> For additional information see <a href="#">“Colours and Fonts” on page 472</a>. Note: This field appears in both the <b>Gain</b> and <b>Phase</b> groups on the dialog.</p>
Display curves on	Allows you to define the number of grids in the output: <b>Single grid</b> or <b>Two grids</b>
Vertical order	<p>Allows you to specify the order of the curves: <b>Phase above Gain</b> or <b>Gain above Phase</b>.</p> <ul style="list-style-type: none"> <li>• If you selected <b>Two grids</b> is in the <b>Display curves on</b> group, the selection in this group specifies the curve output location</li> <li>• If you selected <b>Single grid</b>, both curves are on the lower grid with two axes</li> </ul>
Example curve output	<p>Example curve output illustrates the relative locations of the two curves. Note: The curve data here is fixed; these curves are examples and do not reflect the simulated curves</p>
Save Configuration	Click <b>Save Configuration</b> to preserve this information as the default configuration for all future new Bode plot probes

### 13.3.9 Bode Plot Probe - Basic Version

Connects to the input and output of a circuit to plot its gain and phase. A more sophisticated bode plot probe which provides a wide range of options along with the display of useful measurements is also available and is recommended for most applications. See [“Bode Plot Probe with Measurements” on page 310](#)

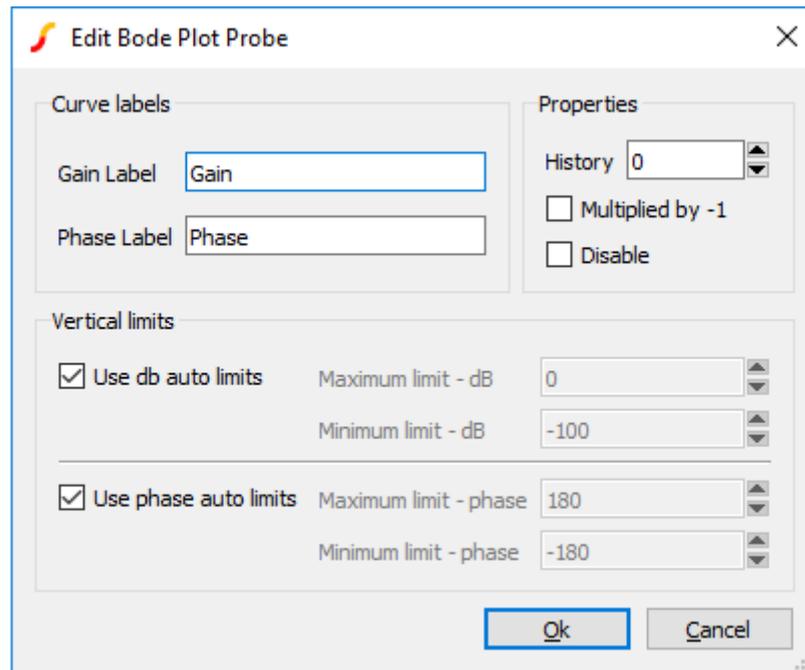
This basic version is useful for quick checks and for compatibility with versions 7.0 or earlier.

Summary - Bode Plot Probe - Basic	
Model Name	Bode Plot Probe - Basic
Simulator	This device is compatible with both SIMetrix and SIMPLIS simulators
Schematic Menu	<b>Probe AC/Noise   Bode Plot Probe - Basic</b>
Parts Selector	<b>Probes   Bode Plot - Gain/Phase</b>
Symbol Library	Connections
Model File	none
Symbol	
Multiple Selections	If multiple Bode plot probes are selected before editing, all properties except the curve labels can be simultaneously changed for all probes. The curve label properties will remain unchanged for all selected probes
Usage	This schematic probe symbol plots the magnitude and phase of the ratio of two complex voltages, OUT/IN. The magnitude can be plotted in db or volts/volts; the horizontal axis scale is determined by the simulation sweep type. If the simulation is swept in log frequency steps, the horizontal axis will automatically be log scaled

## Editing Bode Plot Probe

To configure the Bode Plot Probe, follow these steps:

1. Double click the symbol on the schematic to open the editing dialog to the Parameters tab.
2. Make the appropriate changes to the fields described in the table below the image.




---

### Options - Bode Plot Probe

---

Curve labels	<p>Sets labels for plots</p> <ul style="list-style-type: none"> <li>• Gain label: Label used for gain plot</li> <li>• Phase label: Label used for phase plot</li> </ul>
Properties	<p>Set additional properties on probe:</p> <ul style="list-style-type: none"> <li>• History: Controls how many plots are retained. 0 means use global default setting</li> <li>• Multiplied by -1: Inverts the sense of the phase</li> <li>• Disable: Disable probe. If checked no plots will be created</li> </ul>
Vertical Limits	<p>Divided into two parts to allow setting of axis limits for phase and gain plots.</p> <ul style="list-style-type: none"> <li>• Use db auto limits: If checked the gain plot will automatically scale. Otherwise manual limits can be entered in Maximum limit - dB and Minimum limit - dB</li> <li>• Use phase auto limits: If checked the phase plot will automatically scale. Otherwise manual limits can be entered in Maximum limit - phase and Minimum limit - phase</li> </ul>

### 13.3.10 Fourier Analysis

A fixed Fourier probe is available which will perform a spectral analysis on a node voltage. To place this probe, select menu **Probes | Place Fourier Voltage Probe**.

Double click the probe to edit it. You will see this dialog box:

The settings are similar to that for random Fourier probe plotting as described here: [“Fourier Analysis” on page 325](#). The documentation is repeated here for convenience.

#### Method

SIMetrix offers two alternative methods to calculate the Fourier spectrum: *FFT* and *Continuous Fourier*.

The simple rule is: use FFT unless the signal being examined has very large high frequency components as would be the case for narrow sharp pulses.

A description of the two techniques and their pros and cons follows.

FFT	<p>Fast Fourier Transform. This is an efficient algorithm for calculating a discrete Fourier transform or DFT. DFTs generally operate on evenly spaced sampled data. Unfortunately the data generated by the simulator is not evenly spaced so it is therefore necessary to interpolate the data before presenting it to an FFT algorithm. The interpolation process is in effect the sampling process and the Nyquist sampling theorem applies. This states that the signal can be perfectly reproduced from the sampled data if the sampling rate is greater than twice the maximum frequency component in the signal. In practice this condition can never be met perfectly and any signal components whose frequency is greater than half the sampling rate will be aliased to a different frequency.</p> <p>So if the number of interpolated points is too small there will be errors in the result due to high frequency components being aliased to lower frequencies. This is the Achilles heel of FFTs applied to simulated data.</p> <p>The <i>Continuous Fourier</i> technique, described next, does not suffer from this problem. It suffers from other problems the main one being that it is considerably slower than the FFT.</p>
Continuous Fourier	<p>This calculates the Fourier spectrum by numerically integrating the Fourier integral. With this method, each frequency component is calculated individually whereas with the FFT the whole spectrum is calculated in one - quite efficient - operation. Continuous Fourier does not require the data to be interpolated and does not suffer from aliasing.</p> <p>The problem with continuous Fourier is that compared to the FFT it is a slow algorithm and in many cases an FFT with a very large number of interpolated points can be calculated more quickly and give just as accurate a result.</p> <p>However, in cases where a signal has a very large high frequency content - such as narrow pulses - this method is superior and it is recommended that it is used in preference to the FFT in such situations. The continuous Fourier technique has the additional advantage that it can be applied with greater confidence as the aliasing errors will not be present. It does have its own source of error due to the fact that simulated data itself is not truly continuous but represented by unevenly spaced points with no information about what lies between the points. This error can be minimised by ensuring that close simulation tolerances are used. See <i>Simulator Reference Manual/Convergence, Accuracy and Performance</i> for details.</p>

## Plot (Phase or Magnitude)

The default is to plot the magnitude of the Fourier spectrum. Select **Phase** if you require a plot of phase or **dB** if you need the magnitude in dBs.

## Frequency Display

Resolution/Hz	Available only for the continuous Fourier method. This is the frequency interval at which the spectral components are evaluated. It cannot be less than $1/T$ where $T$ is the time interval over which the spectrum is calculated.
---------------	---

Start Freq./Hz	Start frequency of the display.
Stop Freq./Hz	Stop frequency of the display.
Log X-Axis	Check this to specify a logarithmic x-axis. This will force a minimum value for the start frequency equal to $1/T$ where T is the time interval being analysed.

## Signal Info

If the signal being analysed is repetitive and the frequency of that signal is known *exactly* then a much better result can be obtained if it is specified here. Check the **Know fundamental frequency** box then enter the frequency. The Fourier spectrum will be calculated using an integral number of complete cycles of the fundamental frequency. This substantially reduces *spectral leakage*. Spectral leakage occurs because both the Fourier algorithms work on an assumption that the signal being analysed is a repetition of the analysed time interval from  $t = -\infty$  to  $t = +\infty$ . If the analysed time interval does not contain a whole number of cycles of the fundamental frequency this will be a poor approximation and the spectrum will be in error. In practice this problem is minimised by using a *window* function applied to the signal prior to the Fourier calculation, but using a whole number of cycles reduces the problem further.

Note that the fundamental frequency is not necessarily the lowest frequency in the circuit but the largest frequency for which all frequencies in the circuit are integral harmonics. For example if you had two sine wave generators of 1kHz and 1.1KHz, the fundamental is 100Hz, not 1kHz; 1kHz is the tenth harmonic, 1.1KHz is the eleventh.

You should *not* specify a fundamental frequency for circuits that have self-oscillating elements.

## FFT Interpolation

As explained above, the FFT method must interpolate the signal prior to the FFT computation. Specify here the number of points and the order. The number of points entry may be forced to a minimum if a high stop frequency is specified in the **Frequency Display** section.

The number of interpolation points required depends on the highest significant frequency component in the signal being analysed. If you have an idea what this is, a useful trick to set the number of points to a suitable value, is to increase the stop frequency value in the **Frequency Display** section up to that frequency. This will automatically set the number of interpolation points to the required value to handle that frequency. If you don't actually want to display frequencies up to that level, you can bring the stop frequency back down again. The number of interpolation points will stay at the value reached.

If in doubt, plot the FFT twice using a different number of points. If the two results are significantly different in the frequency band of interest, then you should increase the number of points further.

Usually an interpolation order of 2 is a suitable value but you should reduce this to 1 if analysing signals with abrupt edges. If analysing a smooth signal such as a sinusoid, useful improvements can be gained by increasing the order to 3.

## Data Span

Usually the entire simulated time span is used for the Fourier analysis. To specify a smaller time interval click **Specify** and enter the start and end times.

Note that if you specify a fundamental frequency, the time may be modified so that a whole number of cycles is used. This will occur whether or not you explicitly specify an interval.

## Window

A window function is applied to the time domain signal to minimise spectral leakage (See above).

The choice of window is a compromise. The trade off is between the bandwidth of the main spectral component or lobe and the amplitude of the side-lobes. The rectangular window - which is in effect no window - has the narrowest main lobe but substantial side-lobes. The Blackman window has the widest main lobe and the smallest side lobes. Hanning and Hamming are something in between and have similar main lobe widths but the side lobes differ in the way they fall away further from the main lobe. Hamming starts smaller but doesn't decay whereas Hanning while starting off larger than Hamming, decays as the frequency moves away from the central lobe.

Despite the great deal of research that has been completed on window functions, for many applications the difference between Hanning, Hamming and Blackman is not important and usually Hanning is a good compromise.

There are situations where a rectangular window can give significantly superior results. This requires that the fundamental frequency is specified and also that the simulated signal is consistent over a large number of cycles. The rectangular window, however, *usually* gives considerably *poorer* results and must be used with caution.

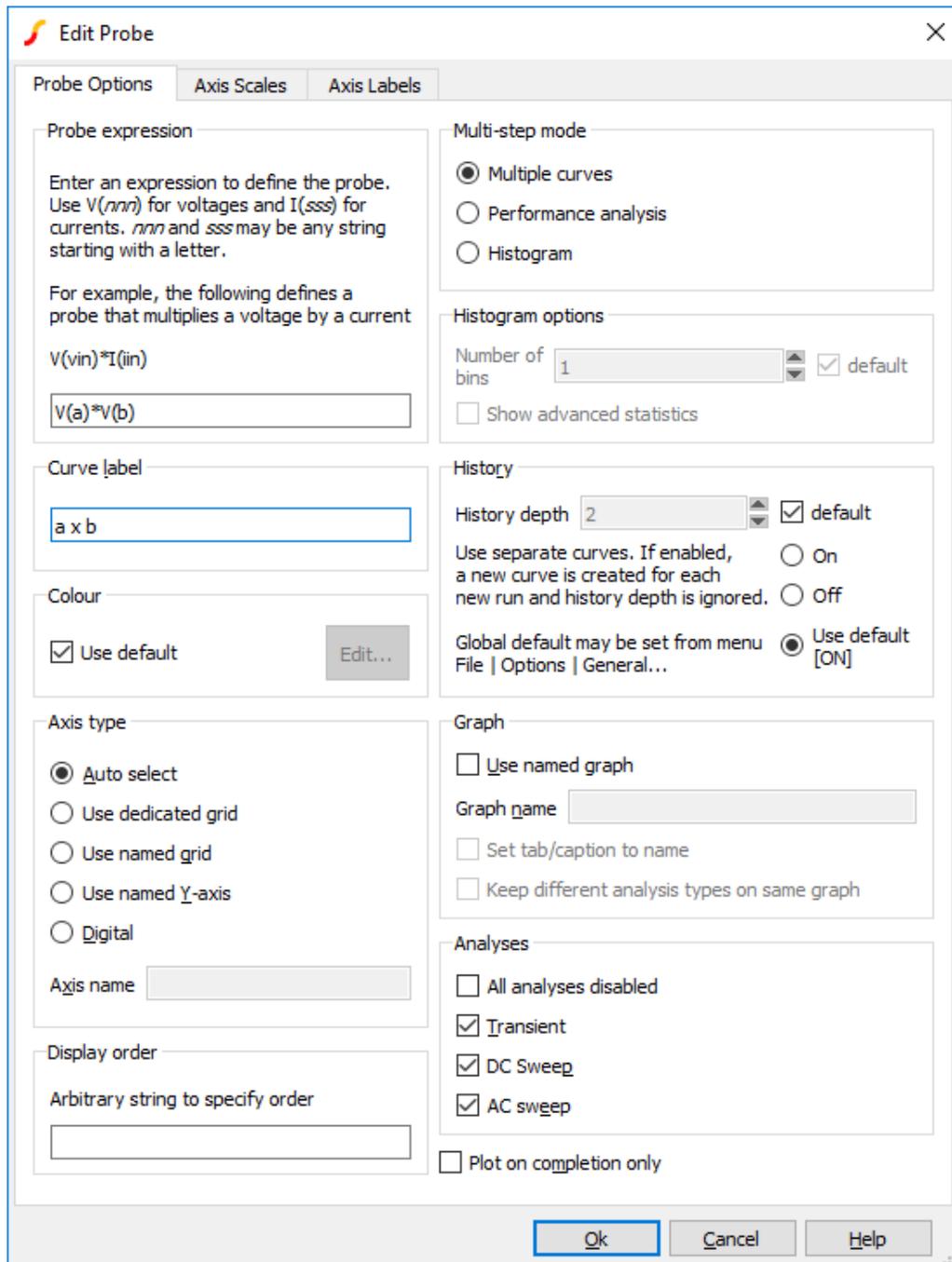
## Run time calculation time limit

If using the Continuous Fourier method, you can enter a maximum time limit to calculate the Fourier spectrum. If the limit is exceeded, the calculation will abort and no display will be made. The Continuous Fourier calculation time can be excessive but cannot be predicted before the simulation is started as it is necessary to know the number of simulation time points to be processed.

For the FFT method, the simulation time can be estimated to a reasonable accuracy before the simulation and this estimate will be displayed labelled Estimated calculation time:.

### 13.3.11 Arbitrary Fixed Probe

You can create an arbitrary fixed probe to plot an expression of any combination of node voltages and device currents. To use this feature, select menu **Probe | Create and Place Arbitrary Probe**. You will see this dialog box:

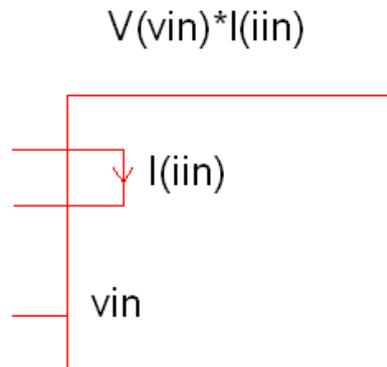


## Probe expression

Enter an expression to define what you wish to be plotted. Use  $V(nnn)$  to access a voltage and  $I(sss)$  to access a current.  $nnn$  and  $sss$  may be any arbitrary string starting with a letter. When you close the dialog box a symbol will be created which will reflect what you enter here. It will have single inputs named according to occurrences of  $V(nnn)$  and pairs of inputs named after occurrences of  $I(sss)$ . For example if you enter the expression:

$$V(vin) * I(iin)$$

The symbol created will look like this:



The result plotted will be the product of the voltage on vin and the current in iin.

The same behaviour could be achieved using a Non-linear transfer function device (see [“Non-linear Transfer Function” on page 188](#) ) and a simple single ended probe. This arbitrary probe has some important advantages over that approach:

1. The arbitrary probe works in SIMPLIS simulations. The Non-linear transfer function device is a SIMetrix-only device.
2. It can evaluate non-linear functions in AC analysis
3. It is a post-processing operation and does not interfere with the simulation in any way

## Multi-step mode

The options in the Multi-step mode group control the behaviour of the probe with multi-step analyses.

1. **Multiple curves.** This is the normal mode and will create groups of curves.
2. **Performance analysis.** A performance analysis based on a goal function will be performed. In this mode a goal function must be entered for the probe expression. A single curve will be created of the value of the goal function against the swept variable.
3. **Histogram.** Used for Monte Carlo analyses. In this mode a goal function must be entered for the probe expression. A histogram plot will be created of the goal function results.

For more information about performance analysis and histograms, see [“Performance Analysis and Histograms” on page 374](#)

## Histogram options

Options for creating histograms. See above. Number of bins may be used to force a particular number of bins of the histogram. The default depends on the size of the sample and is typically half its square-root. A display of statistical calculations is displayed alongside the histogram including mean and standard deviation. The Show advanced statistics enables additional statistics such as skewness and kurtosis.

## Other Arbitrary Probe Options

The other options are the same as for standard fixed probes. Refer to [“Fixed Probe Options” on page 290](#)

### 13.3.12 Changing Update Period and Start Delay

The update period of all fixed probes can be changed from the Options dialog box. Select menu **File | Options | General...** and click on the Graph/Probe/Data analysis tab. In the Probe update times/seconds box there are two values that can be edited. **Period** is the update period and **Start** is the delay after the simulation begins before the curves are first created.

### 13.3.13 Auto Probe Colouring

Fixed probes on the schematic change colour to reflect the colour of the graph curves they create. So if a red curve is created for a particular probe, the text of that probe on the schematic will change to red. This eases associating probes with plotted curves.

This colouring feature is only active for curves that are actually visible. So any probes that generate curves that are not visible or which have been deleted, will show their text in the default colour - usually black on a white background.

If the curve is highlighted in the graph window, the colour in the schematic will also reflect the highlighting.

If a probe generates more than one visible curve, the text for the probe will be shown in italics. This is the case even if all the curves are actually the same colour.

## 13.4 Random Probes

### 13.4.1 General Behaviour

A wide range of functions are available from the schematic **Probe** and **Probe AC/Noise** menus. With a few exceptions detailed below, all random probe functions have the following behaviour.

- If there are no graph windows open, one will be created.
- If a graph window is open and the currently displayed sheet has a compatible x-axis to what you are probing, the new curve will be added to that sheet. E.g. if the currently displayed graph is from a transient analysis and has an x-axis of Time, and you are also probing the results of a transient analysis, then the new curve will be added to the displayed graph. If, however the displayed curve was from an AC analysis, its x-axis would be frequency which is incompatible. In this case a new graph sheet will be created for the new curve.

If you want to force a new graph sheet to be created, press **F10**. This will create an empty graph sheet.

The menus:

**Probe | Voltage (New graph sheet)...**

**Probe | Current (New graph sheet)...**

will always create a new graph.

### 13.4.2 Functions

The following table shows all available random probe functions. Many of these can be found in the schematic's **Probe** menu while others are only available from **Probe | More Probe Functions...**

---

#### Function

---

Single Ended Voltage

---

**Function**


---

Single Ended Voltage - AC coupled  
 Single Ended Voltage - dB  
 Single Ended Voltage - Phase  
 Single Ended Voltage - Fourier  
 Single Ended Voltage - Nyquist  
 Single Ended Voltage - Normalised dB  
 Single Ended Voltage - Group delay  
 Differential Voltage  
 Differential Voltage - dB  
 Differential Voltage - Phase  
 Differential Voltage - Fourier  
 Differential Voltage - Nyquist  
 Differential Voltage - Normalised dB  
 Differential Voltage - Group delay  
 Relative Voltage - dB  
 Relative Voltage - Phase  
 Relative Voltage - Nyquist  
 Relative Voltage - Normalised dB  
 Relative Voltage - Group delay  
 Single Ended Current - In device pin  
 Single Ended Current - AC coupled in device pin  
 Single Ended Current - In wire  
 Single Ended Current - dB  
 Single Ended Current - Phase  
 Single Ended Current - Fourier  
 Single Ended Current - Nyquist  
 Single Ended Current - Normalised dB  
 Single Ended Current - Group delay  
 Differential Current - Actual  
 Differential Current - dB  
 Differential Current - Phase  
 Differential Current - Fourier  
 Differential Current - Nyquist  
 Differential Current - Normalised dB  
 Differential Current - Group delay  
 Power  
 Impedance  
 Output noise (noise analysis only)  
 Input noise (noise analysis only)  
 Device noise (noise analysis only)

---

**Function**


---

Arbitrary expressions and XY plots

### 13.4.3 Notes on Probe Functions

#### Impedance

You may plot the AC impedance at a circuit node using **Probe | More Probe Functions...** This only works in AC analysis. This works by calculating  $V/I$  at the device pin selected.

#### Device Power

Device power is available from **Probe | Power In Device...** . This works by calculating the sum of VI products at each pin of the device. Power is not stored during the simulation. However, once you have plotted the power in a device once, the result is stored with the vector name:

*device\_name#pwr*

E.g. if you plot the power in a resistor R3, its power vector will be called R3#pwr. You can use this as part of an expression in any future plot.

Note that, because SIMetrix is able to find the current in a sub-circuit device or hierarchical block, it can also calculate such a device's power. Be aware, however, that as this power is calculated from the VI product of the device's pins, the calculation may be inaccurate if the sub-circuit uses global nodes.

### 13.4.4 Plotting Noise Analysis Results

Small signal noise analysis does not produce voltage and current values at nodes and in devices in the way that AC, DC and transient analyses do. Noise analysis calculates the overall noise at a single point and the contribution of every noisy device to that output noise. Optionally the input referred noise may also be available.

#### To Plot Output Noise

1. Select menu **Probe AC/Noise | Plot Output Noise**

#### To Plot Input Referred Noise

1. Select menu **Probe AC/Noise | Plot Input Noise**

Note that you must specify an input source for input referred noise to be available. See [“Noise Parameters” on page 242](#) for details.

#### To Plot Device Noise

1. Select menu **Probe AC/Noise | Probe Device Noise**
2. Click on device of interest

Note that noise results are only available for noisy devices such as resistors and semiconductor devices.

### 13.4.5 Plotting Transfer Function Analysis Results

No cross-probing is available with transfer function analysis. Instead, you must use the general purpose Define Curve dialog box. With this approach you must select a vector name from a list. Proceed as follows:

1. Select menu **Probe | Add Curve...**
2. Select a value from the Available Vectors drop down box.

### Transfer Function Vector Names

The vector names for transfer function will be of the form:

```
source_name#Vgain
source_name#Transconductance
source_name#Transresistance
source_name#Igain
```

where source\_name is the name of a voltage or current source.

The vectors Zout, Yout or Zin may also be available. These represent output impedance, output admittance and input impedance respectively.

For more information see the *Command Reference/.TF* chapter of the *Simulator Reference Manual*.

### 13.4.6 Fourier Analysis

A Fourier spectrum of a signal can be obtained in a number of ways. You have a choice of using the default settings for the calculation of the Fourier spectrum or you can customise the settings for each plot. The following menus use the default settings:

```
Probe | Fourier | Probe Voltage Quick...
Probe | More Probe Functions...
Graph menu: Plot | Plot Fourier of Curve
Graph menu: Plot | Plot Fourier of Curve (Cursor span)
```

The following prompt you to customise the settings:

```
Probe | Fourier | Probe Voltage Custom...
Probe | Fourier | Arbitrary...
```

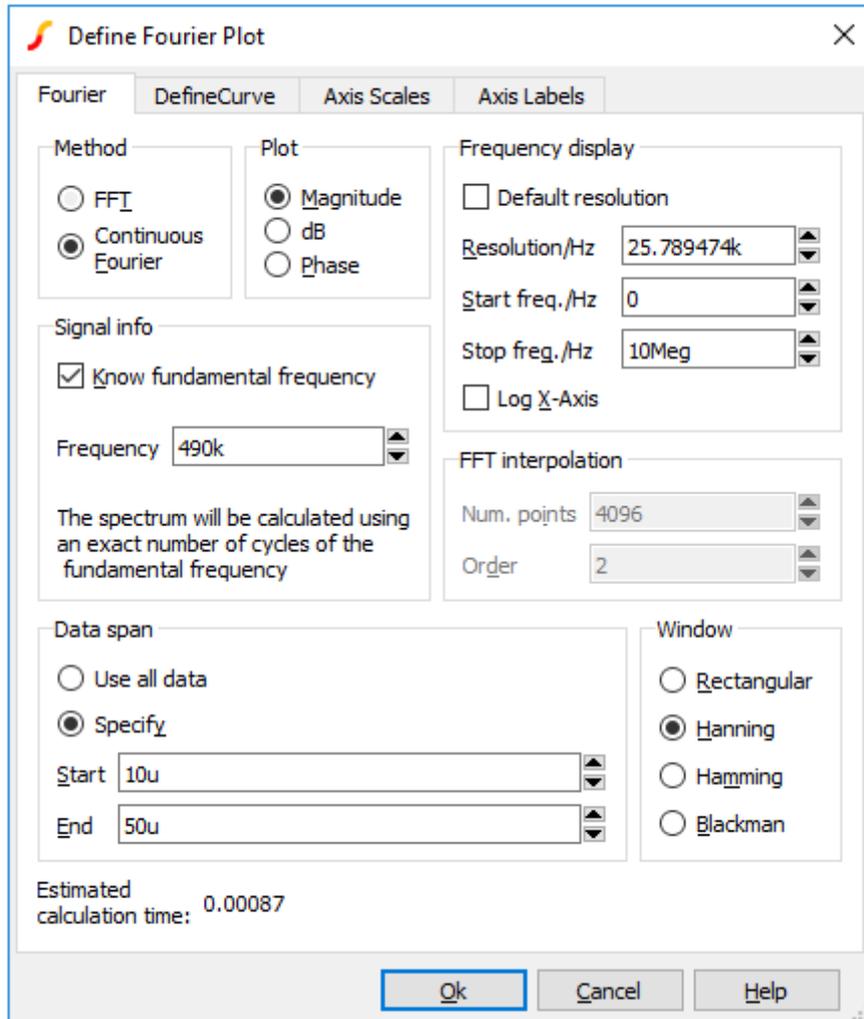
### Default Settings

The default fourier spectrum settings are:

Setting	Default value
Method	Interpolated FFT
Number of points	Next integral power of two larger than number of points in signal
Interpolation order	2
Span	All data except <b>Plot   Plot Fourier of Curve (Cursor span)</b> which uses cursor span

## Custom Settings

With menu **Probe | Fourier | Probe Voltage Custom...** you will see the dialog below. With the menus **Probe | Fourier | Arbitrary...** a dialog box similar to that shown in [“Plotting an Arbitrary Expression” on page 331](#) will be displayed but will include a Fourier tab. Click on the this tab to display the Fourier analysis options as shown below.



## Method

SIMetrix offers two alternative methods to calculate the Fourier spectrum: *FFT* and *Continuous Fourier*.

The simple rule is: use FFT unless the signal being examined has very large high frequency components as would be the case for narrow sharp pulses. When using Continuous Fourier, keep an eye on the Estimated calculation time shown at the bottom right of the dialog.

A description of the two techniques and their pros and cons follows.

FFT	<p>Fast Fourier Transform. This is an efficient algorithm for calculating a discrete Fourier transform or DFT. DFTs generally operate on evenly spaced sampled data. Unfortunately the data generated by the simulator is not evenly spaced so it is therefore necessary to interpolate the data before presenting it to an FFT algorithm. The interpolation process is in effect the sampling process and the Nyquist sampling theorem applies. This states that the signal can be perfectly reproduced from the sampled data if the sampling rate is greater than twice the maximum frequency component in the signal. In practice this condition can never be met perfectly and any signal components whose frequency is greater than half the sampling rate will be aliased to a different frequency.</p> <p>So if the number of interpolated points is too small there will be errors in the result due to high frequency components being aliased to lower frequencies. This is the Achilles heel of FFTs applied to simulated data.</p> <p>The <i>Continuous Fourier</i> technique, described next, does not suffer from this problem. It suffers from other problems the main one being that it is considerably slower than the FFT.</p>
Continuous Fourier	<p>This calculates the Fourier spectrum by numerically integrating the Fourier integral. With this method, each frequency component is calculated individually whereas with the FFT the whole spectrum is calculated in one - quite efficient - operation. Continuous Fourier does not require the data to be interpolated and does not suffer from aliasing.</p> <p>The problem with continuous Fourier is that compared to the FFT it is a slow algorithm and in many cases an FFT with a very large number of interpolated points can be calculated more quickly and give just as accurate a result.</p> <p>However, in cases where a signal has a very large high frequency content - such as narrow pulses - this method is superior and it is recommended that it is used in preference to the FFT in such situations. The continuous Fourier technique has the additional advantage that it can be applied with greater confidence as the aliasing errors will not be present. It does have its own source of error due to the fact that simulated data itself is not truly continuous but represented by unevenly spaced points with no information about what lies between the points. This error can be minimised by ensuring that close simulation tolerances are used. See <i>Simulator Reference Manual/Convergence, Accuracy and Performance</i> for details.</p> <p>Because each frequency component is calculated individually, the calculation time is affected by the values entered in <a href="#">“Frequency Display” on page 327</a>.</p>

## Plot (Phase or Magnitude)

The default is to plot the magnitude of the Fourier spectrum. Select **Phase** if you require a plot of phase or **dB** if you need the magnitude in dBs.

## Frequency Display

Resolution/Hz	Available only for the continuous Fourier method. This is the frequency interval at which the spectral components are evaluated. It cannot be less than $1/T$ where $T$ is the time interval over which the spectrum is calculated.
Start Freq./Hz	Start frequency of the display.
Stop Freq./Hz	Stop frequency of the display.
Log X-Axis	Check this to specify a logarithmic x-axis. This will force a minimum value for the start frequency equal to $1/T$ where $T$ is the time interval being analysed.

## Signal Info

If the signal being analysed is repetitive and the frequency of that signal is known *exactly* then a much better result can be obtained if it is specified here. Check the **Know fundamental frequency** box then enter the frequency. The Fourier spectrum will be calculated using an integral number of complete cycles of the fundamental frequency. This substantially reduces *spectral leakage*. Spectral leakage occurs because both the Fourier algorithms work on an assumption that the signal being analysed is a repetition of the analysed time interval from  $t = -\infty$  to  $t = +\infty$ . If the analysed time interval does not contain a whole number of cycles of the fundamental frequency this will be a poor approximation and the spectrum will be in error. In practice this problem is minimised by using a *window* function applied to the signal prior to the Fourier calculation, but using a whole number of cycles reduces the problem further.

Note that the fundamental frequency is not necessarily the lowest frequency in the circuit but the largest frequency for which all frequencies in the circuit are integral harmonics. For example if you had two sine wave generators of 1kHz and 1.1KHz, the fundamental is 100Hz, not 1kHz; 1kHz is the tenth harmonic, 1.1KHz is the eleventh.

You should *not* specify a fundamental frequency for circuits that have self-oscillating elements.

## FFT Interpolation

As explained above, the FFT method must interpolate the signal prior to the FFT computation. Specify here the number of points and the order. The number of points entry may be forced to a minimum if a high stop frequency is specified in the **Frequency Display** section.

The number of interpolation points required depends on the highest significant frequency component in the signal being analysed. If you have an idea what this is, a useful trick to set the number of points to a suitable value, is to increase the stop frequency value in the **Frequency Display** section up to that frequency. This will automatically set the number of interpolation points to the required value to handle that frequency. If you don't actually want to display frequencies up to that level, you can bring the stop frequency back down again. The number of interpolation points will stay at the value reached.

If in doubt, plot the FFT twice using a different number of points. If the two results are significantly different in the frequency band of interest, then you should increase the number of points further.

Usually an interpolation order of 2 is a suitable value but you should reduce this to 1 if analysing signals with abrupt edges. If analysing a smooth signal such as a sinusoid, useful improvements can be gained by increasing the order to 3.

## Data Span

Usually the entire simulated time span is used for the fourier analysis. To specify a smaller time interval click **Specify** and enter the start and end times.

Note that if you specify a fundamental frequency, the time may be modified so that a whole number of cycles is used. This will occur whether or not you explicitly specify an interval.

## Window

A window function is applied to the time domain signal to minimise spectral leakage (See above).

The choice of window is a compromise. The trade off is between the bandwidth of the main spectral component or lobe and the amplitude of the side-lobes. The rectangular window - which is in effect no window - has the narrowest main lobe but substantial side-lobes. The Blackman window has the widest main lobe and the smallest side lobes. Hanning and Hamming are something in between and have similar main lobe widths but the side lobes differ in the way they fall away further from the main lobe. Hamming starts smaller but doesn't decay whereas Hanning while starting off larger than Hamming, decays as the frequency moves away from the central lobe.

Despite the great deal of research that has been completed on window functions, for many applications the difference between Hanning, Hamming and Blackman is not important and usually Hanning is a good compromise.

There are situations where a rectangular window can give significantly superior results. This requires that the fundamental frequency is specified and also that the simulated signal is consistent over a large number of cycles. The rectangular window, however, *usually* gives considerably *poorer* results and must be used with caution.

### 13.4.7 Probing Buses

It is possible to probe a bus in which case a plot representing all the signals on the bus will be created. Usually this will be a numeric display of the digital bus data, but it is also possible to display the data as an analog waveform. Buses may contain either digital or analog signals; if any analog signals are present then threshold values must be supplied to define the logic levels of the analog signals.

#### To probe bus using default settings

Use the schematic popup menu **Probe Voltage...** or hot key F4 and probe the bus in the same way as you would a single wire.

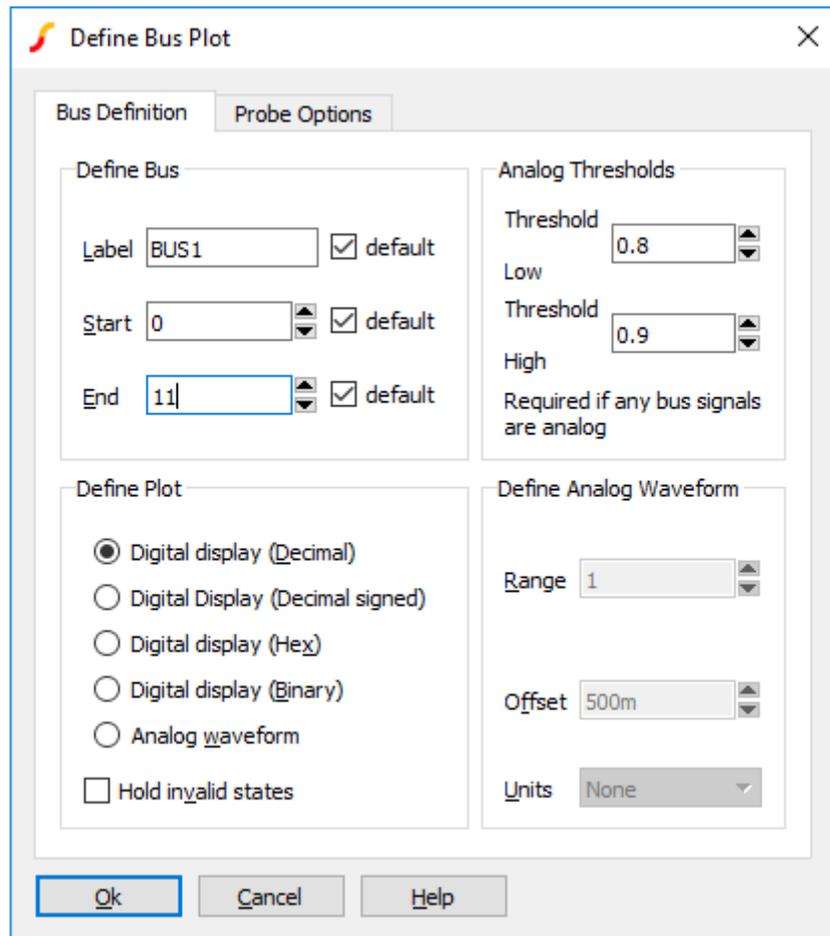
This will plot a numeric trace using decimal values.

#### To probe bus using custom settings

1. Select menu **Probe | Voltage - Bus...**
2. Click on desired bus.
3. Enter the desired bus parameters as described in [“Bus Probe Options” on page 329](#).

### 13.4.8 Bus Probe Options

The following describes the options available for random and fixed bus probes. These options are set using the dialog box shown below. See [“Probing Buses” on page 329](#) and [“Fixed Probe Options” on page 290](#) for details on plotting buses.



## Define Bus

- Label** This is how the curve will be labelled in the plot
- Start, End** Defines which wires in the bus are used to created the displayed data. The default is to use all wires.

## Plot Type

- Decimal/Hexadecimal/Binary** Each of these specifies a numeric display (see below) showing the bus values in the number base selected.
- Analog waveform** Specifies that the bus data should be plotted as an analog waveform.
- Hold invalid states** If checked, then and invalid digital states found in the data will be replaced with the most recent valid state. If not checked, invalid states will be shown as an 'X' in numeric displays. This option is automatically selected for analog waveform mode.

## Analog Thresholds

These are required if any of the signals on the bus is analog. These define the thresholds for converting to logic levels.

Threshold	Low Analog voltage below which the signal is considered a logic zero.
Threshold	High Analog voltage above which the signal is considered a logic one.

If a signal is above the lower threshold but below the upper threshold, it will be considered as 'unknown'.

## Define Analog Waveform

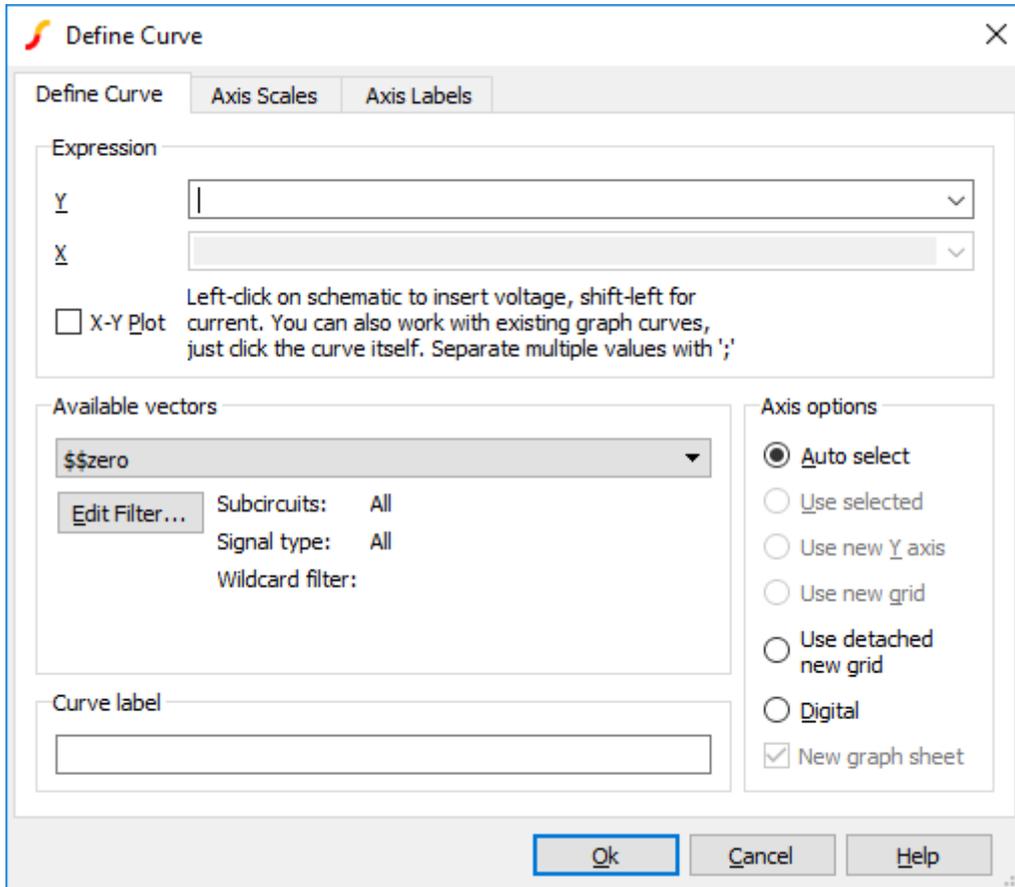
Only enabled if Analog waveform is specified in the Plot Type box. Specifies the scaling values and units for analog waveforms:

Range	Peak-peak value used for display.
Offset	Analog display offset. A value of zero will result in an analog display centred about the x-axis.
Units	Select an appropriate unit from the drop down box.

### 13.4.9 Plotting an Arbitrary Expression

If what you wish to plot is not in one of the probe menus, SIMetrix has a facility to plot an arbitrary expression of node voltages or device currents. This is accessed via one of the menus **Probe | Add Curve....** Selecting one of these menus brings up the Define Curve dialog box shown below.

## Define Curve Sheet



## Expression

- Y Enter arithmetic expression. This can use operators + - \* / and ^ and various functions such as log, sin etc. To enter a node voltage, click on a point on the schematic. To enter a device pin current, hold down the shift key and click on the device pin in the schematic. Both voltages and currents may also be selected from the Available Vectors box.

You may also plot an expression based on any curve that is already plotted. Simply click on the curve itself and you should see a function entered in the form  $cv(n)$  where  $n$  is some integer.

Any entries made in this box are stored for future retrieval. Use the drop down box to select a previous entry.

- X Expression for X data. Only required for X-Y plot and you must check X-Y Plot box. Expression entered in the same way as for Y data.

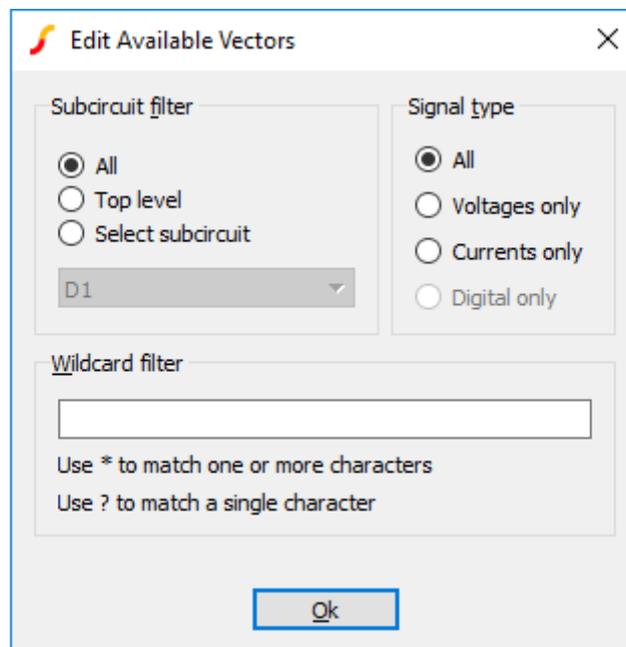
## Available Vectors

Lists values available for plotting. This is for finding vectors that aren't on a schematic either because the simulation was made direct from a netlist or because the vector is for a voltage or current in a sub-circuit. Refer to “[Data Handling and Keeps](#)” on page 255 for more information. Press Edit Filter to alter selection that is displayed. See below.

The names displayed are the names of the vectors created by the simulator. The names of node voltages are the same as the names of the nodes themselves. The names for device currents are composed of device name followed by a '#' followed by the pin name. Note that some devices output internal node voltages which could get confused with pin currents. E.g. q1#base is the internal base voltage of q1 not the base current. The base current would be q1#b. For the vector names output by a noise analysis refer to *Simulator Reference Manual/Command Reference/.NOISE*.

## Edit Filter...

Pressing the Edit Filter... button opens:



This allows you to select what is displayed in the available vectors dialog. This is useful when simulating large circuits and the number of vectors is very large.

## Sub-circuit Filter

All	Vectors at all levels are displayed.
Top level	Only vectors for the top-level are displayed
Select sub-circuit	All sub-circuit references will be displayed in the list box. Select one of these. Only vectors local to that sub-circuit will be displayed in Available Vector list.

## Signal Type

All	List all signal types
Voltages Only	Only voltages will be listed
Currents Only	Only currents will be listed
Digital Only	Only digital vectors will be listed

## Wildcard filter

Enter a character string containing '\*' and/or '?' to filter vector names. '\*' matches 1 or more occurrences of any character and '?' matches any single character. Some examples:

*	matches anything
X1.*	matches any signal name that starts with the three letters: X1.
X?.*	matches any name that starts with an X and with a '.' for the third letter.
*.q10#c	matches any name ending with .q10#c i.e the current into any transistor called q10
*.U1.vout	matches any name ending with .U1.C11 i.e any node called vout in a subcircuit with reference U1.

## Curve Label

Enter text string to label curve

## Axis Options

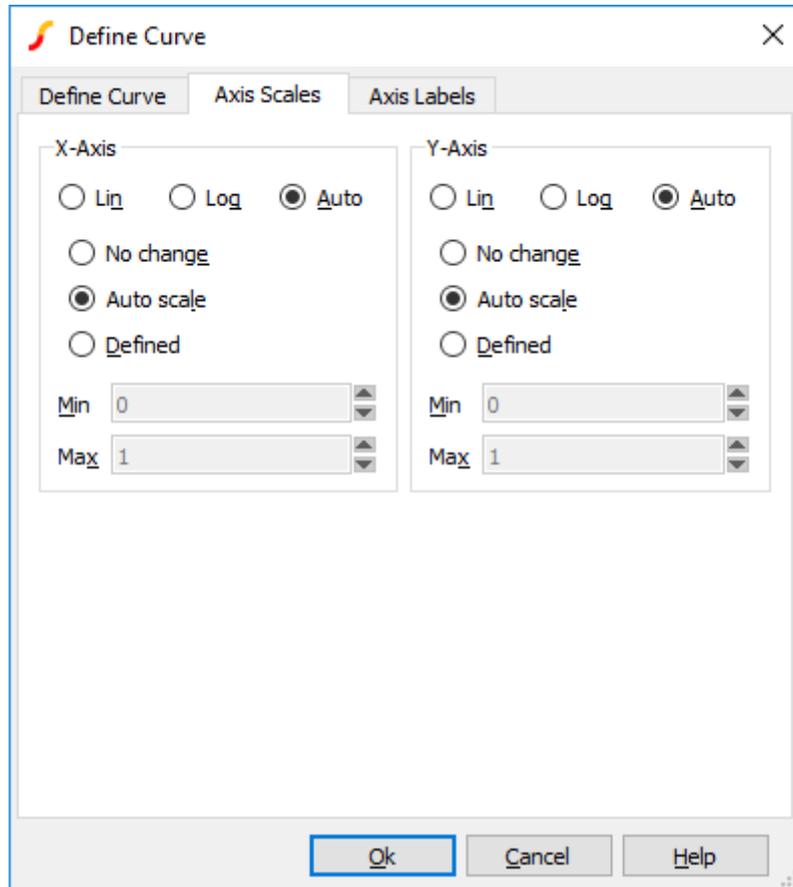
Selections in this group allow you to control where the curve for the probed signal will be placed.

Auto select	Select an appropriate axis automatically. See <a href="#">“AutoAxis Feature” on page 343</a>
Use selected	Use currently selected y-axis
Use new Y-axis	Create a new y-axis in the selected grid. If there is more than one grid, the selected grid will have a double vertical line drawn at the far left. A grid can be selected by clicking in it
Use new Grid	Plots the curve in a new grid
Use detached new grid	Plots the curve in a new grid with a detached x-axis. A detached x-axis is independent and may be zoomed independently of other axes
Digital	Create a new digital axis. Digital axes are placed at the top of the window and are stacked. Each one may only take a single curve. As their name suggests, they are intended for digital traces but can be used for analog signals if required
New graph sheet	Creates a completely new graph sheet for the plot

Note that you can move a curve to a new axis or grid after it has been plotted. See [“Moving Curves to Different Axis of Grid”](#) on page 344.

## Axis Scales Sheet

Allows you to specify limits for x and y axes.



## X-Axis/Y-Axis

- |              |  |
|--------------|--|
| Lin/Log/Auto | Specify whether you want X-Axis to be linear or logarithmic. If Auto is selected, the axis (X or Y) will be set to log if the x values are logarithmically spaced. For the Y-axis it is also necessary that the curve values are positive for a log axis to be selected. |
| No Change    | Keep axes how they are. Only relevant if adding to an existing graph.  |
| Auto scale   | Set limits to fit curves.  |
| Defined      | Set to limits defined in Min and Max boxes.  |

## Axis Labels Sheet

This sheet has four edit boxes allowing you to specify, x and y axis labels as well as their units. If any box is left blank, a default value will be used or will remain unchanged if the axis already has a defined label.

### 13.4.10 Curve Arithmetic

SIMetrix provides facilities for performing arithmetic on existing curves. For example you can plot the difference between two plotted curves.

There are two methods:

1. With the menus:

Plot | Sum Two Curves...

Plot | Subtract Two Curves...

Plot | Multiply Two Curves...

Select one of the above menus and follow instructions given.

2. Using the Add Curve... dialog box. With this method, select menu **Probe | Add Curve...** or **Plot | Add Curve...** then enter an expression as desired. To access an existing curve's data, simply click on the curve. For more information, see ["Plotting an Arbitrary Expression" on page 331](#).

## AC Analysis Notes on Curve Arithmetic

In AC analysis, the results are complex. When plotting a curve from an AC analysis, the magnitude of the complex data is plotted unless some other explicit function is applied such as `phase()` or `imag()`. Although the magnitude of the data is plotted, the graph system retains the original complex values. So any arithmetic operation performed directly on complex plotted data will also be complex. For example, if you have two curves from an AC analysis and you choose the Plot | Subtract Two Curves... menu to subtract them, the new result will be the magnitude of the complex difference not the difference in the magnitudes as might be expected. In mathematical terms, you will see  $|a-b|$  not  $|a|-|b|$ .

### 13.4.11 Using Random Probes in Hierarchical Designs

Random probes may successfully be employed in hierarchical designs. There are however some complications that arise and these are explained below.

## Closed Schematics

Read the following if you find situations where cross-probing inside hierarchical blocks sometimes fails to function.

The names used for cross-probing are stored in the schematic itself and are saved to the schematic file. These netnames are not assigned until the netlist is created and this doesn't usually happen until a simulation is run. A problem arises, however, if the schematic is not open. If netnames have never been created then they won't be updated during the run as, by default, SIMetrix will not update a closed file.

This problem can be resolved by giving SIMetrix permission to update schematic files that are closed. To do this, type at the command line:

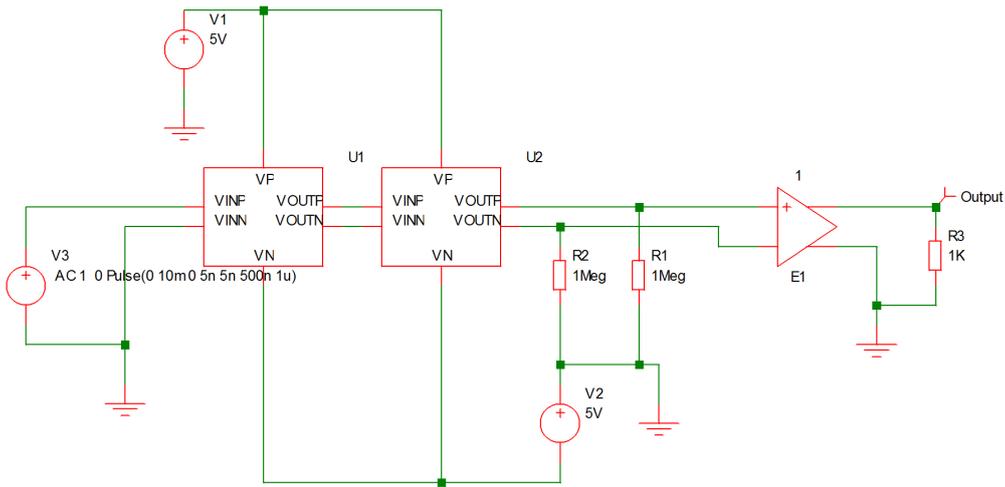
```
Set UpdateClosedSchematics
```

This only needs to be done once. Note that you can only do this with the full versions of SIMetrix.

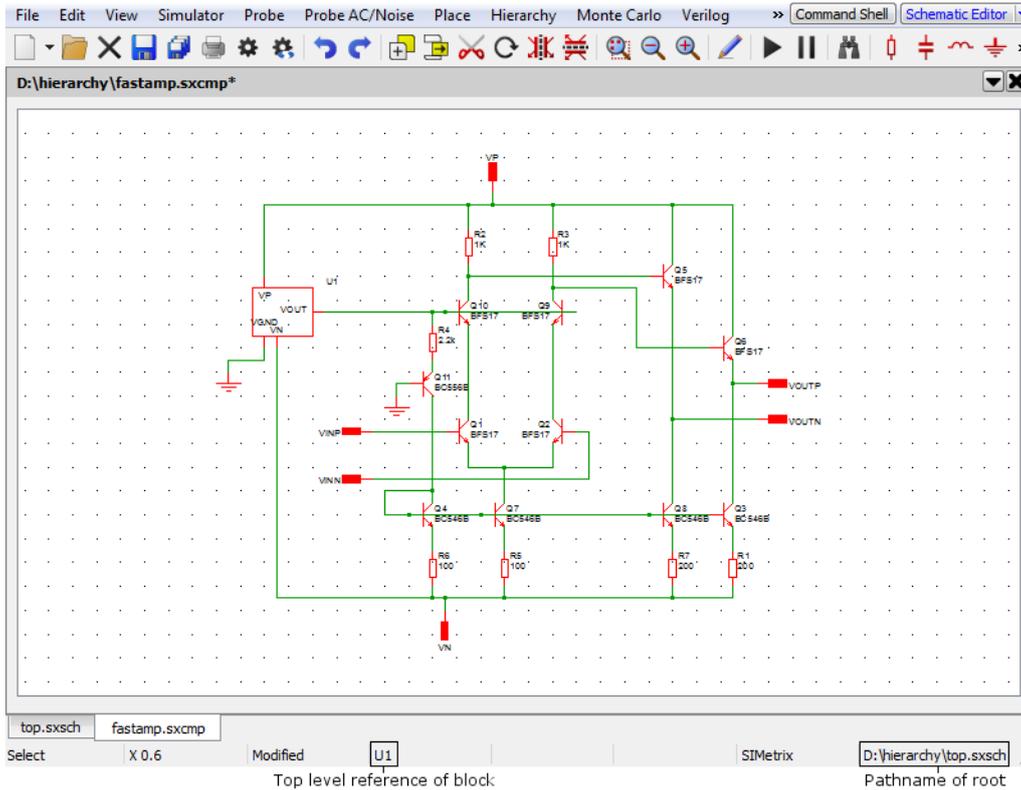
This problem won't arise if you always run every schematic at least once while it is open. If you do this, the netnames will be updated and you will be prompted to save the schematic before closing it.

### Multiple Instances

An issue arises in the situation where there are multiple instances of a block attached to the same schematic. Consider the following top level circuit.



This has two instances of the block fastamp.sxsch U1 and U2. Suppose you wanted to plot the voltage of a node inside U1. The schematic fastamp.sxsch is open but to which block does it refer? The answer is that it will refer to the most recent block that was used to descend into it. The block that a schematic refers to is always displayed in the schematic's status bar as illustrated below



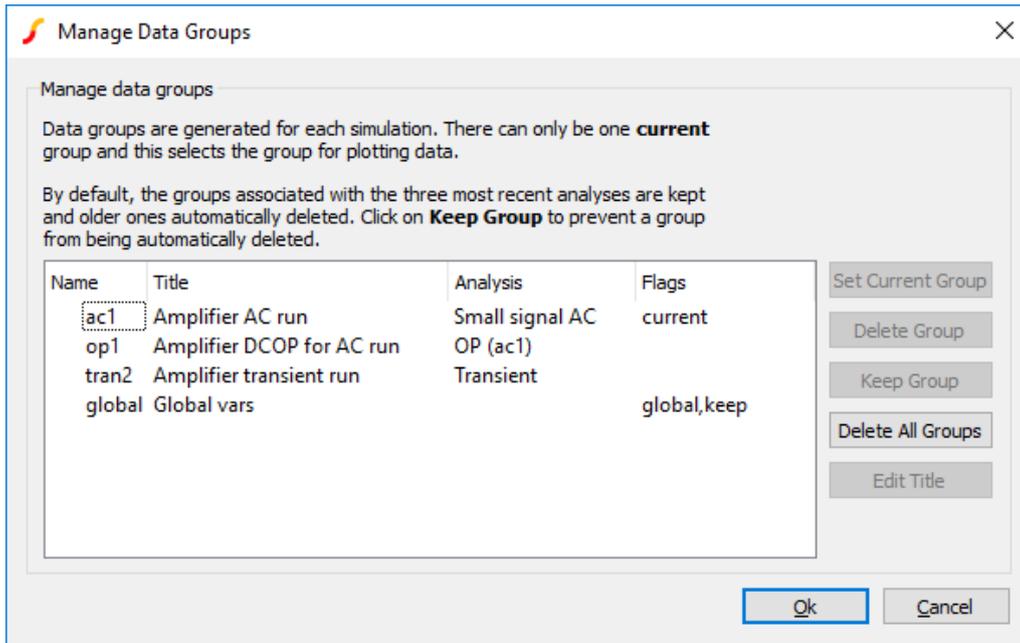
To plot a node in U2, ascend to parent (top.sxsch in the above example) then descend into U2. The same schematic as above will be displayed but will now refer to U2 instead of U1.

## Plotting Currents

In the same way that you can plot currents into subcircuits in a single sheet design, so you can also plot currents into hierarchical blocks at any level.

### 13.4.12 Plotting the Results from a Previous Simulation

1. Select the menu item **Simulator | Manage Data Groups...** This will open a dialog box similar to the following:



2. Select the name of the previous run (or group) that you require. The current group will be highlighted. Note that the AC analysis mode generates two groups. One for the AC results and the other for the dc operating point results. Transient analysis will do the same if the start time is non-zero. The names for the groups usually default to the file name of the schematic that generated them. (They have been edited to be more meaningful in the picture above).
3. Click **Set Current Group**
4. Click **Ok**
5. Plot the result you require in the normal way. A word of warning: If the schematic has undergone any modifications other than part value changes since the old simulation was completed, some of the netnames may be different and the result plotted may not be of what you were expecting.

**Note** By default, only the three most recent groups are kept. This can be changed using the GroupPersistence option using the Set command - see [“Set” on page 396](#)). You can also keep specify that a particular group is kept permanently using the Data Groups Manager. Proceed as follows:

1. Select the menu item **Simulator | Manage Data Groups....** This will open the Data Group Manager dialog box. (See [“Plotting the Results from a Previous Simulation” on page 338](#) for an illustration).
2. Select the group or groups you wish to keep permanently
3. Click **Keep Group**. You should see the Flags column for each group display keep
4. Click **Ok**

Although only three groups are held at a time, the data is actually stored on a disc file which will not necessarily have been deleted. If you wish to access an old run, use **File | Data | Load Temporary Data....** The file will have the same name as the group appended with .SXDAT. Whether or not the data file is still available depends on a preference setting. See [“Graph/Probe/Data Analysis” on page 447](#) for details.

## Combining Results from Different Runs

There are occasions when you wish to - say - plot the difference between a node voltage for different runs. You can do this in SIMetrix using the **Probe | Add Curve...** menu by entering an expression such as `'vector1 -vector2'` in the y-expression box, where *vector1* and *vector2* are the names of the signals. However, as the two signals come from different runs we need a method of identifying the run. This is done by prefixing the name with the *group name* followed by a colon. The group name is an analysis type name (tran, ac, op, dc, noise, tf or sens) followed by a number. The signal name can be obtained from the schematic. For voltages, move the cursor over the node of interest and you will see the name appear in the status box in the form "NET=???". For currents put the cursor on a device pin and press control-P. The group name is displayed in the simulator progress box when the simulation is running. You can also find the current group by selecting menu **Simulator | Manage Data Groups...** The current group is marked **current** in the **Flags** column.

Here is an example. In tutorial 1, the signal marked with the *Amplifier Output* probe is actually called *Q3\_E*. The latest run (group) is called *tran4*. We want to plot the output subtracted from the output for the previous run. The previous run will be *tran3*. So we enter in the y-expression box:

```
tran4:q3_e-tran3:q3_e
```

For more details on data groups, please refer to the *Script Reference Manual/Script Language/Accessing Simulation Data*.

## 13.5 Plot Journals and Updating Curves

### 13.5.1 Overview

You can repeat previous plotting operations in one of two ways.

The 'Update Curves' feature rebuilds the current graph sheet using the latest available data. This allows you to randomly probe a schematic and then update the curves with new results for a new simulation run.

The 'Plot Journal' feature allows you to save the plots in the current graph sheet for later reconstruction. This doesn't save the data, it saves the vector names and expressions used to create the graph's curves. In fact this is done by building a SIMetrix script to plot the curves.

### 13.5.2 Update Curves

Make sure that no curves are selected then select graph menu **Plot | Update Curves**. The curves currently on the graph sheet will be redrawn using the current simulation data. Although this would usually be the latest simulation run, you can also use this feature to restore the curves back to those from an earlier run. Use the **Simulator | Manage Data Groups...** menu to select earlier data. (For more information see "[Plotting the Results from a Previous Simulation](#)" on page 338).

### Options

By default all curves are redrawn, that is the older ones are deleted. You can change this behaviour so that older curves are kept. Select menu **Plot | Update Curves Settings...** then uncheck the **Delete old curves** box.

If there are curves that you would like to remain fixed and so won't be updated, simply select them first. This behaviour can be overridden using the menu **Plot | Update Curves Settings...** Simply uncheck the **Ignore selected curves** box.

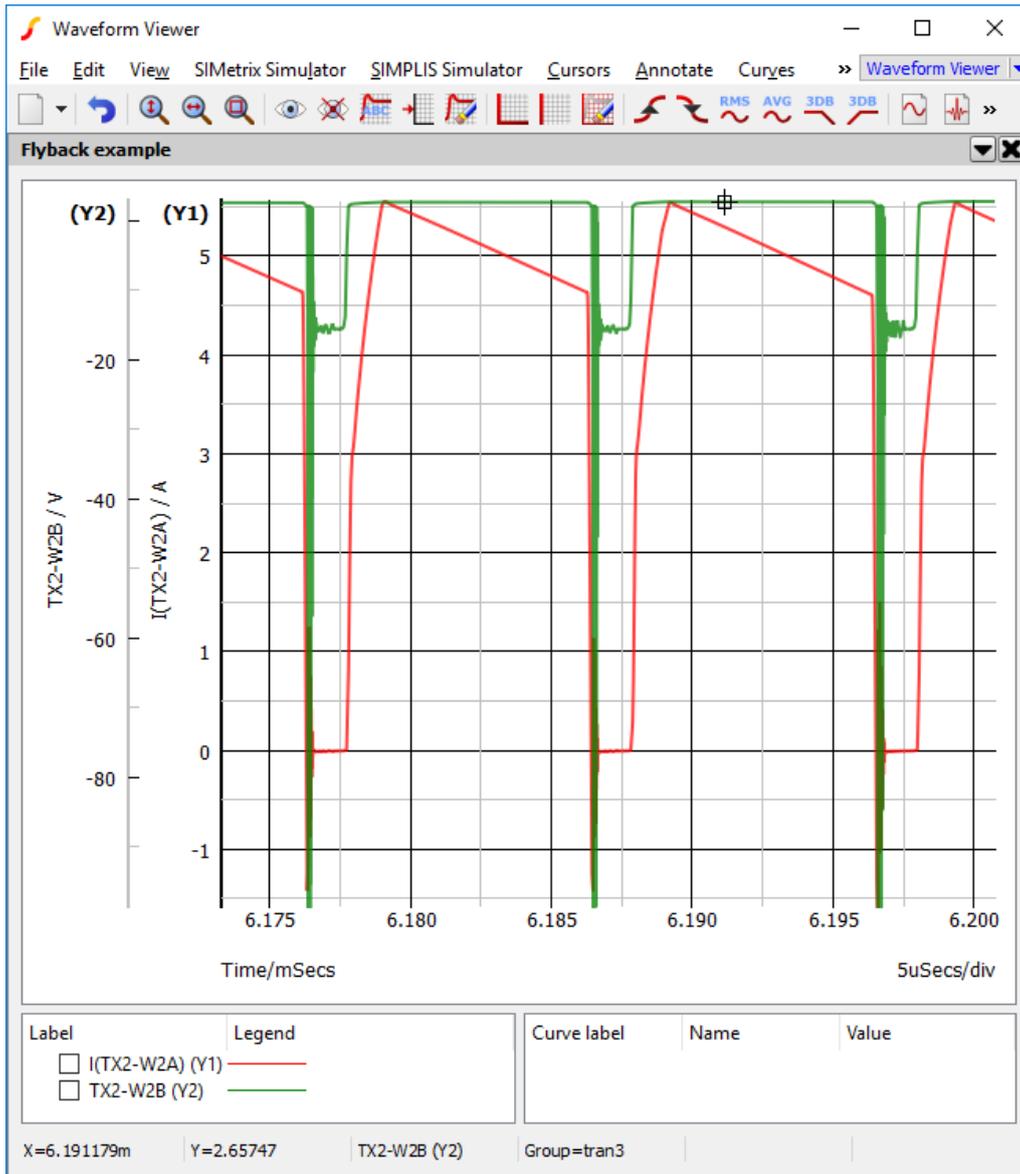
### 13.5.3 Plot Journals

First create a plot journal using the menu **Plot | Create Plot Journal...** then choose a file name. The file created has a .sxsr extension - its the same extension used by scripts because the file created *is* a script.

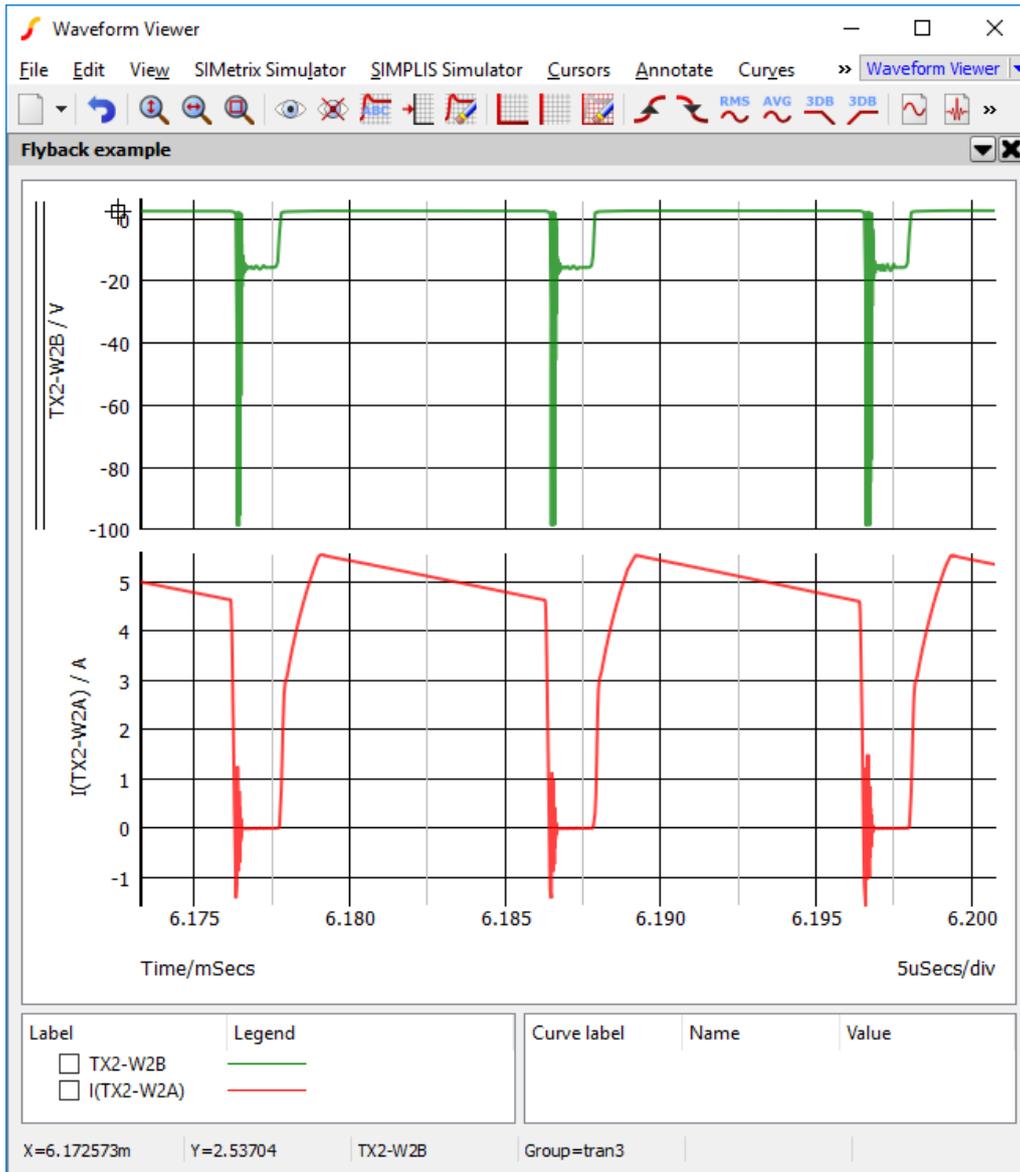
To run the plot journal, you will of course first need to run a simulation or load previous data so that the journal has some data to work with. The plot journal itself does not store any data. With the simulation data you wish to work with in place, select graph menu **Plot | Run Plot Journal...** This simply runs a script located in the current directory. Note that the plot journal always creates a new graph sheet.

## 13.6 Graph Layout

Graphs may have additional Y axes to accommodate plotting results with incompatible scales. This occurs particularly for plotting dB and phase against each other and also for voltage and current. The additional Y axes may either be superimposed or stacked. In the user interface and the remainder of this documentation these are referred to respectively as *Axes* and *Grids*. These are illustrated below.



Current and Voltage plotted on separate Axes



Current and Voltage plotted on separate Grids

### 13.6.1 AutoAxis Feature

When you plot a new curve on an existing graph, SIMetrix will select - or if necessary create - a compatible axis for that curve. The decision is made on the basis of the curve's Units i.e voltage, current etc. The rules it follows are:

1. If the currently selected axis or grid (shown by black axis line) has the same units as curve to be plotted or if it has undefined units (designated by a '?' on label), that axis will be used.
2. If any other axis on the currently selected grid has compatible units (i.e same as curve or undefined) that axis will be used.
3. If no axes exist with compatible units on the selected grid, a new y-axis will be created to accommodate the curve.

The above works for all plots made using random probes. For plots created with fixed probes see ["Fixed Probes" on page 288](#) for more details. For plots created using the Curve command at the command line,

the `/AutoAxis` switch must be specified e.g:

```
Curve /AutoAxis L3#P
```

### 13.6.2 Manually Creating Axes and Grids

Two toolbar buttons **Create new grid** and **Create new axis** allow manual creation of new axes and grids. These will be initially empty. Subsequent random probe operations will use the new axis or grid unconditionally as long as it remains selected (see below).

### 13.6.3 Selecting Axes

Some operations are performed on the selected axis or grid. The selected y-axis will be displayed with its vertical axis line a deep black while the remaining axes will be light grey. Selected grids are displayed with a double vertical line at the far left. Newly created axes and grids are always selected. To select an axis, click the left mouse button immediately to the left of the vertical axis line.

### 13.6.4 Stacking Curves to Multiple Grids

The menu **Curves | Stack All Curves** will place each curve on its own grid.

The menu **Curves | Stack Selected Curves** will place each selected curve on its own grid.

### 13.6.5 Moving Curves to Different Axis or Grid

You can freely move curves around from one axis or grid to another. Proceed as follows:

1. Select the curve or curves you wish to move by checking its checkbox next to the coloured legend which designates the curve.
2. Select the axis you wish to move it to. (See above)
3. Press the **Move selected curves to new axis** button. The curves will be re-drawn on the new axis. Any axes that become empty as a result of this operation will be deleted unless it is the Main axis. See [“Deleting Axes” on page 345](#).

### 13.6.6 Detached X-Axes

Grids may have attached or detached x-axes. Attached x-axes are shared with other x-axes in the same graph sheet. That is they zoom in sync - when the limits are changed on one grid, the other grids' x-axes are also changed.

Detached x-axes are independent and have their own axis limits. Detached axes may be created in a number of ways:

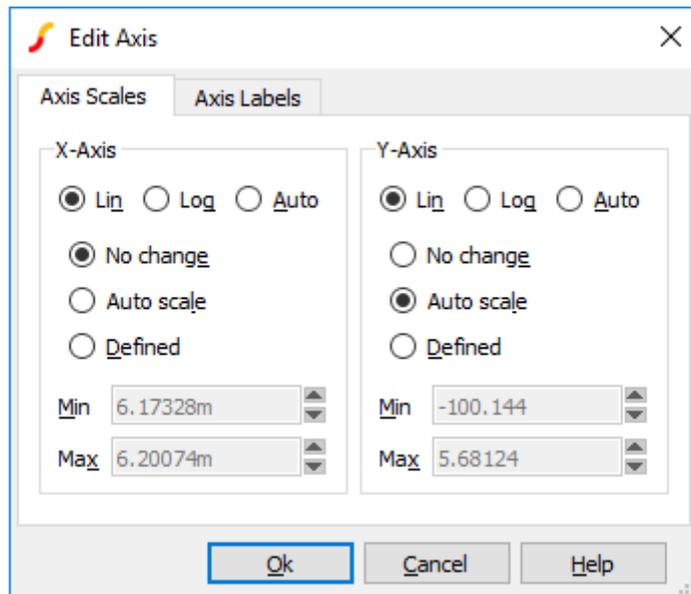
- In the Add Curve dialog box (menu **Probe | Add Curve...**) select the use detached new grid option in the **Axis options** group. This will plot the curve on a new grid with a detached x-axis.
- Any existing grid can be converted to detached status using menu **Axes | Detach selected x-axis**.
- Holding down the control key while zooming will create a new grid with a detached x-axis. This is a convenient way to create a new view of some detail of a curve without losing the original wide view
- Detached axes will automatically be created in situations where the existing shared axis is not compatible with physical type of the new curve being plotted. For example, if you plot an AC analysis result in a graph sheet containing only time axes, a detached axis will be created to plot it.

### 13.6.7 Deleting Axes

To delete an axis, select it then press Erase axis button or menu **Axes | Delete Axis/Gris**.

### 13.6.8 Editing Axes

You can edit axis scales, label and units by selecting the graph popup menu **Axes | Edit Axis....** This brings up the following dialog box:



The function of the Axis scales sheet and axis labels sheet is similar to the sheets of the same name in the define curve dialog box. See [“Plotting an Arbitrary Expression” on page 331](#) for details.

### 13.6.9 Reordering Grids and Digital Axes

You can change the vertical order of the analog grids and digital axes. To change the analog grid order:

1. Select **Axes | Reorder Grids...**
2. You will be presented with a list of currently displayed grids identified by their y-axis title. Use the up and down arrow buttons to arrange them in the order required then click OK

To change the digital axis order:

1. Select menu **Axes | Reorder Digital Axes...**
2. Rearrange entries in list as described above for analog grids.

## 13.7 Curve Operations

### 13.7.1 Selecting Curves

Curves may be selected by clicking on the curve. To select multiple curves, hold down the control key. Curves may also be selected by checking the associated check box in the legend window.

### 13.7.2 Deleting Curves

To delete a curve (or curves), select it (or them) then press the **Erase selected curves** button. Any y-axis that is left empty after deleting curves from it will automatically be deleted. By default any grids left empty after deleting curves from it will also be deleted. This action can be overridden using an option setting. Refer to [“Graph/Probe/Data Analysis Options” on page 447](#)

### 13.7.3 Hiding and Showing Curves

A curve may be hidden without it actually being deleted. This is sometimes useful when there are many curves on a graph but the detail of one you wish to see is hidden by others. In this instance you can temporarily remove the curves from the graph. To hide a curve (or curves) select it (or them) then press the **Hide selected curves** button. To show it (or them) again, press the **Show selected curves** button.

### 13.7.4 Re-titling Curves

You can change the title of a curve by selecting it then pressing the **Name curve** button. This will change the name of the curve as displayed in the legend panel. (Above main graph area and below toolbar)

### 13.7.5 Highlighting Curves

You can highlight one or more curves so that they stand out from the others. This is useful if there are many overlapping curves displayed.

#### To Highlight Curves

Select the curves you wish to highlight then press ‘H’ or menu **Curves | Highlight Selected Curves**.

#### To Highlight Curve Divisions

For grouped curves created from multi-step analyses, you can highlight individual divisions. Position the small cross cursor over the curve of interest then select right-click menu **Highlight | Highlight current division**

#### To Unhighlight Curves

Select the curves you wish to unhighlight then press ‘U’ or menu **Curves | Unhighlight Selected Curves**.

#### To Unhighlight All Curves

Select menu **Curves | Unhighlight All Curves**.

### 13.7.6 Moving and Copying Curves Between Graphs

The menus **Edit | Copy**, **Edit | Cut** and **Edit | Paste** may be used to move and copy curves between graph sheets. Note that these operations use an internal clipboard and do not place any data in the system clipboard.

The internal clipboard uses an efficient method for transferring curves that uses very little memory even if the curve is large. Also, if you copy a curve, the data itself is not copied internally; the two curves just reference the same data. This makes copying a memory efficient operation.

### To Move a Curve to a New Graph Sheet

1. Select the curve or curves you wish to move.
2. Select menu **Edit | Cut**.
3. Either create a new graph sheet to receive the new curves (use **F10**) or switch to an existing graph sheet.
4. Select menu **Edit | Paste**.

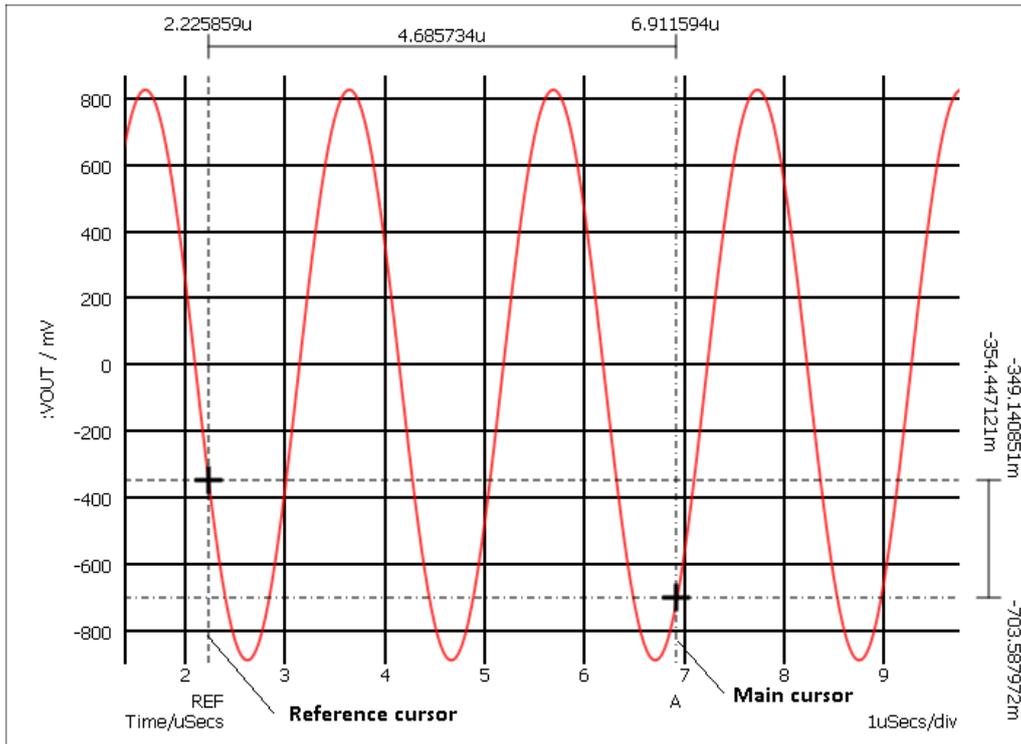
### To Copy a Curve to a New Graph Sheet

1. Select the curve or curves you wish to move.
2. Select menu **Edit | Copy**.
3. Either create a new graph sheet to receive the new curves (use **F10**) or switch to an existing graph sheet.
4. Select menu **Edit | Paste**.

## 13.8 Graph Cursors

### 13.8.1 Overview

Graph cursors can be used to make measurements from waveforms. In their default configuration they consist of two dimensioned crosshairs as shown below:



The cursors can be moved horizontally or vertically while tracking an attached curve or they can be picked up and dragged onto another curve.

Initially there are just two cursors, but there is the facility to add additional cursors without any maximum limit.

## 13.8.2 Cursor Operations

### Displaying

To switch on/off the cursor REF display select the graph menu **Cursors | Toggle On/Off**.

### Moving

Cursors can be moved by a number of methods:

1. Left to right. In this mode the x-position of the cursor is varied while the cursor tracks the curve to which it is attached. To use this method, place the mouse on the vertical crosshair but away from the intersection with the horizontal crosshair. You should see the mouse cursor shape change to a left-right arrow. Press left mouse key and drag.
2. Up-down. Similar to 1. above but instead the y-position is varied. To use this method, place the mouse on the horizontal crosshair but away from the intersection with the vertical crosshair. You should see the mouse cursor shape change to an up-down arrow. Press left mouse key and drag.
3. Drag and drop. In this mode the cursor is picked up and moved without tracking any curve. It can be dropped to any location and will then snap to the nearest curve. To use this method, place the mouse cursor at the intersection of the crosshairs. You will see the cursor shape change to a four-pointed arrow. Press left key and drag to new location.

4. The reference cursor can be moved in a left-right mode using the right mouse button.
5. Both cursors can be moved together using the left button while holding down the shift key.

## Moving Cursors along a Curve

You can move a cursor to a peak or trough using the hot-key defined in the following table

Key	Function
<b>F5</b>	Move main cursor to next peak
<b>shift-F5</b>	Move main cursor to previous peak
<b>F6</b>	Move main cursor to next trough
<b>shift-F6</b>	Move main cursor to previous trough
<b>F7</b>	Move reference cursor to next peak
<b>shift-F7</b>	Move reference cursor to previous peak
<b>F8</b>	Move reference cursor to next trough
<b>shift-F8</b>	Move reference cursor to previous trough

These operations can also be accessed from the graph menu **Cursors | Move**.

## Hiding Cursors

You can temporarily hide all or some of the displayed cursors. Menu **Cursors | Hide/Show | All** has a toggle action and will hide all cursors if all cursors are currently displayed and vice-versa. If some cursors are visible and some are hidden, you will be presented with an option to hide all cursors or show all cursors.

Menu **Cursors | Hide/Show | Select** allows you to selectively hide or show some cursors.

## Freezing Cursors

You can freeze the cursors so that they can't be moved accidentally. Select menu **Cursors | Freeze/Unfreeze**.

## Aligning Cursors

Select menu **Cursors | Align** to align the two cursors so that they have the same y position.

### 13.8.3 Additional Cursors

SIMetrix has the ability to display any number of cursors, not just the standard two.

#### To Add an Additional Cursor

1. Select menu **Add Additional Cursor...**
2. Enter a suitable label for the cursor. This is displayed at the bottom of the graph and to avoid clutter, we recommend that you use a short label such as a single letter.

3. Select to which other cursor you wish the new cursor to be referenced for both horizontal and vertical dimensions. Select ‘\*\*none\*\*’ if you do not wish it to be referenced to any currently displayed cursor. Note that you may reference further additional cursors to this one if desired.
4. Click **Ok**. The new cursor will be initially displayed at the start of the x-axis and attached to the first curve on the sheet. You may subsequently move it as desired.

## To Remove Additional Cursors

1. Select **Cursors | Remove Additional Cursors...**
2. Select the cursor or cursors to be removed. These are identified by their labels.

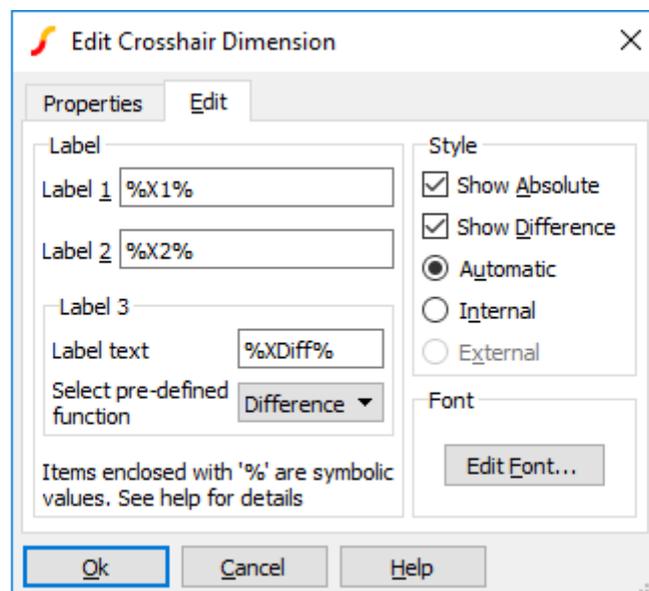
### 13.8.4 Cursor Readout

There are a number of options as to how the cursors’ absolute and relative positions are displayed. Initially all values are displayed as dimensions on the graph. This can be altered in a number of ways:

- You can opt to have just the absolute or just relative readings displayed
- The actual format of the graph readout can be customised. E.g extra text can be added, perhaps something like ‘Delay = xxxnS’ where xxx is the relative reading.

## Editing Style or/and Format of Cursor Dimension

Double click on one of the displayed values of the cursor dimension. The following dialog will open:



Edit values as described below:

**Label** The labels are the three values displayed on the dimension. Label 1 is the value displayed above the reference cursor, label 2 is the value displayed above the main cursor and label 3 is the value displayed as the difference.  $%x1%$ ,  $%x2%$  and  $%xdiff%$  are symbolic values that will be substituted with the absolute position of the reference cursor, the absolute position of the main cursor and the difference between them respectively. You can add additional text to these. For example, if you changed label 1 to ‘Pulse Start =  $%x1%$ ’ the value displayed for the position of the reference cursor would be prefixed with ‘Pulse Start = ‘.

You can use the pre-defined function drop-down box to choose some commonly used symbolic values and expressions.

You can use expressions relating constants and symbolic values enclosed by ‘%’. Expressions must be enclosed in braces: ‘{’ and ‘}’. For example, the expression  $1/%xdiff%$  will cause the difference value to be displayed as a reciprocal. This is useful if you wanted to display a frequency instead of a period. For a detailed description of this feature, see “[Graph Symbolic Values](#)” on page 366.

You can use any arithmetic operator along with many of the functions described in the *Script Reference Manual/Function Reference* in these expressions.

### Style

Show Absolute	Clear check box to disable display of the absolute positions of the cursors.
Show Difference	Clear check box to disable display of relative positions.
Automatic/Internal/External	Style of dimension. Internal means that the arrows will always be displayed between the cursors. External means they will always be displayed outside the cursors. In automatic mode the style will change according to the spacing and position.

Note, if you clear both absolute and difference, you will only be able to restore the display of the dimension by switching cursors off then on again.

**Font** Select font used for readout text.

**Properties Tab** The properties tab lists all available properties of the CrosshairDimension object. This will probably only be of interest if you are writing custom scripts to manipulate cursor dimensions. More information on this subject can be found in the *Script Reference Manual/Applications/Grph Objects*.

## Show Curve Info

The menu **Cursors | Show Curve Info** will display in the command shell information about the curve which currently has the main cursor attached. The following information is listed:

Curve name	
Source group	The name of the simulation group that was current when the curve was created.
Curve id	Only required when accessing curves using script commands.
Run number	If there are multiple curves generated by a Monte Carlo run, this is a number that identifies the run number that created the curve. This number can be used to plot the curve alone and also to identify the seed value used for that Monte Carlo step.

### 13.8.5 Cursor Functions

There are four functions which return the current positions of the cursors and these can be used in script expressions. These are

XDatum()

YDatum()

XCursor()

YCursor()

See *Script Reference Manual/Function Reference* for details.

## 13.9 Curve Measurements

### 13.9.1 Overview

A number of measurements can be applied to selected curves. The results of these measurements are displayed in the measurement window below the main graph drawing area. See [“Elements of the Graph Window” on page 287](#).

Some of these measurements can be selected from the tool bar while the remainder may be accessed via the menu **Measure | More Functions...** or by pressing **F3**.

Measurement functions may also be applied to [“Fixed Probes” on page 288](#) so that they are automatically updated when a simulation is repeated. See [“Applying Measurements to Fixed Probes” on page 356](#) for more details.

In general to perform a measurement, select the curve or curves then select measurement from tool bar or menu. If there is only one curve displayed, it is not necessary to select it.

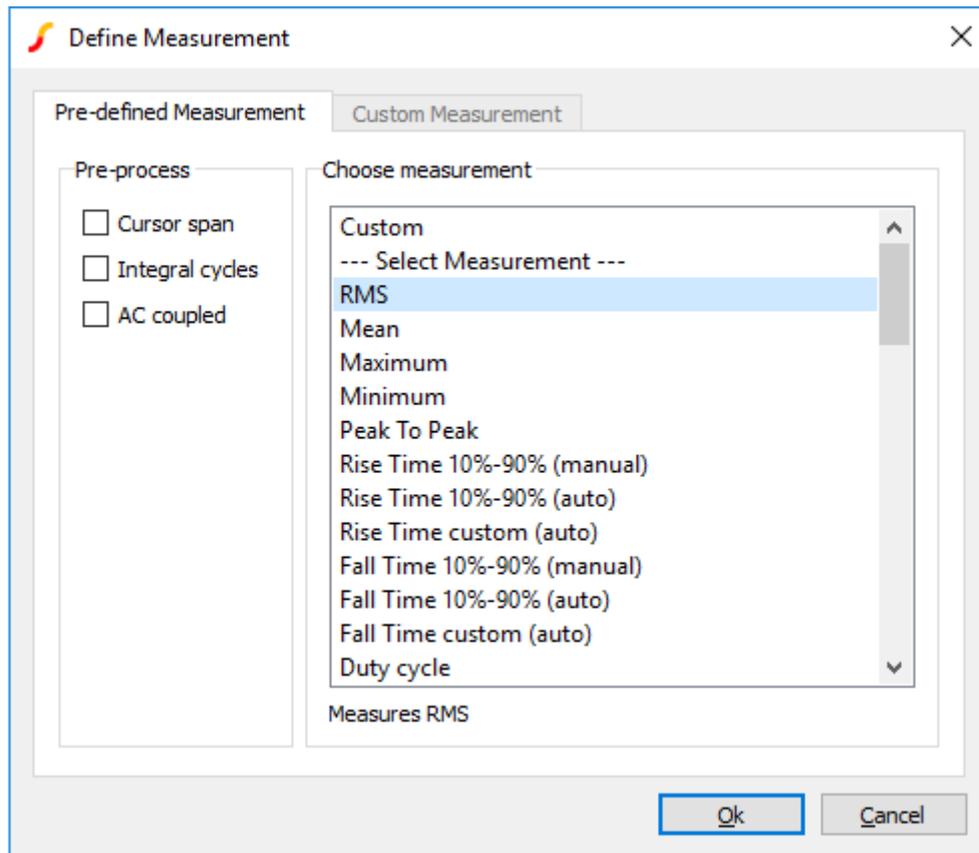
### 13.9.2 Available Measurements

A wide range of measurement functions are available. Select menu **Measure | More Functions...** to see the complete list. For more information see [“Using the Define Measurement GUI” on page 352](#).

### 13.9.3 Using the Define Measurement GUI

The Define Measurement GUI is a general purpose interface to the measurement system and provides access to all measurement functions along with a means to define custom measurements.

To open the Define Measurement GUI, select menu **Measure | More Functions...** . You will see the following dialog box:



## Choose measurement

Lists all available measurement functions. If cursors are not switched on, some of the functions will be greyed out. These are functions that require you to identify parts of the waveform to be measured. For example the manual rise and fall time measurements require you to mark points before and after the rising or falling edge of interest.

When you click on one of the measurements, some notes will appear at the bottom explaining the measurement and how to use it.

## Pre-process

Listed in the pre-process box are three operations that can optionally be performed on the waveform before the measurement function is applied. These are

- |                 |   |
|-----------------|---|
| Cursor span     | Truncates the waveform data to the span defined by the current positions of the cursors. In other words, the measurement is performed on the range defined by the cursor positions. |
| Integral cycles | Truncates the waveform data to an integral number of whole cycles. This is useful for measurements such as RMS which are only meaningful if applied to a whole number of cycles.    |
| AC coupled      | Offsets the data by the mean value. This is equivalent to 'ACcoupling' the data.  |

The above operations are performed in the order listed. So for example, the data is truncated to the cursor span before AC coupling.

## Custom Measurement

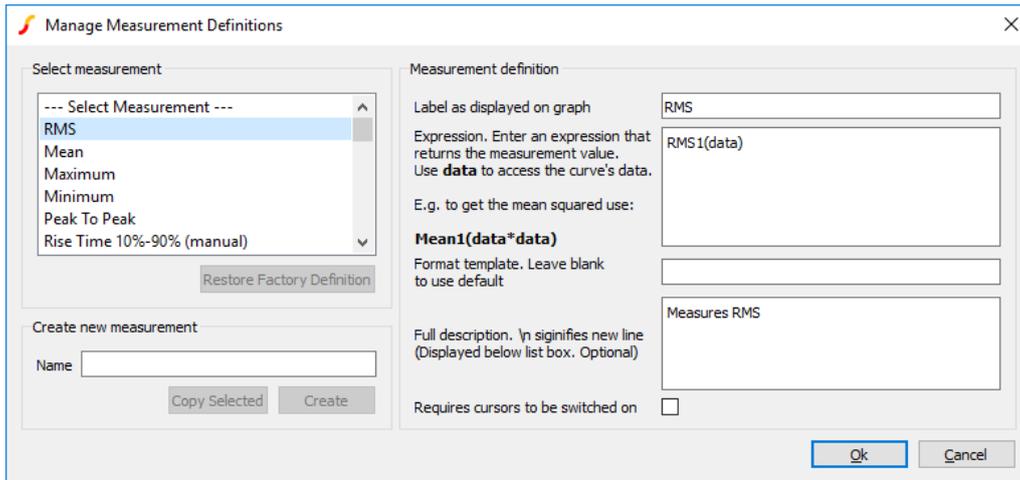
If you select the Custom entry in the Choose measurement list, the Custom measurement tab will be enabled.

The Custom measurement tab allows you to define your own measurement along with an option to add it to the list of pre-defined measurements. The following explains the entries in the Custom measurement tab.

Label as displayed on graph	This is the label that will appear alongside the measurement value in the graph legend panel. Usually, this would be literal text, but you may also enter a template string using special variables and script functions. See <a href="#">“Templates” on page 356</a> for details.
Expression	Expression to define measurement. Use the variable ‘data’ to access the data for the curve being measured. The expression must return a single value (i.e. a scalar). See <a href="#">“Goal Functions” on page 379</a> for details of functions that may be used to define measurement expressions.
Format template	Defines how the value will be displayed. If you leave this blank, a default will be used which will display the result of the expression along with its units if any. See <a href="#">“Templates” on page 356</a> for details.
Save definition to pre-defined measurements	If checked, the measurement definition will be saved to the list shown in pre-defined measurements. You can optionally enter some further details under Save definition. Note that the definition will not appear in the pre-defined list until the dialog is closed and reopened. Further management of custom measurement definitions can be made using the <a href="#">“Measurement Definitions Manager” on page 354</a> .
Short description	This is what will be displayed in the list box under Choose measurement in the Pre-defined Measurement tab.
Full description	This is what will be displayed below the list box when the item is selected.

### 13.9.4 Measurement Definitions Manager

Select menu **Measure | Manage Measurement Definitions**. This will open the Measurement Definitions Manager dialog box shown below:



The Measurement Definitions Manager allows you to edit both built in and custom measurement definitions.

### Select measurement

Select measurement you wish to edit from this list.

### Restore Factory Definition

This button will be enabled for any built-in definition that has been edited in some way. Press it to restore the definition to its original. For custom definitions, this button's label changes to **Delete**. Press it to delete the definition.

It is not possible to delete built-in definitions

### Create new measurement

You can create a completely new empty measurement or you can copy an existing one to edit. Enter a name then press **Create** to create a new empty definition. To copy an existing definition, select the definition under **Select measurement** then press **Copy selected**.

### Measurement definition

Define measurement. There are five entries:

Label as displayed on graph	This is the label that will appear alongside the measurement value in the graph legend panel. Usually, this would be literal text, but you may also enter a template string using special variables and script functions. See <a href="#">“Templates” on page 356</a> for details.
Expression	Expression to define measurement. Use the variable 'data' to access the data for the curve being measured. The expression must return a single value (i.e. a scalar). See <a href="#">“Goal Functions” on page 379</a> for details of functions that may be used to define measurement expressions.

Format template	Defines how the value will be displayed. If you leave this blank, a default will be used which will display the result of the expression along with its units if any. See <a href="#">“Templates” on page 356</a> for details.
Save definition to pre-defined measurements	If checked, the measurement definition will be saved to the list shown in pre-defined measurements. You can optionally enter some further details under Save definition. Note that the definition will not appear in the pre-defined list until the dialog is closed and reopened. Further management of custom measurement definitions can be made using the <a href="#">“Measurement Definitions Manager” on page 354</a> .
Full description	This is what will be displayed below the list box when the item is selected.
Requires cursors to be switched on	If checked, the measurement will be disabled unless graph cursors are enabled

## Templates

Both the graph label and Format template may be entered using a template containing special variables and expressions. The following is available:

<code>%yn%</code>	Where $n$ is a number from 1 to 5. $y$ -value returned by expression. The value returned by the expression may be a vector with up to 5 elements. <code>%y1%</code> returns element 1, <code>%y2%</code> returns element 2 etc.
<code>%xn%</code>	Where $n$ is a number from 1 to 5. $x$ -value returned by expression. The value returned by the expression may be a vector with up to 5 elements. <code>%x1%</code> returns the $x$ -value of element 1, <code>%x2%</code> returns the $x$ -value of element 2 etc. The $x$ values are the values used for the $x$ -axis. You should be aware that not all functions return $x$ -data.
<code>%uyn%</code>	The units of the $y$ values.
<code>%uxn%</code>	The units of the $x$ values.
<code>%%</code>	Literal <code>%</code> character.
<code>expression</code>	<code>expression</code> will be evaluated and substituted. <code>expression</code> may contain any valid and meaningful script function. For full details, see <i>Script Reference Manual/Function Reference</i> .

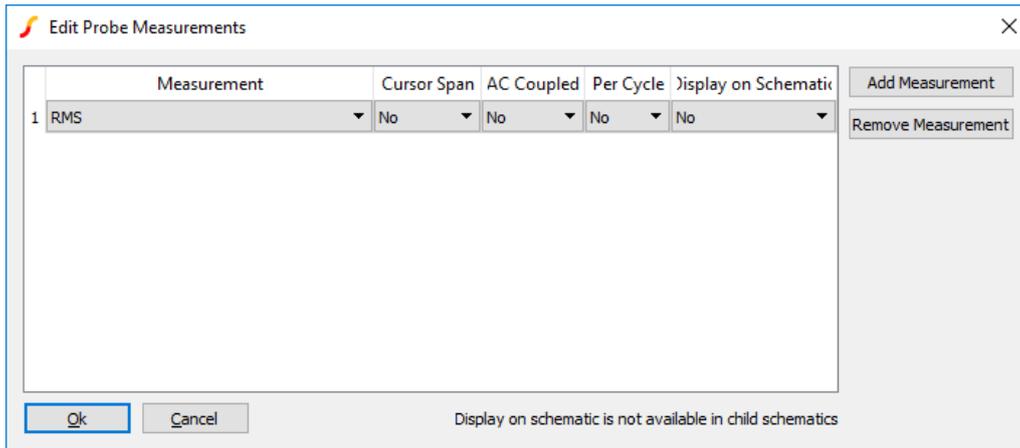
### 13.9.5 Repeating the Same Measurement

The menu Measure | Repeat Last Measurement will repeat the most recent measurement performed.

### 13.9.6 Applying Measurements to Fixed Probes

Measurements may be applied to fixed probes so that the measurement is performed automatically when the simulation is complete. To apply a measurement to a probe, proceeds as follows:

1. Select the probe then right click menu **Edit/Add Measurements...**
2. The following box should be displayed:



3. To begin with a single measurement is shown. Using the drop down box in the **Measurement** column, select the desired measurement type
4. You may also change some of the attributes:
  - Cursor Span specifies that the measurement will be calculated over the current cursor range. If cursors are not switched on, this will be ignored.
  - AC coupled specifies that the data will have its DC component removed before the measurement is made.
  - Per Cycle will apply an algorithm to detect whole numbers of cycles and apply the measurement over that range.
  - Display on schematic will additionally display the measurement result on the schematic using a label attached to the probe.
5. To add additional measurements, click the **Add Measurement** button. A new line will appear. There is no limit to the number of measurements that may be applied.
6. To remove a measurement, select the line to be removed then click **Remove Measurement**.

### 13.9.7 Notes on Curve Measurement Algorithms

Some of the measurements algorithms make some assumptions about the wave shape being analysed. These work well in most cases but are not fool-proof. The following notes describe how the algorithms work and what their limitations are.

All the measurement algorithms are implemented by internal scripts. The full source of these scripts can be found on the install CD (see [“Install CD”](#) on page 474).

#### ‘per Cycle’ and Frequency Measurements

These measurements assume that the curve being analysed is repetitive and of a fixed frequency. The results may not be very meaningful if the waveform is of varying frequency or is of a burst nature. The /cycle measurements calculate over as many whole cycles as possible.

Each of these measurements use an algorithm to determine the location of x-axis crossings of the waveform. The algorithm is quite sophisticated and works very reliably. The bulk of this algorithm is concerned with finding an optimum base line to use for x-axis crossings.

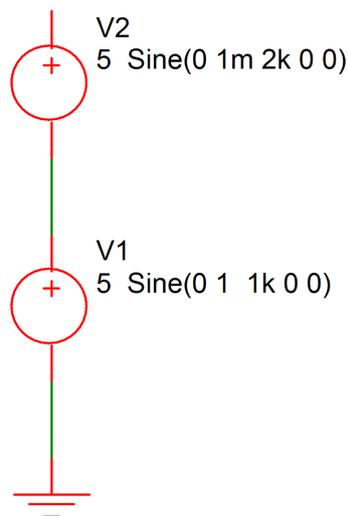
The per cycle measurements are useful when the simulated span does not cover a whole number of cycles. Measurements such as RMS on a repetitive waveform only have a useful meaning if calculated over a whole number of cycles. If the simulated span does cover a whole number of cycles, then the full version of the measurement will yield an accurate result.

## Rise and Fall Time, and Overshoot Measurements

These measurements have to determine the waveforms pulse peaks. A histogram method is used to do this. Flat areas of a waveform produce peaks on a histogram. The method is very reliable and is tolerant of a large number of typical pulse artefacts such as ringing and overshoot. For some wave-shapes, the pulse peaks are not well enough defined to give a reliable answer. In these cases the measurement will fail and an error will be reported.

## Distortion

This calculates residue after the fundamental has been removed using an FFT based method. This algorithm needs a reasonable number of cycles to obtain an accurate result. The frequency of the fundamental is displayed in the message window. Note that most frequency components between 0Hz and just before the second harmonic are excluded. The precision of the method can be tested by performing the measurement on a test circuit such as:



The signal on the pos side of V2 has 0.1% distortion. Use V1 as your main test source (assuming you are testing an amplifier) then after the simulation is complete, check that the distortion measurement of V2 is 0.1%. If it is inaccurate, you will need either to increase the number of measurement cycles or reduce the maximum time step or both. You can adjust the amplitude of V2 appropriately if the required resolution is greater or less than 0.1%.

Note, that in general, accuracies of better than around 1% will require tightening of the simulation tolerance parameters. In most cases just reducing RELTOL (relative tolerance) is sufficient. This can be done from the Options tab of the Choose Analysis Dialog (**Simulator | Choose Analysis...**). For a more detailed discussion on accuracy see the chapter “Convergence and Accuracy” in the *Simulator Reference Manual*.

## Frequency Response Calculations

These must find the passband for their calculations. Like rise and fall a histogram approach is used to find its approximate range and magnitude. Further processing is performed to find its exact magnitude.

Note that the algorithms allow a certain amount of ripple in the passband which will work in most cases but will fail if this in excess of about 3dB.

Note that the frequency response measurements are general purpose and are required to account for a wide variety of responses including those with both high and low pass elements as well as responses with band pass ripple. This requirement compromises accuracy in simpler cases. So, for example, to calculate the -3dB point of a low pass response that extends to DC, the 0dB point is taken to be a point midway between the start frequency and the frequency at which roll-off starts. A better location would be the start frequency but this would be inaccurate if there was a high pass roll off at low frequencies. Taking the middle point is a compromise which produces good - but not necessarily perfect - results in a wide range of cases. To increase accuracy in the case described above, start the analysis at a lower frequency, this will lower the frequency at which the 0dB reference is taken.

### 13.9.8 Plots from curves

Two plots can be made directly from selected curves. These are described below

#### FFT of Selected Curve

With a single curve selected, select menu **Plot | Fourier of Selected Curve**. A new graph sheet will be opened with the FFT of the curve displayed. To plot an FFT of the curve over the span defined by the cursor locations select menu **Plot | Fourier of Selected Curve (Cursor span)**.

#### Smoothed Curves

With a single curve selected, select one of the **Plot | More LP Filtered** menus. For **Plot | More LP Filtered - Custom TC** you will need to enter a time constant value. A new curve will be displayed which is a filtered version of the selected curve.

The above functions will still work if you don't select any curves. In this case you will be prompted for the curve on which to perform the operation.

This system uses a first order digital IIR filter to perform the filtering action.

## 13.10 Efficiency Calculator

### 13.10.1 Overview

Generating the Efficiency Calculator report involves three basic steps:

1. In the first step, you use the menu **Tools | Efficiency Calculator | 1. Add Power Probes to Schematic** to place a minimum of two probes – one on the input source and one on the output load. The algorithm then calculates the power delivery efficiency from the source(s) to the load(s).

The fixed power probes must be placed on the terminals of each symbol. It does not matter to which pin the probe is attached as long as no other device is attached to that pin.

The efficiency-calculation algorithm produces an additional section that details the loss breakdown for all probes, assuming that each power probe represents a loss element in the design; therefore, for

the most comprehensive report, place a fixed power probe on each lossy component on your schematic.

2. In the second step, you use the menu **Tools | Efficiency Calculator | 2. Define Inputs and Outputs** to specify which probes are assigned to inputs and which are assigned to output loads.  
Note: All power probes not assigned as inputs or outputs are considered losses.
3. In the third step, you use the menu **Tools | Efficiency Calculator | 3. Calculate Efficiency** to run the report which appears in a dialog and in the command shell window.

The efficiency-calculation algorithm reads the "Mean" measurements of the fixed power probes on the schematic and reports the efficiency to the command shell window. To calculate efficiency, the algorithm assumes the following:

- The circuit has entered into a steady state.
- The waveform viewer has an integral number of periodic cycles of data.

Although a SIMPLIS POP analysis always meets these two requirements, you must ensure that a SIMetrix simulation circuit has entered into a steady state. The efficiency-calculation algorithm uses numerical integration over the total number of simulation points; therefore, the simulation must have a high number of data points to ensure that the efficiency calculation is accurate.

To specify the number of SIMetrix simulation points, follow these steps:

1. From the Schematic Editor, select menu **Simulator | Choose Analysis...**
2. At the bottom of the Transient tab, click **Advanced Options...**
3. In the Advanced Options dialog, set the **Max time step** to 100ns or less. **Note:** In some converters, the value may be 1ns or less. Some experimentation may be necessary. If the efficiency data appears noisy, the most likely cause is an insufficient number of plot points.

To specify the number of SIMPLIS simulation points, follow these steps:

1. From the Schematic Editor, select menu **Simulator | Choose Analysis...**
2. Click on the **Transient** tab.
3. Set the **Number of plot points** to at least 10k. **Note:** In some converters, the value may be 100k or greater. Some experimentation may be necessary. If the efficiency data appears noisy, the most likely cause is an insufficient number of plot points.

### 13.10.2 Step1: Place Power Probes

The menu **Tools | Efficiency Calculator | 1. Add Power Probes to Schematic**, allows you to place multiple fixed power probes on the current schematic. To place the power probes on your schematic, close the hint window and then follow these steps:

1. Position the power probe circle on a component pin, and then click the left mouse button. **Note:** To zoom in while placing the power probes, hold down the **Control** key and scroll the mouse wheel to pan up and down. To pan left and right, hold down the **Alt** key while you scroll the mouse wheel.
2. Repeat the previous step until you have placed all the power probes.
3. To stop placing power probes, click the right mouse button.

Note: You can also place and delete power probes on the schematic with the following menus:

- **Tools | Efficiency Calculator | Add Power Probes to Schematic**
- **Tools | Efficiency Calculator | Delete all Power Probes**

### 13.10.3 Step 2: Define Input and Output probes

The menu, **Tools | Efficiency Calculator | 2. Define Inputs and Outputs...**, allows you to specify which probes are assigned to input sources and which are assigned to output loads. The overall Efficiency is calculated from the Input Probes and Output Probes selected at the bottom of the Efficiency Calculator dialog. The probes remaining in the upper box are considered lossy elements in the design. You can use any of the following methods to move probes from the All Power Probes box to either the Input Probes or Output Probes box.

- Click to select a single probe, and then click the appropriate **Add to ...** button
- Hold down the **Control** key, click on multiple probes, and then click an **Add to...** button
- With your mouse, draw a box around adjacent probes and then click an **Add to...** button

To define the input and output probes, follow these steps:

1. In the upper box, select the probe(s) to assign to the input sources and then click **Add to Inputs**
2. Select the probe(s) to assign to the output loads and then click **Add to Outputs**
3. Click **Ok**

### 13.10.4 Step 3: Calculate Efficiency

The menu, **Tools | Efficiency Calculator | 3. Calculate Efficiency**, generates the efficiency report from existing data and displays the report in a dialog and in the command shell window.

**Note:** If no data exists, you will be prompted to select input and output probes before the report can run. After assigning the input and output probes, you must again select menu **Tools | Efficiency Calculator | 3. Calculate Efficiency**.

## 13.11 Graph Zooming and Scrolling

### 13.11.1 Zooming with the Mouse

To zoom in on a portion of a graph, place the cursor at the top left of the area you wish to view, press and hold the left mouse key then move cursor to bottom right of area and release left key. The axes limits will be modified appropriately.

If the graph has multiple stacked grids, you should be sure that the first left click is within the area of the grid you wish to zoom. You will notice a thin grey line separating each grid. You should start the mouse drag within the grey lines for the chosen axis.

You can zoom just the x axis by dragging a horizontal line across an area outside any grid - e.g. right at the bottom of the window below the lower x axis line.

To view whole graph again select the graph pop-up **Zoom | Full** or toolbar button.

### 13.11.2 Zooming with the Keyboard

F12 to zoom out

shift-F12 to zoom in

HOME returns graph to full view. (Same as graph popup **Zoom | Full**)

### 13.11.3 Recovering an Earlier Zoom

Press the **Undo Zoom** toolbar button to recover earlier zoom or scroll positions.

### 13.11.4 Scrolling with the Keyboard

up, down, left and right cursor keys will scroll the active graph.

### 13.11.5 Zooming with Multiple Grids

If the zoom box covers multiple grids, only the x-axes of those grids will be altered.

### 13.11.6 Adding new Curves to a Zoomed Grid

If you add a new curve to a graph which has been zoomed, the axes limits will not change to accommodate that curve; if the new curve does not lie within the zoomed area you will not see it. Selecting graph popup **Zoom | Full** or pressing the HOME key restores the graph to auto-scaling and the limits will always adjust so that all curves are visible even ones subsequently added.

### 13.11.7 Zoom to Fit Y-axis

To zoom in the y-axis only to fit the displayed x-axis, press the **Fit Height** toolbar button:

## 13.12 Annotating a Graph

A number of objects are available to annotate graphs for documentation purposes. These are:

- **Curve Marker.** A single arrow, line and item of text to identify a curve or feature of a curve
- **Legend Box.** Box of text that lists all the names of curves currently displayed.
- **Text Box.** Box containing text message.
- **Free Text.** Similar to text box but without border and background.
- **Caption.** As free text but designed for single line heading.

### 13.12.1 Curve Markers

#### Placing

To place a curve marker, select menu **Annotate | Add Curve Marker**. A single curve marker should appear in the right hand margin of the graph.

#### Moving

To move it, place the mouse cursor at the arrow head - you should see the cursor shape change to a four pointed arrow - then left click and drag to your desired location. When you release the marker it will snap to the nearest curve.

## Moving Label

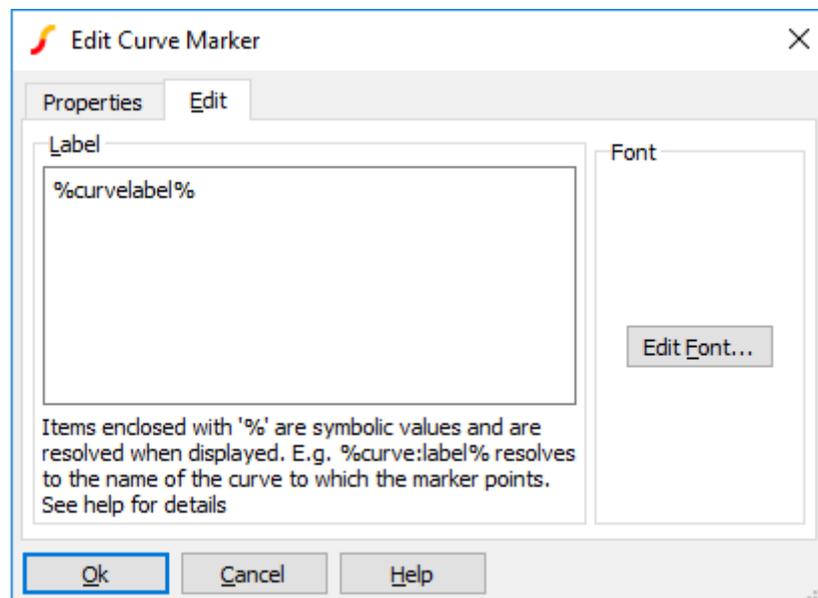
To move the text label alone, place the mouse cursor to lie within the text, then left click and drag. You will notice the alignment of the text with respect to the arrowed line change as you move the text around the arrow. You can fix a particular alignment if preferred by changing the marker's properties. See below.

## Deleting

First select the marker by a single left click in the text. The text should change colour to blue. Now press delete key or menu **Annotate | Delete Selected Object**.

## Editing Properties

Double click the marker's label or select then menu **Annotate | Edit Selected Object**. The following dialog will open:



**Label** Text of the marker's label. `%curve:label%` automatically resolves to the curve's label. If the curve name is edited with menu **Curves | Rename curve** this value will reflect the change. You can of course enter any text in this box.

You can also use expressions in the same manner as for cursor dimensions. See *Label* in ["Editing Style or/and Format of Cursor Dimension"](#) on page 350.

**Text Alignment** This is how the label is aligned to the arrowed line. If set to automatic the alignment will be chosen to be the most appropriate for the relative position of the label and the arrowhead. Uncheck automatic and select from the list to fix at a particular alignment.

**Font** Press **Edit Font...** to change font for text.

In the Properties panel you can do the following:

**Snap To Curve** You can switch off the action that causes curve markers to always snap to a curve. Select Properties tab then double click on SnapToCurve item. Select Off. You will now be able to move the curve marker to any location.

### 13.12.2 Legend Box

#### Placing

Select menu **Annotate | Add Legend Box**. A box listing all the curve names will appear at the top left of the graph.

#### Moving

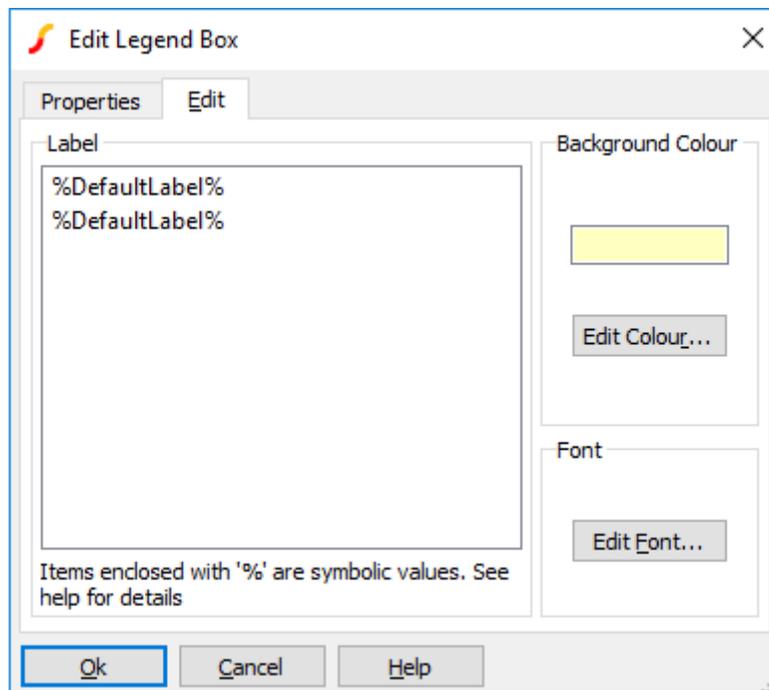
Place cursor inside the box and drag to new location.

#### Resizing

You can alter the maximum height of the box by placing the mouse cursor on it's bottom edge and dragging. The text in the box will automatically reposition to comply with the new maximum height.

#### Editing Properties

Double click on the box or select then menu **Annotate | Edit Selected Object**. The following dialog will be opened:



The dialog box options are:

<b>Label</b>	Lists each label in the box. These are usually <code>%DefaultLabel%</code> which resolves to the name of the referenced curve. To edit, double click on the desired item. You can also enter the symbols <code>%X1%</code> and <code>%Y1%</code> which represent the x and y coordinates of the marker respectively. These can be combined with other text in any suitable manner. For example: 'Voltage @ <code>%X1%</code> S = <code>%Y1%</code> ' might resolve to something like 'Voltage at 10u = 2.345'. The values of <code>%X1%</code> and <code>%Y1%</code> will automatically update if you move the marker.  You can also use expressions in the same manner as for cursor dimensions. See <a href="#">Label</a> in "Editing Style or/and Format of Cursor Dimension" on <a href="#">page 350</a> .
Background Colour	Select button Edit Colour... to change background colour. To change the default colour select menu <b>File   Options   Colour...</b> then select item Text Box. Edit colour as required.
Font	Select button Edit Font... to change font. To change the default font select menu <b>File   Options   Font...</b> then select item Legend Box. Edit font as required.

### 13.12.3 Text Box

#### Placing

Select menu **Annotate | Text Box**. Enter required text then Ok. You can use the symbolic constants `%date%`, `%time%`, and `%version%` to represent creation date, creation time and the product version respectively.

#### Moving

Place cursor inside the box and drag to new location.

#### Editing Properties

Double click on the box or select then menu **Annotate | Edit Selected Object**. A dialog like the one shown for legend boxes (see "Placing" on [page 365](#)) will be displayed.

Note when editing the label, you can use the symbolic constants as detailed in above.

### 13.12.4 Caption and Free Text

The Caption and Free Text objects are essentially the same, the only difference is their initial font size and position.

#### Placing

Select menu **Annotate | Caption** or **Annotate | Free Text**. Enter required text then Ok. You can use the symbolic constants `%date%`, `%time%`, and `%version%` to represent creation date, creation time and the product version respectively.

#### Moving

Place cursor inside the box and drag to new location.

## Editing Properties

Double click on the box or select then menu **Annotate | Edit Selected Object**. This will open a dialog similar to the one shown for curve markers but without the *Automatic* option for text alignment.

### 13.12.5 Graph Symbolic Values

Most graph objects have one or more label properties that can be used to display text on the graph. As well as literal text, these label properties may also use symbolic values enclosed with ‘%’. These symbolic values return values of other properties belonging to the object. For example curve marker objects have a property called X1 which is always set to the x-location of the curve to which it is attached. So %X1% in a curve marker label will return the x-location allowing it to be displayed on the graph. The X1 property is updated every time the curve marker is moved; the label value is re-evaluated every time the graph is repainted. (Sometimes it is necessary to force a repaint to get labels with symbolic values or/and expressions to update. You can do this by moving another window over the graph or adjusting the size of the window slightly)

Some properties return the ID of another graph object. For example the CURVE property returns the ID of the curve to which it is attached. These can be used to access properties of the referenced object. This is done by appending with a ‘:’ followed by the referenced object’s property name. For example %curve:label% returns the LABEL property of the curve attached to the curve marker.

This indirect access to graph object properties can be nested to any level although there is probably no good reason for any more than two levels. %curve:xaxis:label%, for example has two levels; %curve% returns the ID of a curve, then %curve:xaxis% returns the id of an axis then %curve:xaxis:label% returns the LABEL property belonging to the axis.

Full documentation is available in *Script Reference Manual/Applications/Graph Objects*. This lists the available objects and their property names. There is also a sub-heading titled Symbolic Values that explains the above.

However, deducing all the different possibilities for symbolic values, especially the indirect values, requires some effort. For this reason, the following table has been compiled which lists a range of complete symbolic values that are meaningful for use in labels for various objects. This list is not exhaustive, but probably has everything that is useable.

Note that the symbolic variable names, like everything in SIMetrix, are not case sensitive.

Variable	Description	Can use with
%curve:label%	Curve’s label	Curve marker
%curve:shortlabel%	Curve’s label without the groupname suffix that is sometimes displayed	Curve marker
%curve:xunit%	Curve’s x-axis units	Curve marker
%curve:yunit%	Curve’s y-axis units	Curve marker
%x1%	Curve’s x-value at curve marker	Curve marker
%y1%	Curve’s y-value at curve marker	Curve marker
%curve:xaxis:label%	Curve’s x-axis label	Curve marker
%curve:yaxis:label%	Curve’s y-axis label	Curve marker
%curve:measurements%	Any measurements assigned to the curve	Curve marker
%graph:maincursor:x1%	x-position of main cursor	Anything
%graph:maincursor:y1%	y-position of main cursor	Anything

Variable	Description	Can use with
<code>%graph:refcursor:x1%</code>	x-position of ref cursor	Anything
<code>%graph:refcursor:y1%</code>	y-position of ref cursor	Anything
<code>%graph:grouptitle%</code>	Title of initial data group. This is actually the netlist title (first line) and for schematic simulations will be the full path of the schematic.	Anything
<code>%graph:sourcegroup%</code>	Data group name that was current when first curve added to graph. E.g. 'tran1', 'dc5' etc.	Anything
<code>%curve1:label%</code>	Label for curve attached to crosshair 1	Dimension
<code>%curve2:label%</code>	Label for curve attached to crosshair 2	Dimension
<code>%curve1:shortlabel%</code>	Curve1's label without the groupname suffix that is sometimes displayed	Dimension
<code>%curve2:shortlabel%</code>	Curve2's label without the groupname suffix that is sometimes displayed	Dimension
<code>%curve1:xunit%</code>	Curve1's x-axis units	Dimension
<code>%curve2:xunit%</code>	Curve2's x-axis units	Dimension
<code>%curve1:yunit%</code>	Curve1's y-axis units	Dimension
<code>%curve2:yunit%</code>	Curve2's y-axis units	Dimension
<code>%date%</code>	Date when object created	Textbox, free text, caption, legend box
<code>%time%</code>	Time when object created	Textbox, free text, caption, legend box
<code>%version%</code>	Product name and version	Textbox, free text, caption, legend box

## Expressions

Graph object labels may contain expressions enclosed in curly braces. These will be evaluated and the result of the evaluation replaces the complete expression and curly braces. Any script function may be used although only a subset are applicable.

The function 'cv()' is particularly useful. cv() returns the data for a curve and you can use this with functions that return a scalar from a vector to attach measurements to curve markers or cursors. Use `%curve%` as the argument for cv(), i.e.

```
cv (%curve%)
```

For example, this will return the RMS value for the curve attached to a curve marker:

```
{RMS1 (cv (%curve%)) }
```

For crosshair dimension objects (the cursor dimensions) use `%curve1%` or `%curve2%` instead of `%curve%`.

The `Truncate()` function is useful if you want to display a measurement applied to a range marked out by the cursors. So the following example will return the RMS value of the curve attached to a curve marker between the range marked out by the cursors.

```
{RMS1 (Truncate (cv (%curve%),
%graph:refcursor:x1%, %graph:maincursor:x1%)) }
```

You can also use string functions. For example, `%graph:title%` usually returns the pathname of the schematic. (This is not guaranteed - but this will always be the case if the schematic has been saved and was run using the regular menus). You can use the `SplitPath` function to obtain just the file name. E.g.:

```
{ (splitpath (`%Graph:GroupTitle%')) [2] }
```

You can use the above in any object including free text, text boxes and captions. (Captions are identical to free text, they just have a different default position and font).

The numeric functions above will usually result in a display with more significant digits than desirable. To format the result with less accuracy, use the `FormatNumber()` function. For example:

```
{FormatNumber (RMS1 (cv (%curve%)), 5) }
```

Will display the result to 5 digits.

## 13.13 Exporting/Importing Graphs and Data

### 13.13.1 Clipboard Operations

#### Overview

SIMetrix offers facilities to copy both graph data and the graph's graphical image to the system clipboard. This provides the ability to export simulation results to other applications. The data - for example - may be exported to a spreadsheet application for custom processing, while the graphical image may be exported to a word processor for the preparation of documents.

SIMetrix may also import data in a tabulated ASCII format. This feature may be used to display data from a spreadsheet allowing, for example, a comparison between measured and simulated data.

#### Copy Data to the Clipboard

1. Select the graphs you wish to export
2. Select the menu **Edit | Copy ASCII Data**

The data will be copied in a tabulated ASCII format. The first line will contain the names of the curves, while the remaining lines will contain the curves' data arranged in columns

#### Copying Graphics to the Clipboard

There are three different ways a graph can be copied to the clipboard. Use the menus under **Edit | Copy Graphics**. These are detailed below:

<b>Colour</b>	Copies graph to clipboard in full colour. The curve legends identify the curves using coloured squares similar to how the graph is displayed on the screen.
---------------	---

<b>Monochrome</b>	Copies graph to clipboard in monochrome. Curves are distinguished using varying markers and line styles. Curve legends distinguish curves with a straight line example
<b>Colour with markers</b>	Copies graph to clipboard in full colour but also differentiates curves using markers and line styles. Curve legends distinguish curves with a straight line example.

## Paste Data from the Clipboard

SIMetrix can plot curves using tabulated ASCII data from the clipboard. The format is the same as used for exporting data. See [“Copy Data to the Clipboard” on page 368](#) for more details.

### 13.13.2 Exporting Graphics

You may export schematic graphics to other applications such as word processors or drawing programs. You can do this via the clipboard (see [“Copying Graphics to the Clipboard” on page 368](#)) or by writing out to a file. To export waveform graphics to a file, select the graph menu **File | Save Picture...** then select the format of your choice using the **Save as type:** drop down box. The choices are:

1. **Windows Meta File (.EMF and .WMF)**. Nearly all windows applications that support graphics import will accept this format. Note that this is a scalable format and therefore suitable for high resolution printing.
2. **Scalable Vector Graphics (.svg)**. Like EMF and WMF, this is a scalable format. It is an open format and supported by programs on all platforms whereas EMF and WMF is only available on Windows.
3. **Bitmap - default image size (.png, .jpg, .bmp)**. These are available on all platforms, are widely supported by graphics applications but these are not scalable formats and so do not offer good quality when printed using high resolution printers. PNG is the default format if you do not choose a file extension and generally this format works well for schematics and graphs. To choose JPG (JPEG format) or BMP (windows bitmap format) you must explicitly enter .jpg or .bmp file extensions respectively. With this option the image size will match the image size currently displayed on screen. If you wish to specify a different image size, use next option.
4. **Bitmap - specify image size (.png, .jpg, .bmp)**. As 3 above but you must explicitly define the image resolution in pixels. You will be prompted for this when you close the file selection dialog box.

### 13.13.3 Data Import and Export

SIMetrix provides the capability to export simulation data to a file in text form and also to import data from a file in text form. This makes it possible to process simulation data using another application such as a spreadsheet or custom program. The operations described in this section handle data independent of the graph system. There are also ways to export plotted data from a graph and also directly plot external data. For information on these topics refer to [“Copying Data to the Clipboard” on page 368](#) and [“Paste Data from the Clipboard” on page 369](#)

SIMetrix may also import data in SPICE3 raw file format and CSDF format. Some other simulation products can output in one or both of these formats.

### Importing SPICE3 Raw and CSDF Files

1. Select menu **Simulator | Load Simulation Data...**

2. In Files of type select SPICE3 Raw Files or CSDF Files as required.
3. Select file to import.

SIMetrix will read the entire file and write its data out to a temporary .sxdat file in the same way as it does when saving its own simulation data. The data read from the raw file is buffered in RAM in order to maximise the efficiency of the saved data. SIMetrix will use up to 10% of system RAM for this purpose.

Note that this feature is not available with SIMetrix/SIMPLIS Elements.

## Importing Tabulated ASCII Data

SIMetrix can import data in a tabulated ASCII format allowing the display of data created by a spreadsheet program. There is a no menu for this, but this can be done using the command “OpenGroup” on page 395 with the /text switch. E.g. at the command line type:

```
OpenGroup /text data.txt
```

This will read in the file data.txt and create a new group called text*n*. See “Data Files Text Format” on page 371 for details of format.

Note that if you create the file using another program such as a spreadsheet, the above command may fail if the file is still open in the other application. Closing the file in the other application will resolve this.

## Exporting SPICE3 Raw Files

SIMetrix can export all simulation data to a SPICE3 raw file. This format may be accepted by third party waveform viewers.

To export a SPICE3 raw file, proceed as follows:

1. Select menu **File | Data | Save...**
2. Under Save as type: choose SPICE3 Raw Files.

Note that various applications use slightly different variants of this format. By default, SIMetrix outputs the data in a form that is the same as the standard unmodified SPICE3 program. This can be modified using the option setting “ExportRawFormat”. Use the Set command to set this value. See “Set” on page 396 for details. Set this value to ‘spice3’, ‘spectre’ or ‘other’.

## Exporting Data

To export data, use the command Show (see “Show” on page 396) with the /file switch. E.g:

```
Show /file data.txt vout r1_p q1#c
```

will output to data.txt the vectors vout, r1\_p, and q1#c. The values will be output in a form compatible with OpenGroup /text. **Vector Names** In the above example the vector names are vout, r1\_p and q1#c. If you simulate a schematic, the names used for voltage signals are the same as the node names in the netlist which in turn are assigned by the schematic’s netlist generator. To find out what these names are, move the mouse cursor over the node of interest on the schematic. You should see the node name and therefore the vector name in the status box in the form “NET=???”. To find the current name, place the mouse cursor on the device pin of interest and press control-P.

## Launching Other Applications

Data import and export makes it possible to process simulation data using other applications. SIMetrix has a facility to launch other programs using the Shell command. You could therefore write a script to export data, process it with your own program then read the processed data back in for plotting. To do this you must specify the /wait switch for the Shell command to force SIMetrix to wait until the external application has finished. E.g.

```
Shell /wait procddata.exe
```

will launch the program procddata.exe and will not return until procddata.exe has closed.

## Data Files Text Format

SIMetrix has the ability to read in data in text form using the OpenGroup command (see “OpenGroup” on page 395). This makes it possible to use SIMetrix to graph data generated by other applications such as a spreadsheet. This can be useful to compare simulated and measured results.

There are two alternative formats. The first is simply a series of values separated by white space. This will be read in as a single vector with a reference equal to its index.

The second format is as follows:

A text data file may contain any number of blocks. Each block has a header followed by a list of datapoints. The header and each datapoint must be on one line.

The header is of the form: reference\_name ydata1\_name [ ydata2\_name ... ]

Each datapoint must be of the form: reference\_value ydata1\_value [ ydata2\_value ... ]

The number of entries in each datapoint must correspond to the number of entries in the header. The reference is the x data (e.g. time or frequency).

## Data Files Text Format - Example

```
Time      Voltage1 Voltage2
0         14.5396  14.6916
1e-09    14.5397  14.6917
2e-09    14.5398  14.6917
4e-09    14.54     14.6917
8e-09    14.5408  14.6911
1.6e-08  14.5439  14.688
3.2e-08  14.5555  14.6766
6.4e-08  14.5909  14.641
1e-07    14.6404  14.5905
1.064e-07 14.6483  14.5821
```

If the above was read in as a text file (using OpenGroup /text), a new group called textn where n is a number would be generated. The group would contain three vectors called time, Voltage1 and Voltage2. The vectors Voltage1 and Voltage2 would have a reference of Time. Time itself would not have a reference.

To read in complex values, enclose the real and imaginary parts in parentheses and separate with a comma. E.g:

```
Frequency :VOUT
1000      (-5.94260997, 0.002837811)
1004.61579 (-5.94260997, 0.00285091)
1009.252886 (-5.94260996, 0.002864069)
1013.911386 (-5.94260995, 0.002877289)
```

```

1018.591388 (-5.94260994, 0.00289057)
1023.292992 (-5.94260993, 0.002903912)
1028.016298 (-5.94260992, 0.002917316)
1032.761406 (-5.94260991, 0.002930782)
1037.528416 (-5.9426099, 0.00294431)
1042.317429 (-5.94260989, 0.0029579)
1047.128548 (-5.94260988, 0.002971553)

```

## 13.14 Saving Graphs

You can save a graph complete with all its curves, cursor settings and annotations to a binary file for later retrieval. Note that all the graph data is stored not just that needed for the current view. If a long run was needed to create the graph, the file could be quite large.

### 13.14.1 Saving

Select graph menu **File | Save Graph As...** to save a graph that has never been saved before. To update a saved graph use graph menu **File | Save Graph**

### 13.14.2 Restoring

Select menu **File | Open...** then select **Graph Files** in the file type drop down box.

## 13.15 Saving Data

### 13.15.1 Saving the Data of a Simulation

The simulator usually saves all its data to a binary file stored in a temporary location. This data will eventually get deleted. To save this data permanently, select menu **File | Data | Save....** You will be offered two options:

#### Move existing data file to new location

This will move the data file to location that you specify and thus change its status from temporary to permanent. As long as the new location is in the same volume (=disk partition) as the original location, this operation will be very quick. However, if the data is from the most recent simulation, SIMetrix needs to 'unhook' it in order to be able to move the file. This will make it impossible to resume the simulation (if paused) or restart the simulation (transient only).

Note that if you specify a location on a different volume as the original data, then the file's data has to be copied and for large data files, this will take a long time.

#### Make new copy

This makes a fresh copy of the data. This option does not suffer from the drawbacks of moving the file but if the data file is large can take a very long time.

### 13.15.2 Restoring Simulation Data

Select menu **File | Data | Load...** Navigate to a directory where you have previously saved data files.

You can also reload data from temporary files using menu **File | Data | Load Temporary Data...**

Whether or not there will be any files available to opened depends on the temporary data file delete options. See “[Graph/Probe/Data Analysis](#)” on page 447 for information about these options.

The error “The process cannot access the file because it is being used by another process” means that the temporary data file is still in use. Unless the file is in use by another instance of SIMetrix you will be able to use its data by selecting its associated group. Use menu **Simulator | Manage Data Groups...**

## 13.16 Viewing DC Operating Point Results

### 13.16.1 Schematic Annotation

You can annotate the schematic with the results of a DC operating point analysis. This requires special markers to be placed on the schematic. You can instruct SIMetrix to place markers at every node or you can place them manually.

To place a voltage marker manually use the schematic popup: **Bias Annotation | Place Marker** or use control-M. The text displaying the value will be placed on the *sharp* side of the marker which to start with points up. If you are placing the marker on a vertical wire you might wish the text to be on one side. To do this, rotate the marker before placing by pressing the rotate toolbar button or the **F5** key.

To place a current marker use the menu **Place | Bias Annotation | Place Current Marker**.

To place markers as all nodes select **Bias Annotation | Auto Place Markers**. This does however clutter up the schematic and you may prefer to place them manually.

To display the values select **Bias Annotation | Update Values**. These values are automatically updated after each simulation run.

The menu **Bias Annotation | Delete Markers** deletes all the markers and **Bias Annotation | Hide Values** removes the text but leaves the markers in place.

### 13.16.2 Displaying Device Operating Point Info

The menu **Bias Annotation | Display Device Bias Info** will display in the message window the node voltages, pin currents and total power dissipation for the selected schematic part. Note that power dissipation is calculated from the node voltages and currents.

### 13.16.3 List File Data

A great deal of information about each device in the circuit can be obtained from the list file. Use menu **Simulator | Open List File** to see it. Also see the Simulator Reference Manual for more information about the list file.

### 13.16.4 Other Methods of Obtaining Bias Data

You can also display a voltage or current in the command shell without placing any part on the schematic. For voltages, place the mouse cursor over the point of interest and press control-N. For currents, place the cursor over the part pin and press control-I.

### 13.16.5 Bias Annotation in SIMPLIS

The above apply to operation in both SIMetrix and SIMPLIS modes. When in SIMPLIS mode the dc values displayed represent the results at time=0. For AC analysis this will be the time=0 value for its associated POP analysis.

### 13.16.6 Bias Annotation Display Precision

By default, bias annotation values are displayed with a precision of 6 digits. To change this, select menu **File | Options | General...** then edit the value in box Bias Annotation Precision in the Schematic sheet.

### 13.16.7 Bias Annotation and Long Transient Runs

If you are running a long transient analysis and plan to use bias annotation extensively, you might like to set a simulator option that will make this process more efficient. The simulator option is:

```
.OPTIONS FORCETRANOPGROUP
```

This forces a separate data group and separate data file to be created for the transient analysis bias point data. Unless tstart>0 bias point data is usually taken from t=0 values. The problem with this approach is that to view a single value, the entire vector has to be loaded from the data file to memory. This isn't a problem if the run is only a 100 points or so, but could be a problem if it was 100,000 points. It can take a long time to load that amount of data. By specifying this option, the bias point data is stored separately and only a single value needs to be read from the file. This is much more efficient.

## 13.17 Performance Analysis and Histograms

### 13.17.1 Overview

When running multi-step analyses which generate multiple curves, it is often useful to be able to plot some characteristic of each curve against the stepped value. For example, suppose you wished to investigate the load response of a power supply circuit and wanted to plot the fall in output voltage vs transient current load. To do this you would set up a transient analysis to repeat a number of times with a varying load current. (See [“Multi-step Analyses”](#) on page 259 to learn how to do this). After the run is complete you can plot a complete set of curves, take cursor measurements and manually produce a plot of voltage drop vs. load current. This is of course is quite a time consuming and error prone activity.

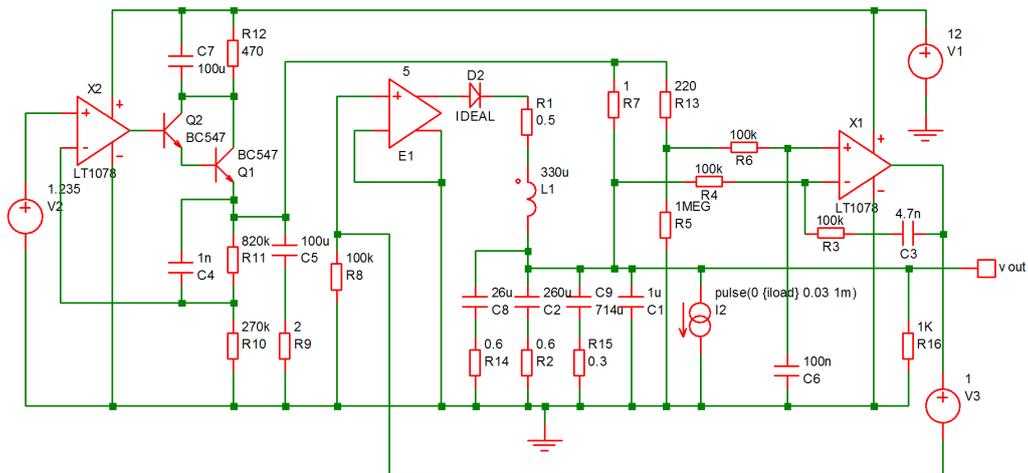
Fortunately, SIMetrix has a means of automating this procedure. A range of functions - sometimes known as *goal functions* - are available that perform a computation on a complete curve to create a single value. By applying one or a combination of these functions on the results of a multi-step analysis, a curve of the goal function versus the stepped variable may be created.

This feature is especially useful for Monte Carlo analysis in which case you would most likely wish to plot a histogram.

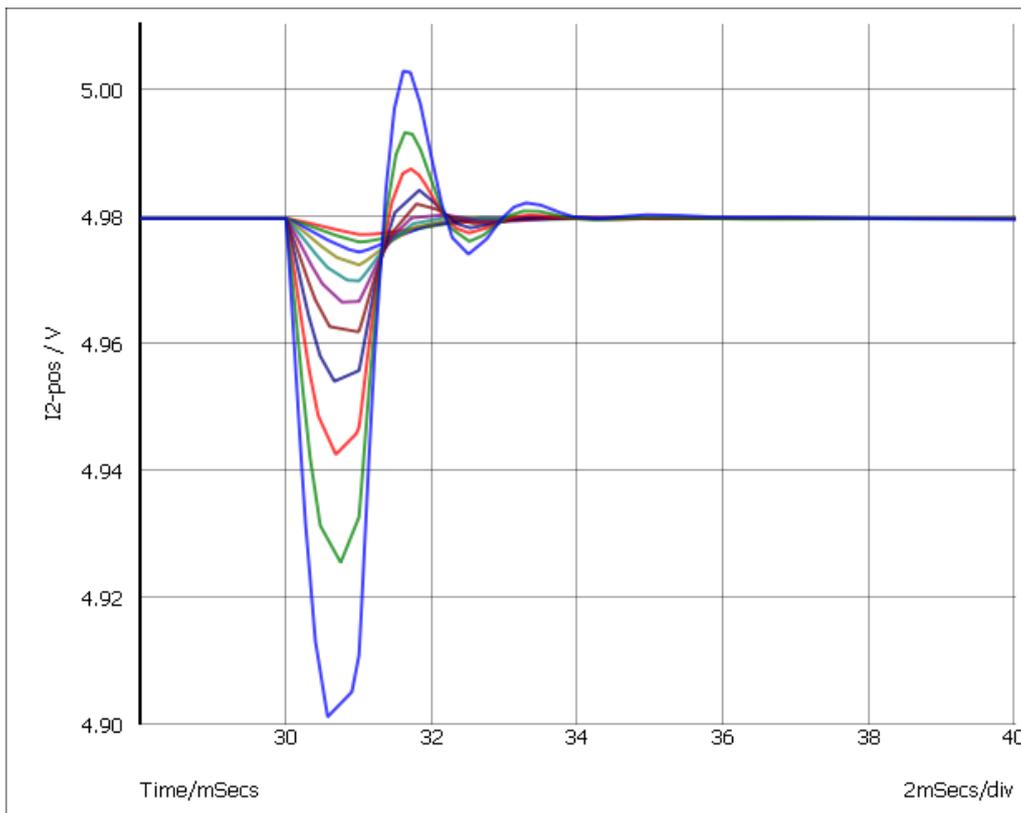
We start with an example and in fact it is a power supply whose load response we wish to investigate.

### 13.17.2 Example

The following circuit is a model of a hybrid linear-switching 5V PSU. See [Work/Examples/HybridPSU/5vpsu\\_v1.sxsch](#)



I2 provides a current that is switched on for 1mS after a short delay. A multi-step analysis is set up so that the load current is varied from 10mA to 1A. The output for all runs is:



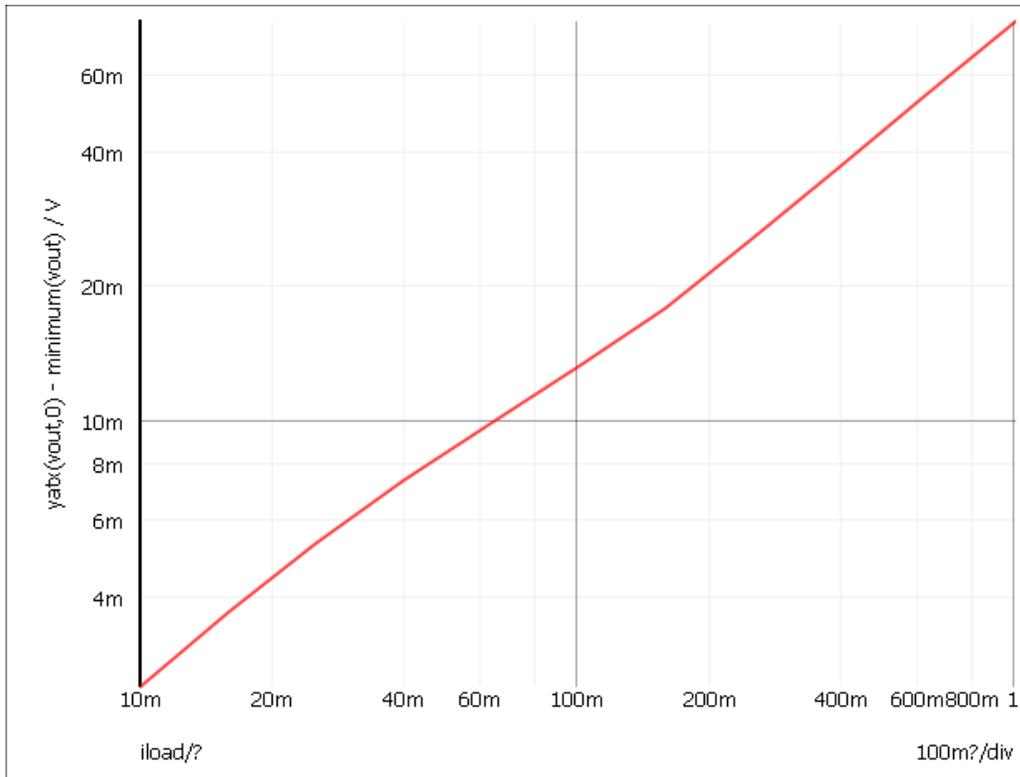
We will now plot the a graph of the voltage drop vs the load current. This is the procedure:

1. Select menu **Probe | Performance Analysis...**
2. You will see a dialog box very similar to that shown in [“Plotting an Arbitrary Expression” on page 331](#). In the expression box you must enter an expression that resolves to a single value for each

curve. For this example we use:

```
yatx(vout, 0) - minimum(vout)
```

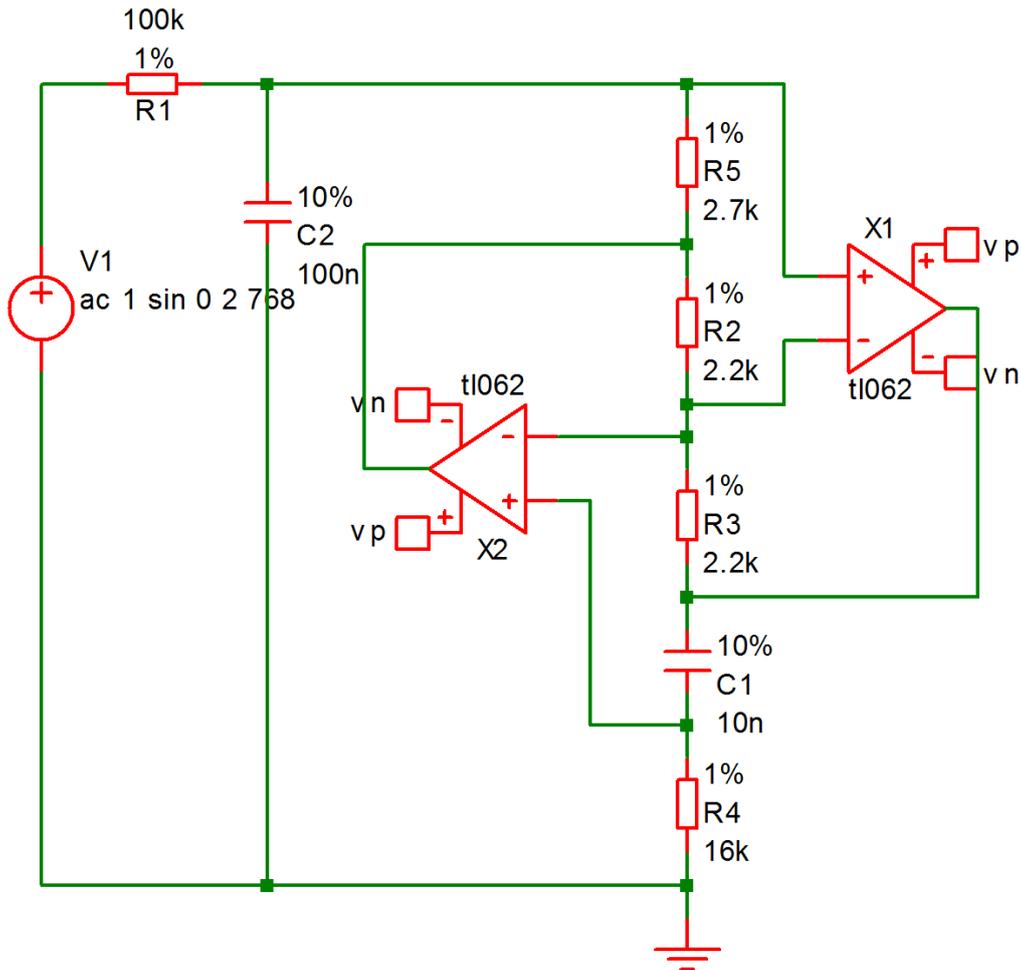
yatx(vout, 0) returns the value of vout when time=0. minimum(vout) returns the minimum value found on the curve. The end result is the drop in voltage when the load pulse occurs. Click **Ok** and the following curve should appear:



### 13.17.3 Histograms

The procedure for histograms is the same except that you should use the menu **Probe | Histogram...** instead. Here is another example.

This is a design for an active band-pass filter using the simulated inductor method. See Examples/MonteCarlo/768Hz\_bandpass.sxsch. We want to plot a histogram of the centre frequency of the filter.



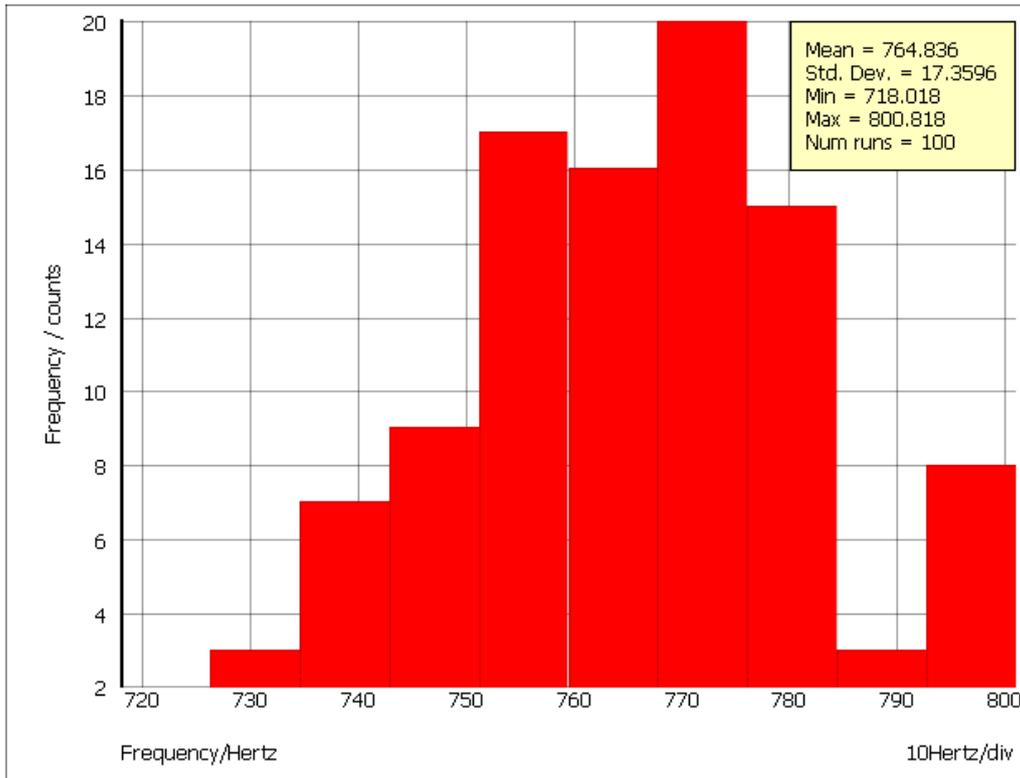
The example circuit has been set up to do 100 runs. This won't take long to run, typically less than one second. This is the procedure:

1. Run the simulation using **F9** or equivalent menu.
2. Select menu **Probe | Plot Histogram...**
3. Left click on the output of the filter. This is the junction of R1 and C2.
4. You should see R1\_P appear in the expression box. We must now modify this with a goal function that returns the centre frequency. The function CentreFreq will do this. This measures the centre frequency by calculating the half way point between the intersections at some specified value below the peak. Typically you would use 3dB.

Modify the value in the expression box so that it reads:

```
CentreFreq(R1_p, 3)
```

5. In the Monte Carlo group, set Number of histogram bins to 10.
6. At this stage you can optionally modify the graph setting to enter your own axis labels etc. Now close the box. This is what you should see:



Note that the mean and standard deviation are automatically calculated.

## Creating Histograms using Fixed Probes

Histograms may be created using the arbitrary fixed probe. See [“Arbitrary Fixed Probe/Multi-step mode” on page 321](#)

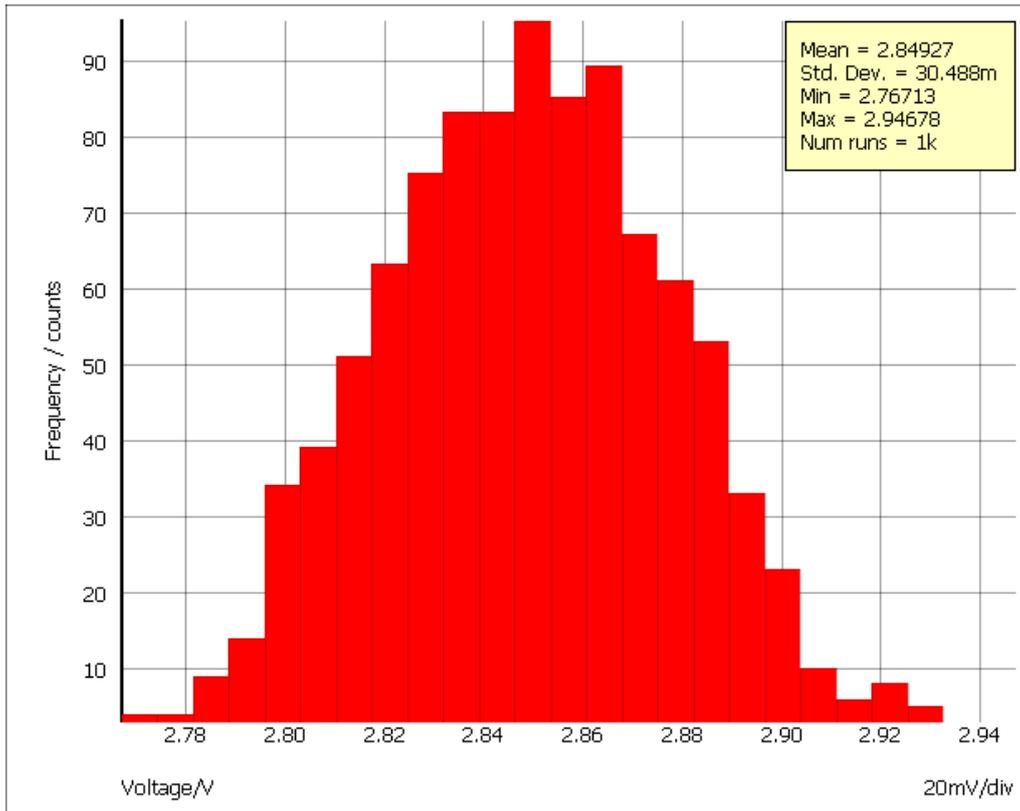
## Histograms for Single Step Monte Carlo Sweeps

An example of this type of run is shown in the section [“Monte Carlo” on page 233](#). These runs produce only a single curve with each point in the curve the result of the Monte Carlo analysis. With these runs you do not need to apply a goal function, just enter the name of the signal you wish to analyse. To illustrate this we will use the same example as shown in the section [“Monte Carlo” on page 233](#).

1. Open the example circuit Examples/Sweep/AC\_Param\_Monte
2. Run simulation.
3. Select menu **Probe | Plot Histogram...**
4. Left click on ‘+’ pin of the differential amplifier E1. You should see R4\_N appear in the box. Now enter a ‘-’ after this then click on the ‘-’ pin of the E1. This is what should be in the box:

R4\_N-R3\_N

5. In the Monte Carlo group, set the Number of histogram bins to 25.
6. Close box. You should see something like this:



This is a histogram showing the distribution of the gain of the amplifier at 100kHz.

### 13.17.4 Goal Functions

A range of functions are available to process curve data. Some of these are *primitive* and others use the user defined function mechanism. Primitive functions are compiled into the binary executable file while user defined functions are defined as scripts and are installed at functions at start up. User defined functions can be modified and you may also define your own. For more information refer to *Script Reference Manual/Applications/User Defined Script Based Functions*.

The functions described here aren't the only functions that may be used in the expression for performance analysis. They are simply the ones that can convert the array data that the simulator generates into a single value with some useful meaning. There are many other functions that process simulation vectors to produce another vector for example: log; sqrt; sin; cos and many more. These are defined in "[Script Reference Manual/Function Reference](#)" on page 399.

Of particular interest is the "[Truncate function](#)" on page 409. This selects data over a given X range so you can apply a goal function to work on only a specific part of the data.

### Primitive Functions

The following primitive functions may be used as goal functions. Not all actually return a single value. Some return an array and the result would need to be indexed. Maxima is an example.

Name	Description
“Maxima(real [, real, string])” on page 406	Returns array of all maximum turning points
“Maximum(real/complex [, real, real])” on page 406	Returns the largest value in a given range.
“Mean(real/complex)” on page 407	Returns the mean of all values. (You should not use this for transient analysis data as it fails to take account of the varying step size. Use Mean1 instead.)
“Mean1(real [, real, real])” on page 407	Finds the true mean accounting for the interval between data points
“Minima(real [, real, string])” on page 407	Returns array of all minimum turning points.
“Minimum(real/complex)” on page 407	Returns the largest value in a given range.
“RMS1(real [, real, real])” on page 408	Finds RMS value of data
“SumNoise(real [, real, real])” on page 409	Integrates noise data to find total noise in the specified range.
“XFromY(real, real [, real, real])” on page 410	Returns an array of X values at a given Y value.
“YFromX(real, real [, real])” on page 410	Returns an array of Y values at a given X value.

## User Defined Functions

The following functions are defined using the user defined functions mechanism. They are defined as scripts but behave like functions.

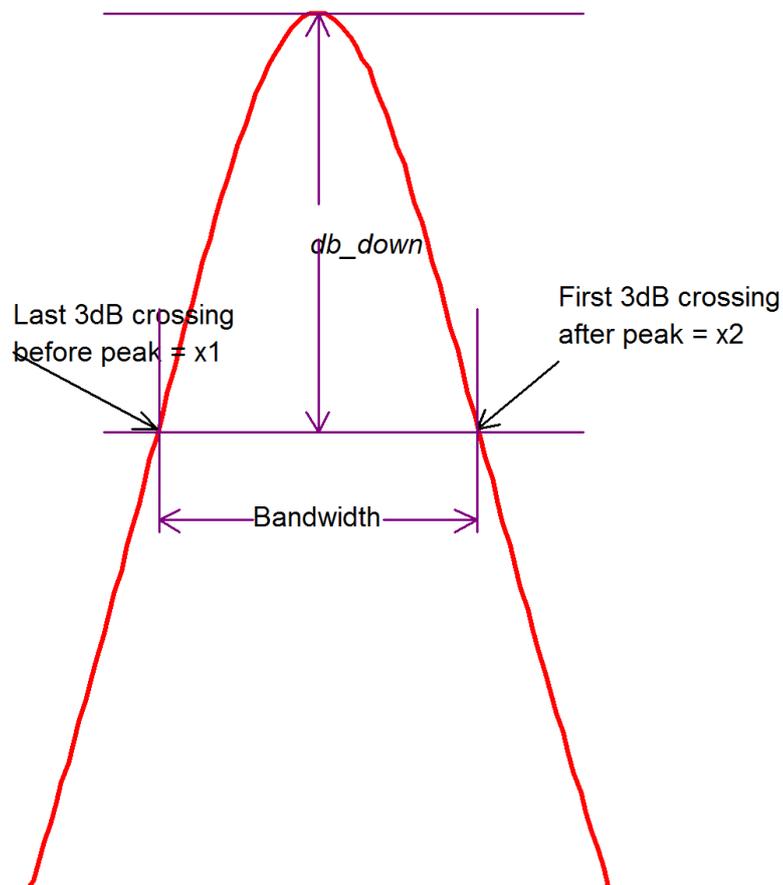
Name	Description
“BPBW(data, db_down)” on page 381	Band-pass bandwidth.
“Bandwidth(data, db_down)” on page 381	Same as BPBW
“CentreFreq(data, db_down)” on page 382	Centre frequency
“Duty(data, [threshold])” on page 382	Duty cycle of first pulse
“Fall(data, [start, end])” on page 382	Fall time
“Frequency(data, [threshold])” on page 383	Average frequency
“GainMargin(data, phaseInstabilityPoint)” on page 384	Gain Margin
“HPBW(data, db_down)” on page 384	High pass bandwidth
“LPBW(data, db_down)” on page 385	Low pass bandwidth
“Overshoot(data, [start, end])” on page 385	Overshoot
“PeakToPeak(data, [start, end])” on page 386	Peak to Peak
“Period(data, [threshold])” on page 386	Period of first cycle.
“PhaseMargin(data, phaseInstabilityPoint)” on page 386	Phase Margin
“PulseWidth(data, [threshold])” on page 386	Pulse width of first cycle
“Rise(data, [start, end])” on page 387	Rise time
“XatNthY(data, yValue, n)” on page 387	X value at the Nth Y crossing
“XatNthYn(data, yValue, n)” on page 387	X value at the Nth Y crossing with negative slope

Name	Description
<a href="#">“XatNthYp(data, yValue, n)” on page 388</a>	X value at the Nth Y crossing with positive slope
<a href="#">“XatNthYpct(data, yValue, n)” on page 388</a>	X value at the Nth Y crossing. y value specified as a percentage.
<a href="#">“YatX(data, xValue)” on page 388</a>	Y value at xValue
<a href="#">“YatXpct(data, xValue)” on page 388</a>	Y value at xValue specified as a percentage

## BPBW, Bandwidth

`BPBW(data, db_down)`

Finds the bandwidth of a band pass response. This is illustrated by the following graph



Function return  $x2-x1$  as shown in the above diagram.

Note that data is assumed to be raw simulation data and may be complex. It must not be in dBs.

Implemented by built-in script `uf_bandwidth`. Source may be obtained from the install CD (see ).

## CentreFreq, CenterFreq

```
CentreFreq(data, db_down)
```

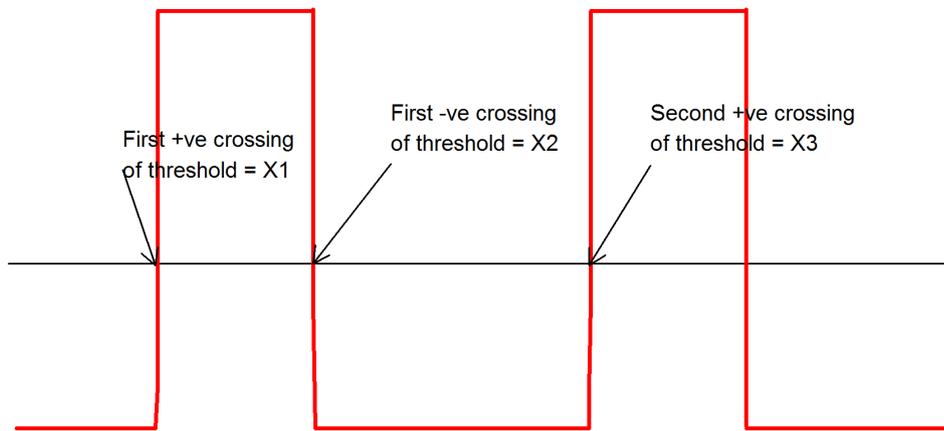
See diagram in “[BPBW, Bandwidth](#)” on page 381. Function returns  $(x1 + x2)/2$ .

Both British and North American spellings of centre (center) are accepted.

Implemented by built-in script uf\_centre\_freq. Source may be obtained from the install CD (see “[Install CD](#)” on page 474).

## Duty

```
Duty(data, [threshold])
```



Function returns  $(X2-X1)/(X3-X1)$ , where X1, X2 and X3 are defined in the above graph.

Default value for *threshold* is

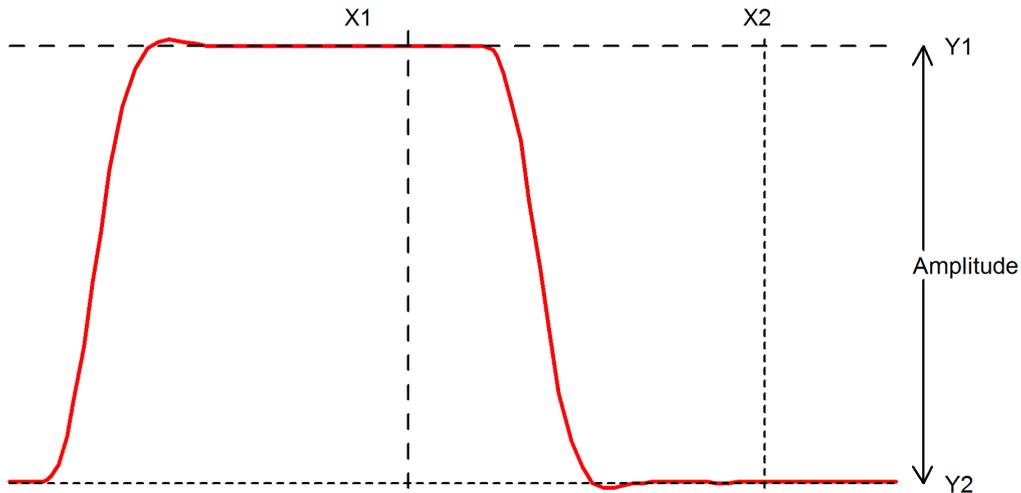
$(Y_{max}+Y_{min})/2$

Where  $Y_{max}$  = largest value in *data* and  $Y_{min}$  in smallest value in *data*.

Implemented by built-in script uf\_duty. Source may be obtained from the install CD (see “[Install CD](#)” on page 474).

## Fall

```
Fall(data, [xStart, xEnd])
```



Function returns the 10% to 90% fall time of the first falling edge that occurs between  $x_1$  and  $x_2$ . The 10% point is at  $y$  threshold  $Y_1 + (Y_2 - Y_1) * 0.1$  and the 90% point is at  $y$  threshold  $Y_1 + (Y_2 - Y_1) * 0.9$ .

If  $x_{start}$  is specified,  $X_1 = x_{start}$  otherwise  $X_1 = x$  value of first point in data.

If  $x_{end}$  is specified,  $X_2 = x_{end}$  otherwise  $X_2 = x$  value of last point in data.

If  $x_{start}$  is specified,  $Y_1 = y$  value at  $x_{start}$  otherwise  $Y_1 =$  maximum  $y$  value in data.

If  $x_{end}$  is specified,  $Y_2 = y$  value at  $x_{end}$  otherwise  $Y_2 =$  minimum  $y$  value in data.

Implemented by built-in script `uf_fall`. Source may be obtained from the install CD (see [“Install CD” on page 474](#)).

## Frequency

`Frequency(data, [threshold])`

Finds the average frequency of data.

Returns:

$$(n - 1) / (x_n - x_1)$$

Where:

$n$  = the number of positive crossings of *threshold*

$x_n$  = the  $x$  value of the  $n$ th positive crossing of *threshold*

$x_1$  = the  $x$  value of the first positive crossing of *threshold*

If *threshold* is not specified a default value of  $(y_{max} + y_{min}) / 2$  is used where  $y_{max}$  is the largest value in data and  $y_{min}$  is the smallest value.

Implemented by built-in script `uf_frequency`. Source may be obtained from the install CD (see [“Install CD” on page 474](#)).

## GainMargin

```
GainMargin(data, [phaseInstabilityPoint])
```

Finds the gain margin in dB of *data* where *data* is the complex open loop transfer function of a closed loop system. The gain margin is defined as the factor by which the open loop gain of a system must increase in order to become unstable. *phaseInstabilityPoint* is the phase at which the system becomes unstable. This is used to allow support for inverting and non-inverting systems. If *data* represents an inverting system, *phaseInstabilityPoint* should be zero. If *data* represents a non-inverting system, *phaseInstabilityPoint* should be -180.

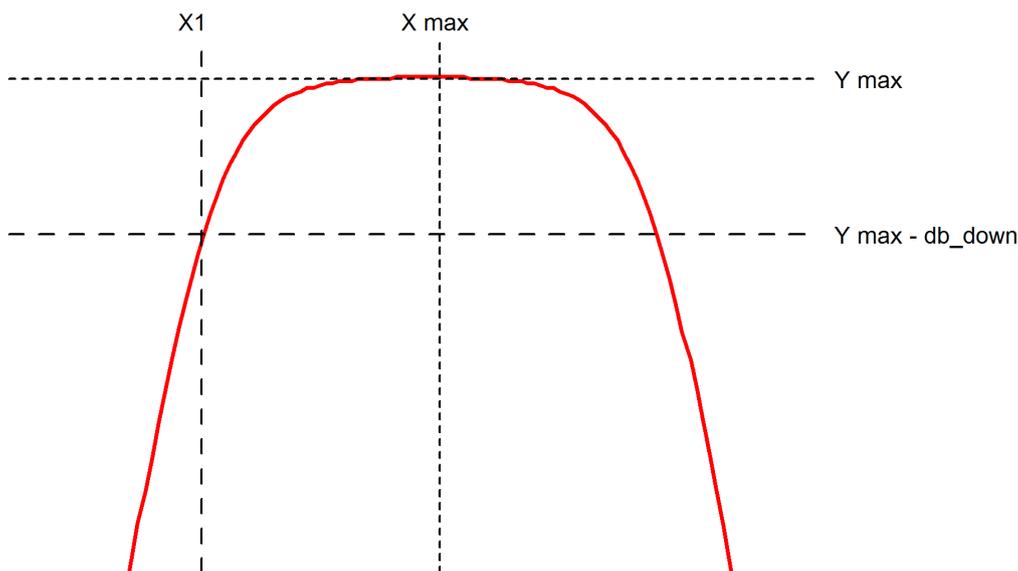
The function detects the frequencies at which the phase of the system is equal to *phaseInstabilityPoint*. It then calculates the gain at those frequencies and returns the value that is numerically the smallest. This might be negative indicating that the system is probably already unstable (but could be conditionally stable).

If the phase of the system does not cross the *phaseInstabilityPoint* then no gain margin can be evaluated and the function will return an empty vector.

## HPBW

```
HPBW(data, db_down)
```

Finds high pass bandwidth.



Returns the value of X1 as shown in the above diagram.

Y max is the y value at the maximum point.

X max is the x value at the maximum point.

X1 is the x value of the first point on the curve that crosses (Y max - db\_down) starting at X max and scanning *right to left*.

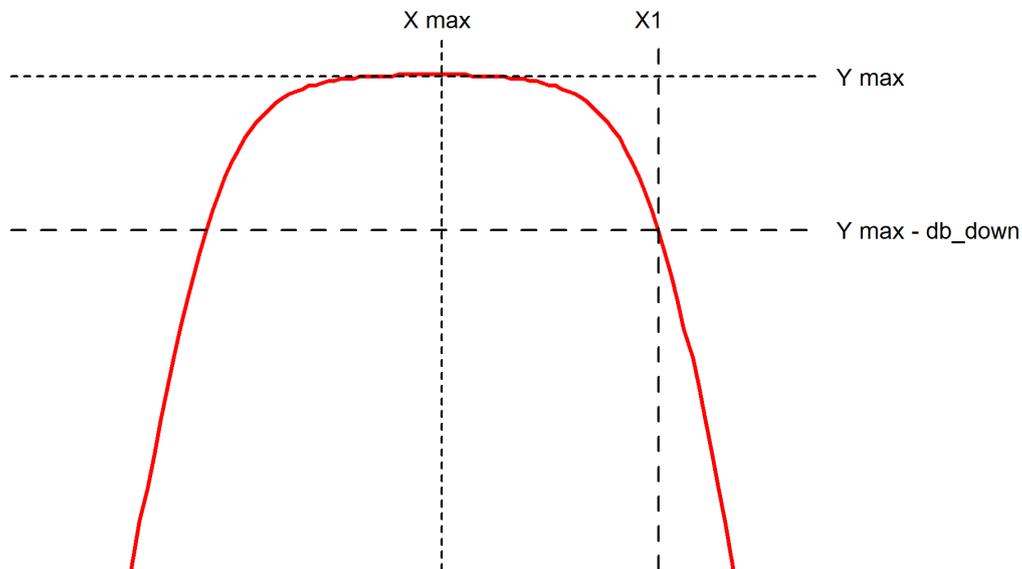
Note that *data* is assumed to be raw simulation data and may be complex. It must not be in dBs.

Implemented by built-in script uf\_hpbw. Source may be obtained from the install CD (see ).

## LPBW

```
LPBW (data, db_down)
```

Finds low pass bandwidth



Returns the value of X1 as shown in the above diagram.

Y max is the y value at the maximum point.

X max is the x value at the maximum point.

X1 is the x value of the first point on the curve that crosses (Y max - db\_down) starting at X max and scanning left to right.

Note that data is assumed to be raw simulation data and may be complex. It must not be in dBs.

Implemented by built-in script uf\_lpbw. Source may be obtained from the install CD (see [“Install CD” on page 474](#)).

## Overshoot

```
Overshoot (data, [xStart, xEnd])
```

Finds overshoot in percent.

Returns:

$$(y_{\text{Max}} - y_{\text{Start}}) / (y_{\text{Start}} - y_{\text{End}})$$

Where

yMax is the largest value found in the interval between xStart and xEnd.

yStart is the y value at xStart.

yEnd is the y value at xEnd.

If xStart is omitted it defaults to the x value of the first data point.

If `xEnd` is omitted it defaults to the `x` value of the last data point.

Implemented by built-in script `uf_overshoot`. Source may be obtained from the install CD (see “[Install CD](#)” on page 474).

## PeakToPeak

```
PeakToPeak(data, [xStart, xEnd])
```

Returns the difference between the maximum and minimum values found in the data within the interval `xStart` to `xEnd`.

If `xStart` is omitted it defaults to the `x` value of the first data point.

If `xEnd` is omitted it defaults to the `x` value of the last data point.

Implemented by built-in script `uf_peak_to_peak`. Source may be obtained from the install CD (see “[Install CD](#)” on page 474).

## Period

```
Period(data, [threshold])
```

Returns the period of the data.

Refer to diagram for the “[Duty](#)” on page 382 function. The Period function returns:

$X3 - X1$

Default value for *threshold* is

$(Y_{max} + Y_{min})/2$

Where  $Y_{max}$  = largest value in *data* and  $Y_{min}$  in smallest value in *data*.

Implemented by built-in script `uf_period`. Source may be obtained from the install CD (see “[Install CD](#)” on page 474).

## PhaseMargin

```
PhaseMargin(data, [phaseInstabilityPoint])
```

Finds the phase margin in dB of *data*, where *data* is the complex open loop transfer function of a closed loop system. The phase margin is defined as the angle by which the open loop phase shift of a system must increase in order to become unstable. *phaseInstabilityPoint* is the phase at which the system becomes unstable. This is used to allow support for inverting and non-inverting systems. If *data* represents an inverting system, *phaseInstabilityPoint* should be zero. If *data* represents a non-inverting system, *phaseInstabilityPoint* should be -180.

The function detects the frequencies at which the magnitude of the gain is unity. It then calculates the phase shift at those frequencies and returns the value that is numerically the smallest. This might be negative indicating that the system is probably already unstable (but could be conditionally stable).

If the gain of the system does not cross unity then no phase margin can be evaluated and the function will return an empty vector.

## PulseWidth

```
PulseWidth(data, [threshold])
```

Returns the pulse width of the first pulse in the data.

Refer to diagram for the “Duty” on page 382 function. The PulseWidth function returns:

$X2 - X1$

Default value for *threshold* is

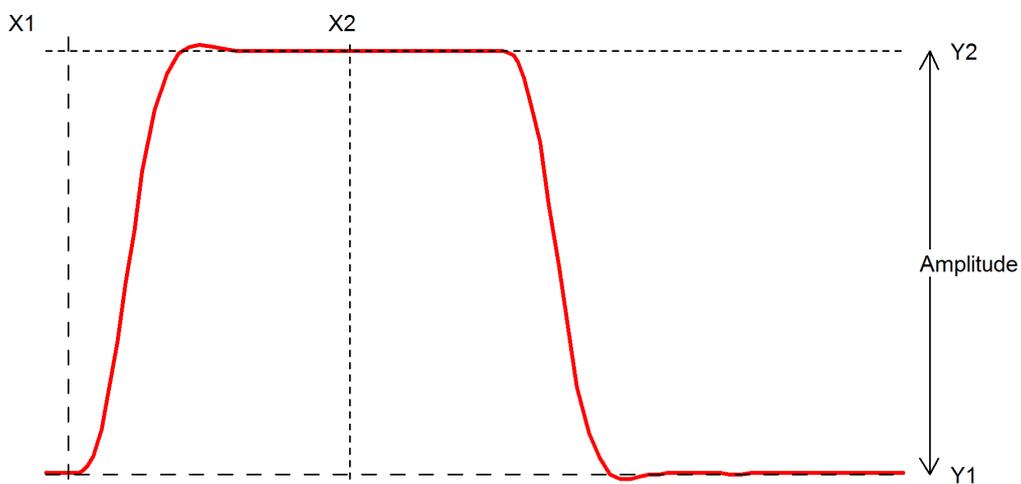
$(Y_{max} + Y_{min})/2$

Where  $Y_{max}$  = largest value in *data* and  $Y_{min}$  in smallest value in *data*.

Implemented by built-in script `uf_pulse_width`. Source may be obtained from the install CD (see “Install CD” on page 474).

## Rise

`Rise(data, [xStart, xEnd])`



Function returns the 10% to 90% rise time of the first rising edge that occurs between  $x1$  and  $x2$ . The 10% point is at  $y$  threshold  $Y1 + (Y2 - Y1) * 0.1$  and the 90% point is at  $y$  threshold  $Y1 + (Y2 - Y1) * 0.9$ .

If  $xStart$  is specified,  $X1 = xStart$  otherwise  $X1 = x$  value of first point in data.

If  $xEnd$  is specified,  $X2 = xEnd$  otherwise  $X2 = x$  value of last point in data.

If  $xStart$  is specified,  $Y1 = y$  value at  $xStart$  otherwise  $Y1 =$  maximum  $y$  value in data.

If  $xEnd$  is specified,  $Y2 = y$  value at  $xEnd$  otherwise  $Y2 =$  minimum  $y$  value in data.

## XatNthY

`XatNthY(data, yValue, n)`

Returns the  $x$  value of the data where it crosses  $yValue$  for the  $n$ th time.

## XatNthYn

`XatNthYn(data, yValue, n)`

Returns the x value of the data where it crosses *yValue* for the *nth* time with a negative slope.

### **XatNthYp**

```
XatNthYp(data, yValue, n)
```

Returns the x value of the data where it crosses *yValue* for the *nth* time with a positive slope.

### **XatNthYpct**

```
XatNthYpct(data, yValue, n)
```

As **XatNthY** but with *yValue* specified as a percentage of the maximum and minimum values found in the data.

### **YatX**

```
YatX(data, xValue)
```

Returns the y value of the data at x value = *xValue*.

### **YatXpct**

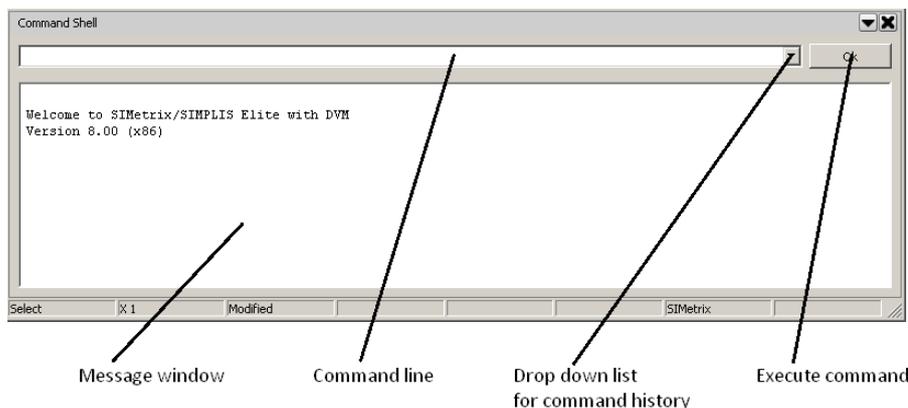
```
YatXpct(data, xValue)
```

As **YatX** but with *xValue* specified as a percentage of the total x interval of the data.

# Chapter 14

## The Command Shell

### 14.1 Command Line



The command line is at the top of the command shell. See diagram above.

The vast majority of operations with SIMetrix can be executed from menus or pre-defined keys and do not require the use of the command line. However, a few more advanced operations do require the use of the command line. From the command line you can run a script or an internal command. You can also define a new menu to call a script, command or series of commands. In fact all the built in menu and keys are in fact themselves defined as commands or scripts. These definitions can be changed as well as new ones defined. See [“Editing the Menu System”](#) on page 390.

Details of some of the available commands are given in [“Command and Function Reference”](#) on page 394. The remainder are documented in the *Script Reference Manual/Command Reference*.

#### 14.1.1 Command History

A history of manually entered commands is available from the drop down list (select arrow to the right of the command line). Some other commands entered via menus or from a script may also be placed in the command history.

### 14.1.2 Message Window

Various messages may be displayed in the message window below the command line. These include command progress, errors, warnings and listing outputs. The text in the window may be copied to the clipboard using a context sensitive menu activated by the right mouse button.

### 14.1.3 Multiple commands on one line

You can place multiple commands on the same line separated by a semi-colon - ';' . This is the only way a menu or key can be defined to execute more than one command.

### 14.1.4 Scripts

SIMetrix features a comprehensive scripting language. Full details of this can be found in the *Script Reference Manual/Script Language*.

### 14.1.5 Command Line Editing

The command line itself is a windows edit control. The cursor keys, home and end all work in the usual way. You can also copy (control-C), cut (control-X) and paste (control-V) text. There is also a right click popup menu with the usual edit commands.

### 14.1.6 Command Line Switches

Many commands have switches. These are always preceded by a '/' and their meaning is specific to the command. There are however four global switches which can be applied to any command. *These must always be placed immediately after the command and before any command specific switches*. Global switches are as follows:

- /e Forces command text to copied to command history
- /ne Inhibits command text copying to command history
- /quiet Inhibits error messages for that command. This only stops error message being displayed. A script will still be aborted if an error occurs but no message will be output
- /noerr Stops scripts being aborted if there is an error. The error message will still be displayed.

## 14.2 Editing the Menu System

### 14.2.1 Overview

The menu system in SIMetrix may be edited by one of two methods:

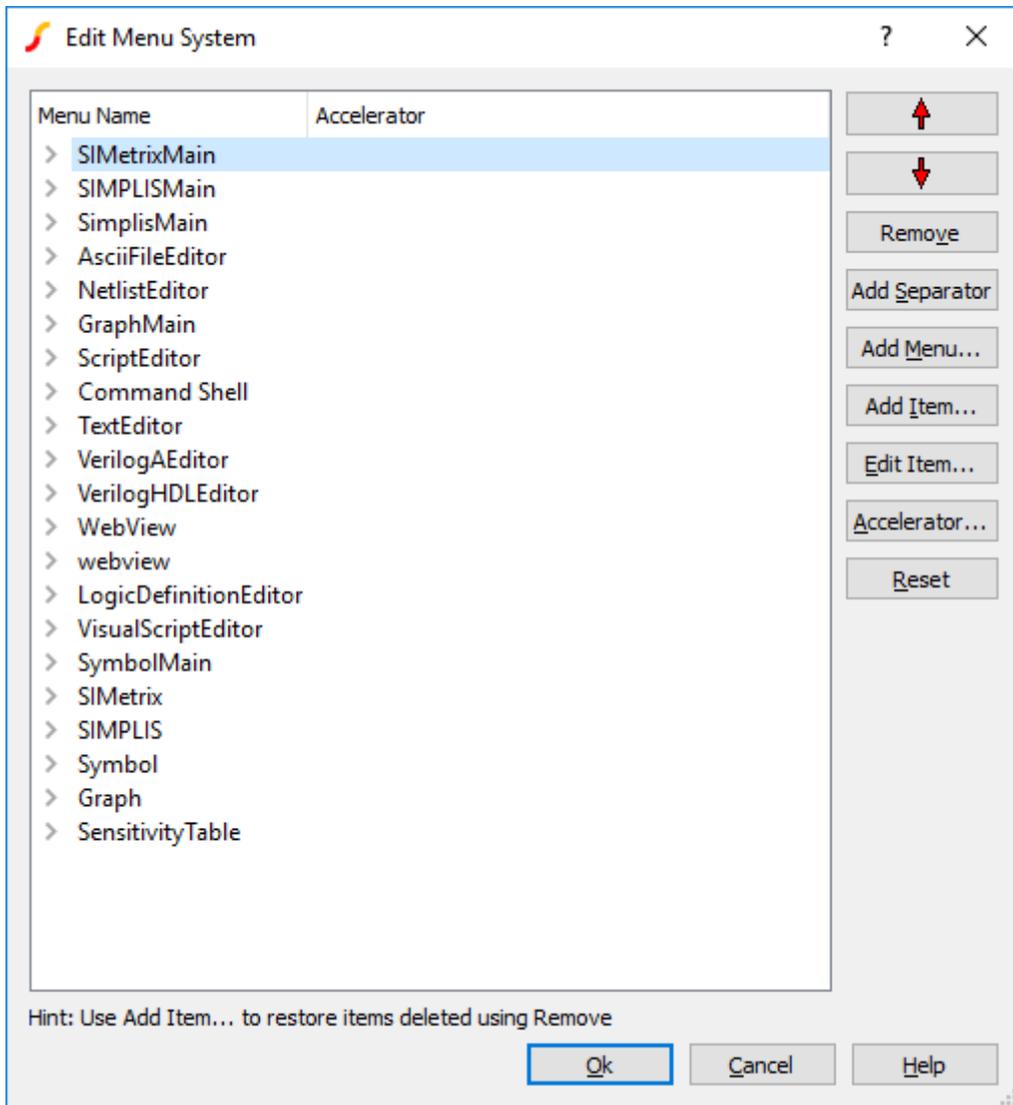
1. Using the script language. You can either edit the menu.sxscr script that builds the menu system on program start, or you can make additions by adding commands to the start up script.
2. Using the GUI based menu editor.

The first method offers the ultimate in flexibility but has a steep learning curve. The second method is quite simple to use and is appropriate for simple changes to the menu system such as deleting unused menus or changing hot-key definitions.

The following describes the second method. For details about the first method, please refer to *Script Reference Manual/Applications/Schematic Symbol Script Definition*.

### 14.2.2 Procedure

Select menu **File | Options | Edit Menu....** This will open the following dialog box



The above view shows the layout for SIMetrix/SIMPLIS products. For SIMetrix only products the top level menu choices will be slightly different.

The left hand pane shows the current menu system in a tree structured list. The buttons on the right allow you to move, delete or add new menu items and sub menus.

#### To Delete a Menu Item or Entire Sub-menu

Select the item you wish to delete on the left hand side then press **Remove**. To subsequently restore a deleted item, use **Add Item...** (see below).

## To Move a Menu Item or Sub-menu

Select the item then use the up and down arrows key to move as required. If you wish to move a menu item to a different sub-menu, you should remove it (as described above) then add it again using **Add Item...** (see below).

## To Add a New Menu Item

You can add a previously removed menu item to a new location or you can create a completely new menu item.

In both cases, select the location within a sub-menu and press **Add Item....** To add a previously removed menu select **Add Existing** and choose an item in the left hand pane. This pane will be empty if you have not previously removed any menu items. (This could be in a previous session as deleted items are always remembered).

To add a new item press **Add New** then enter the required values. For **Name** enter the name of the menu as you would like to see it in the menu itself. Use the ‘&’ character to denote underlined letters used to denote **alt**-key short-cuts. E.g. **&My Menu** will be displayed **My Menu** and will be activated with **alt-M**. For **Command String** you must enter a valid SIMetrix command. Typically this would call a user defined script, but you may also enter primitive commands or a list of primitive commands separated by semi-colons. See the *Script Reference Manual/Command Reference* for full details.

To add a separator to the sub-menu, press **Add Separator**.

## To Create a New Sub-menu

Select the location for your new menu then press **Add Menu....** Enter the menu name as required. The new sub-menu will have one single empty item which you can select to add new items to the sub-menu. The empty item will not appear in SIMetrix menus.

## To Edit or Add an Accelerator Key

Select the menu item to which you wish to assign the accelerator key. Press **Accelerator....** You will now be asked to press a single key or key combination with **shift**, **control** or **alt**. The key you press will be assigned to the menu.

Press **Remove** to delete the accelerator key assignment.

## To Reset to ‘Factory’ Settings

Press **Reset** to reset the menu system back to ‘factory settings’. Usually this means that the menu system will revert to the structure defined when SIMetrix was first installed. However, if you defined any menus in the startup script using the **DefMenu** command, these menus will be faithfully restored as well.

## ‘Reopen’ Menu

The **Reopen** menu in the File menu is dynamically updated to include schematic files recently opened or saved. The items in this menu are not listed in the Menu Editor and cannot be edited. The **Reopen** sub-menu itself should not be deleted, or moved to a new sub-menu nor should it be renamed, but it may be repositioned in the File menu as desired.

## 14.3 User Defined Toolbars and Buttons

All toolbars and buttons are user definable and it is also possible to create new toolbars and buttons. Full details are provided in the Applications chapter of the *Script Reference Manual/Applications/Creating and Modifying Toolbars*.

## 14.4 Message Window

The *message window* is the window in the command shell below the command line. The majority of messages, including errors and warnings, are displayed here. The window can be scrolled vertically with the scroll bar.

You can copy a line of text from the *message window* to the command line by placing the cursor on the line and either double clicking the left mouse key or pressing the Insert key.

# Chapter 15

## Command and Function Reference

### 15.1 Introduction

There are about 600 functions and 300 commands available but only a few are covered in this chapter. Details of all available functions and commands can be found in *Script Reference Manual/Function Reference* and *Script Reference Manual/Command Reference*.

#### 15.1.1 Notation

##### Symbols Used

Square brackets: [ ]	These signify a command line parameter or switch which is optional.
Pipe symbol:	This signifies either/or.
Ellipsis: ...	This signifies 1 or more optional multiple entries.

##### Fonts

Anything that would be typed in is displayed in a `fixed width` font.

Command line parameters are in *italics*.

##### Case

Although upper and lower cases are used for the command names, they are NOT in fact case sensitive.

##### Examples

```
OpenGroup [ /text ] [ filename ]
```

Both `/text` (a switch) and *filename* (a parameter) are optional in the above example.

So the following are all legitimate:

```

OpenGroup
OpenGroup /text
OpenGroup run23.sxdat
OpenGroup /text output.txt

DelCrv curve_number...

```

One or more *curve\_number* parameters may be given.

So the following are all legitimate:

```

DelCrv 1 2 3
DelCrv 1

```

## 15.2 Command Summary

Only a 4 of the approximately 300 available commands are detailed in this chapter and a list is given in the table below. Documentation for the remainder is provided in the *Script Reference Manual/Command Reference*.

Command name	Description
OpenGroup	Create new group (of simulation data) from data file
Set	Set option
Show	Displays the value of an expression. Can be used to export data to a file in ASCII form
Unset	Delete option

## 15.3 Reference

### 15.3.1 OpenGroup

```
OpenGroup [/text] [/overwrite] filename
```

Reads in a data file. There are more options available in addition to the above. Please refer to the *Script Reference Manual/Command Reference/OpenGroup* for further information.

<i>/text</i>	If specified, data file is assumed to be in text format. Otherwise the file is input as a SIMetrix binary data file as saved by the SaveGroup command. See <a href="#">“Data Import and Export” on page 369</a> for details of text format.
<i>/overwrite</i>	Forces existing group of the same name to be overwritten. If not specified, the group being read in will be renamed if a group of the same name already exists.
<i>filename</i>	Name of file to be input.

OpenGroup creates a new Group. If */text* is not specified then the name of the group will be that with which it was stored provided the name does not conflict with an existing group. If there is a conflict the name will be modified to be unique. If */text* is specified then the group will be named:

```
textn
```

where 'n' is chosen to make the name unique.

### 15.3.2 Set

```
Set [/temp] [option_spec [option_spec...]]
```

Defines an option.

<code>/temp</code>	If specified, the option setting will be temporary and will be restored to its original value when control returns to the command line. (i.e when all scripts have completed).
<code>option_spec</code>	Can be one of two forms: Form1: <i>option_name</i> Form2: <i>option_name = option_value</i>  <i>option_name</i> can be any of the names listed in <a href="#">“Options” on page 444</a> . For options of type Boolean, use form1. For others, use form 2.

### See Also

[“Options” on page 444](#)

[“Unset” on page 397](#)

### 15.3.3 Show

```
Show [/file filename] [/append filename] [/noindex] [/noHeader] [/plain]
[/force] [/clipboard] [/names names] [/width width] expression
[expression ...]
```

Displays the value of an expression. This command can be used to export data from the simulator in ASCII form. See [“Data Import and Export” on page 369](#) for more details.

<code>/file filename</code>	If specified, outputs result to filename. The values are output in a format compatible with OpenGroup /text. (See <a href="#">“OpenGroup” on page 395</a> ).
<code>/append filename</code>	As /file except that file is appended if it already exists.
<code>/noindex</code>	If the vector has no reference, the index value for each element is output if this switch is <i>not</i> specified.
<code>/noHeader</code>	If specified, the header providing vector names etc. will be inhibited.
<code>/plain</code>	If specified, no index (as /noindex), and no header (as /noHeader) will be output. In addition, string values will be output without enclosed single quotation marks.
<code>/force</code>	File specified by /file will be unconditionally overwritten if it exists.
<code>/clipboard</code>	Outputs data to system clipboard.
<code>/names names</code>	Semi-colon delimited string providing names to be used as headings for tabulated data. If not specified, the vector names are used instead.

<code>/width width</code>	Page width in columns for tabulated data. If not specified no limit will be set.
<code>/lock</code>	If specified with <code>/file</code> , a lock file will be created while the write operation is being performed. The file will have the extension <code>.lck</code> . This can be used to synchronise data transfers with other applications. The file will be locked for write operations.
<code>expression</code>	Expression to be displayed. If expression is an array, all values will be displayed.

## Notes

To enter multiple expressions, separate each with a comma.

The display of arrays with a very large number of elements (>500) can take a long time. For large arrays it is recommended that the `/file` switch is used to output the results to a file. The file can then be examined with a text editor or spreadsheet program.

The results will be tabulated if all vectors are compatible that is have the same x-values. If the any vectors listed are not compatible, each vector's data will be listed separately.

The precision of numeric values can be controlled using the "Precision" option setting. Use the command `Set precision = value`. This sets the precision in terms of the column width.

### 15.3.4 Unset

```
UnSet [/temp] option_name
```

Deletes specified option. See ["Options" on page 444](#) for a full explanation.

<code>/temp</code>	If specified, the option setting will be deleted temporarily and will be restored to its original value when control returns to the command line. (i.e when all scripts have completed).
<code>option_name</code>	Name of option. This would usually be defined with a value using <a href="#">"Set" on page 396</a> . See <a href="#">"Options" on page 444</a> for a complete list.

Note that most Option values are *internal*. This means that they always have a value. If such an option is Unset, it will be restored to its default value and not deleted. See ["Options" on page 444](#) for more details.

If Unset is called for an option that has not been Set and which is not internal and error will be displayed.

## 15.4 Function Summary

The following table lists a small selection of the functions available with SIMetrix. Full documentation for these is provided in the *Script Reference Manual/Function Reference*.

Function name	Description
<a href="#">"abs(real/complex)" on page 399</a>	Absolute value
<a href="#">"arg(real/complex)" on page 399</a>	phase (result wraps at 180/-180 degrees)
<a href="#">"arg_rad(real/complex)" on page 399</a>	phase (radians). Result wraps at pi/-pi radians

Function name	Description
“atan(real/complex)” on page 400	Arc tangent
“cos(real/complex)” on page 400	Cosine
“db(real/complex)” on page 400	$dB(x) = 20 * \log_{10}(\text{mag}(x))$
“diff(real)” on page 400	Return derivative of argument
“exp(real/complex)” on page 400	Exponential
“fft(real [, string])” on page 400	Fast Fourier Transform
“FIR(real, real [, real])” on page 400	Finite Impulse Response digital filter
“Floor(real)” on page 401	Returns argument truncated to next lowest integer
“GroupDelay(real/complex)” on page 401	Returns group delay of argument
“Histogram(real, real)” on page 402	Returns histogram of argument
“Iff(real, any, any)” on page 403	Returns a specified value depending on the outcome of a test
“IIR(real, real [, real])” on page 403	Infinite Impulse Response digital filter
“im, imag(real/complex)” on page 404	Return imaginary part of argument
“integ(real)” on page 404	Returns integral of argument
“Interp(real, real [, real, real])” on page 404	Interpolates argument to specified number of evenly spaced points
“IsComplex(any)” on page 404	Returns TRUE if argument is complex
“length(any)” on page 405	Returns number of elements in vector.
“ln(real/complex)” on page 405	Natural logarithm
“log, log10(real/complex)” on page 405	Base 10 logarithm
“mag, magnitude(real/complex)” on page 406	Magnitude (same as abs())
“maxidx(real/complex)” on page 406	Returns index of vector where largest value is held
“Maxima(real [, real, string])” on page 406	Returns locations of maxima of specified vector
“mean(real/complex)” on page 407	Returns statistical mean of all values in vector
“Mean1(real [, real, real])” on page 407	Returns mean of data in given range
“minidx(real/complex)” on page 407	Returns index of vector where smallest value is held
“Minima(real [, real, string])” on page 407	Returns locations of minima of specified vector
“norm(real/complex)” on page 407	Returns argument scaled so that its largest value is unity.
“ph, phase(real/complex)” on page 408	Returns phase of argument
“phase_rad(real/complex)” on page 408	As ph() but result always in radians
“Range(real/complex [, real, real])” on page 408	Returns range of vector
“re, real(real/complex)” on page 408	Return real part of argument
“Ref(real/complex)” on page 408	Returns reference of argument

Function name	Description
<a href="#">“Rms(real)” on page 408</a>	Returns accumulative RMS value of argument
<a href="#">“RMS1(real [, real, real])” on page 408</a>	Returns RMS of argument over specified range
<a href="#">“rnd(real)” on page 408</a>	Returns random number
<a href="#">“RootSumOfSquares(real [, real, real])” on page 409</a>	Returns root sum of squares of argument over specified range
<a href="#">“sign(real)” on page 409</a>	Return sign of argument
<a href="#">“sin(real/complex)” on page 409</a>	Sine
<a href="#">“sqrt(real/complex)” on page 409</a>	Square root
<a href="#">“tan(real/complex)” on page 409</a>	Tangent
<a href="#">“Truncate(real [, real, real])” on page 409</a>	Returns vector that is a sub range of supplied vector
<a href="#">“unitvec(real)” on page 410</a>	Returns vector of specified length whose elements are all 1
<a href="#">“vector(real)” on page 410</a>	Returns vector of specified length with each element equal to its index
<a href="#">“XFromY(real, real [, real])” on page 410</a>	Returns array of values specifying horizontal locations where specified vector crosses given y value
<a href="#">“YFromX(real, real [, real])” on page 410</a>	Returns array of values specifying the vertical value of the specified vector at the given x value.

## 15.5 Function Reference

Only a few of the approx. 200 functions are documented here. For the rest, please refer to the *Script Reference Manual/Function Reference*. The ones detailed here are the functions that accept and return numeric values and that could conceivably be used for graph plots.

### 15.5.1 abs(real/complex)

Returns absolute value or magnitude of argument. This function is identical to the mag() function.

### 15.5.2 arg(real/complex)

Same as [phase\(\)](#) except the result wraps at 180/-180 degrees.

### 15.5.3 arg\_rad(real/complex)

Same as [phase\\_rad\(\)](#) except the result wraps at  $\pi / -\pi$  radians.

### 15.5.4 atan(real/complex)

Returns the arc tangent of its argument. If degrees option is set return value is in degrees otherwise radians.

### 15.5.5 cos(real/complex)

Return cosine of argument in radians. Use cos\_deg if the argument is in degrees.

### 15.5.6 db(real/complex)

Returns  $20 \cdot \log_{10}(\text{mag}(\text{argument}))$

### 15.5.7 diff(real)

Returns the derivative of the argument with respect to its reference (x-values). If the argument has no reference the function returns the derivative with respect to the argument's index - in effect a vector containing the difference between successive values in the argument.

### 15.5.8 exp(real/complex)

Returns e raised to the power of argument. If the argument is greater than 709.016, an overflow error occurs.

### 15.5.9 fft(real [, string])

Arg1 Vector to be processed

Arg2 String specifying window type. Possible values are: Hanning (default) and None

Performs a Fast Fourier Transform on supplied vector. The number of points used is the next binary power higher than the length of argument 1. Excess points are zero-filled. Window used may be *Hanning* (default) or *None*.

Users should note that using this function applied to raw transient analysis data will not produce meaningful results as the values are unevenly spaced. If you apply this function to simulation data, you must either specify that the simulator outputs at fixed intervals (select the *Output at interval* option in the **Choose Analysis** dialog box) or you must interpolate the results using the [Interp\(\)](#) function. (The FFT plotting menu items run a script which interpolate the data if it detects that the results are unevenly spaced. Use of these menus does not require special consideration by the user.)

Further information on FFTs can be found in [“Fourier Analysis” on page 325](#).

### 15.5.10 FIR(real, real [, real])

Arg1 Vector to be filtered

Arg2 Filter coefficients

Arg3 Initial conditions. Default - all zero

Performs Finite Impulse Response digital filtering on supplied vector. This function performs the operation:

$$y_n = x_n \cdot c_0 + x_{n-1} \cdot c_1 + x_{n-2} \cdot c_2 \dots$$

Where:

x is the input vector (argument 1)

c is the coefficient vector (argument 2)

y is the result (returned value)

The third argument provide the history of  $x$  i.e.  $x_{-1}, x_{-2}$  etc. as required.

The operation of this function (and also the [IIR\(\)](#) function) is simple but its application can be the subject of several volumes! Below is the simple case of a four sample rolling average. In principle an almost unlimited range of FIR filtering operations may be performed using this function. Any text on Digital Signal Processing will provide further details.

Users should note that using this function applied to raw transient analysis data will not produce meaningful results as the values are unevenly spaced. If you apply this function to simulation data, you must either specify that the simulator outputs at fixed intervals (select the *Output at interval* option in the **Choose Analysis** dialog box) or you must interpolate the results using the [Interp\(\)](#) function.

## Example

Suppose a vector VOUT exist in the current group (simulation results). The following will plot VOUT with a 4 sample rolling average applied

```
Plot FIR(vout, [0.25, 0.25, 0.25, 0.25])
```

Alternatively, the following does the same

```
Plot FIR(vout, 0.25*unitvec(4))
```

### 15.5.11 Floor(real)

Returns the argument truncated to the next lowest integer.

## Examples

```
Floor(3.45) = 3
Floor(7.89) = 7
Floor(-3.45) = -4
```

### 15.5.12 GroupDelay(real/complex)

Returns the group delay of the argument. Group delay is defined as:

$$\frac{d(\text{phase}(y))}{dx} \cdot \frac{1}{2\pi}$$

where y is the supplied vector and x is its reference. The GroupDelay() function expects the result of AC analysis where y is a voltage or current and its reference is frequency.

This function will yield an error if its argument is complex and has no reference.

### 15.5.13 Histogram(real, real)

Arg1                      Vector  
Arg2 Number of bins

Creates a histogram of argument 1 with the number of bins specified by argument 2. The bins are divided evenly between the maximum and minimum values in the argument.

Histograms are useful for finding information about waveforms that are difficult to determine by other means. They are particularly useful for finding flat areas such as the flat tops of pulses as these appear as well defined peaks. The Histogram() function is used in the rise and fall time scripts for this purpose.

Users should note that using this function applied to raw transient analysis data will produce misleading results as the simulation values are unevenly spaced. If you apply this function to simulation data, you must either specify that the simulator outputs at fixed intervals (select the *Output at interval* option in the **Choose Analysis** dialog box) or you must interpolate the results using the [Interp\(\)](#) function.

### 15.5.14 Iff(real, any, any)

Arg1    Test  
Arg2    Result if test TRUE  
Arg3    Result if test FALSE

Arg2 and Arg3 must both be the same type

If the first argument evaluates to TRUE (i.e. non-zero) the function will return the value of argument 2. Otherwise it will return the value of argument 3. Note that the type of arguments 2 and 3 must both be the same. No implicit type conversion will be performed on these arguments.

### 15.5.15 IIR(real, real [, real])

Arg1    Vector to be filtered  
Arg2    Coefficients  
Arg3    Initial conditions - default zero

Performs Infinite Impulse Response digital filtering on supplied vector. This function performs the operation:

$$y_n = x_n \cdot c_0 + y_{n-1} \cdot c_1 + y_{n-2} \cdot c_2 \dots$$

Where:

- $x$  is the input vector (argument 1)
- $c$  is the coefficient vector (argument 2)
- $y$  is the result (returned value)

The third argument provide the history of  $y$  i.e.  $y_{-1}$ ,  $y_{-2}$  etc. as required.

The operation of this function (and also the [FIR\(\)](#) function) is simple but its application can be the subject of several volumes! In principle an almost unlimited range of IIR filtering operations may be performed using this function. Any text on Digital Signal Processing will provide further details.

Users should note that using this function applied to raw transient analysis data will not produce meaningful results as the values are unevenly spaced. If you apply this function to simulation data, you must either specify that the simulator outputs at fixed intervals (select the *Output at interval* option in the

**Choose Analysis** dialog box) or you must interpolate the results using the **Interp()** function. The following graph shows the result of applying a simple first order IIR filter to a step:

The coefficients used give a time constant of 10 \* the sample interval. In the above the sample interval was 1µSec so giving a 10µSec time constant. As can be seen a first order IIR filter has exactly the same response as an single pole RC network. A general first order function is:

$$y_n = x_n \cdot c_0 + y_{n-1} \cdot c_1$$

where

$$c_0 = 1 - \exp(-T/\tau)$$

$$\text{and } c_1 = \exp(-T/\tau)$$

and  $\tau$  = time constant

and  $T$  = sample interval

The above example is simple but it is possible to construct much more complex filters using this function. While it is also possible to place analog representations on the circuit being simulated, use of the IIR function permits viewing of filtered waveforms after a simulation run has completed. This is especially useful if the run took a long time to complete.

### 15.5.16 im(real/complex), imag(real/complex)

Returns imaginary part of argument.

### 15.5.17 integ(real)

Integrates the argument with respect to its reference (x-values).

The function uses simple trapezoidal integration.

An error will occur if the argument supplied has no reference.

### 15.5.18 Interp(real, real [, real, real])

- Arg1 Vector to be interpolated
- Arg2 Number of points in result
- Arg3 Interpolation order
- Arg4 (Boolean) include last point

Returns a vector with length specified by argument 2 obtained by interpolating the vector supplied as argument 1 at evenly spaced intervals. The optional third argument specifies the interpolation order. This can be any integer 1 or greater but in practice there are seldom reasons to use values greater than 4.

The Interp() function overcomes some of the problems caused by the fact that raw transient analysis results are unevenly spaced. It is used by the FFT plotting scripts to provide evenly spaced sample points for the **FFT()** function. The Interp() function also makes it possible to perform operations on two vectors that originated from different transient runs and therefore will have different sample points.

### 15.5.19 IsComplex(any)

Returns 1 (=TRUE) if the supplied argument is complex and 0 (=FALSE) if the argument is any other type

### 15.5.20 length(any)

Returns the number of elements in the argument. The result will be 1 for a scalar and 0 for an empty value.

The length() function is the only function which will not return an error if supplied with an *empty* value. Empty variables are returned by some functions when they cannot produce a return value. All other functions and operators will yield an error if presented with an empty value and abort any script that called it.

### 15.5.21 ln(real/complex)

Returns the natural logarithm of the argument.

#### Using ln() with Negative or Complex Values

If the argument is real and 0 or negative an error will result. If the argument is complex it will return a complex result even if the imaginary part is 0 and the real part negative.

E.g.

$$\ln(-1)$$

will produce an error. But:

$$\ln((-1, 0))$$

will give the answer  $(0, 3.1415926535897931) = j\pi$ .

An error will always occur if both real and imaginary parts are zero.

#### Using ln() with AC Analysis Data

See notes for the [log10\(\)](#) function.

### 15.5.22 log10(real/complex), log(real/complex)

Returns log to base 10 of argument. In general, we recommend using log10 rather than log. Software products of all types vary in their interpretation of log(). Some treat it as log to the base 10 and others treat it as log to the base  $e$ . By using log10() there will never be any doubt.

#### Using log10() with Negative or Complex Values

See notes for the [ln\(\)](#) function

#### Using log10() with AC Analysis Data

The data output by the simulator when running an AC or TF analysis is complex. As described in [“Using ln\(\) with Negative or Complex Values” on page 405](#), all SIMetrix logarithm functions correctly handle complex arguments and return a complex value. This means that the following expression to calculate dB will not produce the expected result:

$$20 * \log(\text{data})$$

where *data* is a value produced by an AC analysis simulation. What you should do is:

$$20 * \log(\text{mag}(\text{data}))$$

The `mag()` function will convert the complex data to real values which is actually what is intended. Better still use:

$$\text{db}(\text{data})$$

This is equivalent to  $20 * \log(\text{mag}(\text{data}))$ .

Note that the graph system will always plot the magnitude of complex data. But, any expression presented for plotting will be evaluated as complex and only the final result is converted to real. So  $20 * \log(\text{data})$  will be plotted as  $\text{mag}(20 * \log(\text{data}))$ . This is not the same as  $20 * \log(\text{mag}(\text{data}))$  when *data* is complex.

### 15.5.23 `mag(real/complex)` , `magnitude(real/complex)`

Returns the magnitude of the argument. This function is identical to the `abs()` function.

### 15.5.24 `maxidx(real/complex)`

Returns index of the array element in argument 1 with the largest magnitude.

### 15.5.25 `Maxima(real [, real, string])`

- Arg1 Vector
- Arg2 Minimum value. Default  $-\infty$
- Arg3 Options array. Possible values are:
  - `xSort` Sort output in order of x values
  - `noInterp` Don't interpolate.

Returns an array of values holding every maximum point in the supplied vector whose value is above argument 2. The value returned - if *noInterp* is not specified - is obtained by fitting a parabola to the maximum and each point either side then calculating the x,y location of the point with zero slope. If *noInterp* is specified, the peak values are those found in argument 1 without any interpolation. The vector returned by this function has an attached reference which contains the x values of the maximum points. If *xSort* is not specified, the vector is arranged in order of descending y values i.e. largest y value first, smallest last. Otherwise, they are organised in *ascending* x-values.

### 15.5.26 `Maximum(real/complex [, real, real])`

- Arg1 Vector
- Arg2 Start x value
- Arg3 End x value

Returns the largest value found in the vector in the interval defined by start x value and end x value. If the vector is complex the operation will be performed on the magnitude of the vector.

**15.5.27 mean(real/complex)**

Returns the average of all values in supplied argument. If the argument is complex the result will also be complex.

**15.5.28 Mean1(real [, real, real])**

- Arg1 Vector
- Arg2 Start x value. Default: start of vector
- Arg3 End x value. Default: end of vector

Returns the integral of the supplied vector between the ranges specified by arguments 2 and 3 divided by the span (= arg 3 -arg 2). If the values supplied for argument 2 and/or 3 do not lie on sample points, second order interpolation will be used to estimate y values at those points.

**15.5.29 minidx(real/complex)**

Returns index of the array element in argument 1 with the smallest magnitude.

**15.5.30 Minima(real [, real, string])**

- Arg1 Vector
- Arg2 Maximum value. Default  $+\infty$
- Arg3 Options array. Possible values are:
 

xSort	Sort output in order of x values
noInterp	Don't interpolate.

Returns array of values holding every minimum point in the supplied vector whose value is below argument 2. The value returned - if *noInterp* is not specified - is obtained by fitting a parabola to the minimum and each point either side then calculating the x,y location of the point with zero slope. If *noInterp* is specified, the values are those found in argument 1 without any interpolation. The vector returned by this function has an attached reference which contains the x values of the minimum points. If *xSort* is not specified, the vector is arranged in order of ascending y values i.e. smallest y value first, largest last. Otherwise, they are organised in ascending x values.

**15.5.31 Minimum(real/complex [, real, real])**

- Arg1 Vector
- Arg2 Start x value

Returns the smallest value found in the vector in the interval defined by start x value and end x value. If the vector is complex the operation will be performed on the magnitude of the vector.

**15.5.32 norm(real/complex)**

Returns the input vector scaled such that the magnitude of its largest value is unity. If the argument is complex then so will be the return value.

**15.5.33 ph(real/complex) , phase(real/complex)**

Returns the phase of the argument. `ph()` is identical to `phase()` and return the phase in degrees.

The `ph()` and `phase()` functions produces a continuous output i.e. it does wrap from 180 degrees to -180 degrees.

**15.5.34 phase\_rad(real/complex)**

Identical to the `ph()` and `phase()` functions except that the result is in radians.

**15.5.35 Range(real/complex [, real, real])**

Arg1 Vector

Arg2 Start index. Default: 0

Arg3 End index. Default: vector length -1

Returns a vector which is a range of the input vector in argument 1. The range extends from the indexes specified by arguments 2 and 3. If argument 3 is not supplied the range extends to the end of the input vector. If neither arguments 2 or 3 are supplied, the input vector is returned unmodified.

**15.5.36 re(real/complex), real(real/complex)**

Returns the real part of the complex argument.

**15.5.37 Ref(real/complex)**

Returns the reference or x-values of the argument.

**15.5.38 Rms(real)**

Returns a vector of the accumulative RMS value of the input. Unlike `RMS1()` this function returns a vector which can be plotted.

**15.5.39 RMS1(real [, real, real])**

Arg1 Vector

Arg2 Start x value. Default: start of vector

Arg3 End x value. Default: end of vector

Returns the root mean square value of the supplied vector between the ranges specified by arguments 2 and 3. If the values supplied for argument 2 and/or 3 do not lie on sample points, second order interpolation will be used to estimate y values at those points.

**15.5.40 rnd(real)**

Returns a vector with each element a random value between 0 and the absolute value of the argument's corresponding element.

**15.5.41 RootSumOfSquares(real [, real, real])**

- Arg1 Vector
- Arg2 Start x value. Default: start of vector
- Arg3 End x value. Default: end of vector

Similar to RMS1 function but returns the root of the sum without performing an average.

**15.5.42 sign(real)**

Returns 1 if argument is greater than 0 otherwise returns 0.

**15.5.43 sin(real/complex)**

Return sine of argument in radians. Use sin\_deg if the argument is in degrees.

**15.5.44 sqrt(real/complex)**

Returns the square root of the argument. If the argument is real and negative, an error will result. If however the argument is complex a complex result will be returned.

**15.5.45 SumNoise(real [, real, real])**

Identical to the [RootSumOfSquares\(\)](#) function.

**15.5.46 tan(real/complex)**

Return tan of argument in radians. Use tan\_deg if the argument is in degrees.

**15.5.47 Truncate(real [, real, real])**

- Arg1 Data
- Arg2 Start x Value
- Arg3 End x value

Returns a portion of the input vector with defined start and end points. Interpolation will be used to create the first and last points of the result if the start and end values do not coincide with actual points in the input vector.

Arguments 2 and 3 define the beginning and end of the vector.

**Example**

Suppose we have a vector called VOUT which was the result of a simulation running from 0 to 1mS. We want to perform some analysis on a portion of it from 250 $\mu$ S to 750 $\mu$ S. The following call to Truncate would do this:

```
Truncate (VOUT, 250u, 750u)
```

If VOUT did not actually have points at  $250\mu S$  and  $750\mu S$  then the function would create them by interpolation. Note that the function will not extrapolate points before the start or after the end of the input vector.

### 15.5.48 unitvec(real)

Returns a vector consisting of all 1's. Argument specifies length of vector.

### 15.5.49 vector(real)

Returns a vector with length specified by the argument. The value in each element of the vector equals its index.

### 15.5.50 XFromY(real, real [, real, real])

- Arg1 Input vector
- Arg2 Y value
- Arg3 Interpolation order (1 or 2)
- Arg4 Direction of slope. 0 = any, 1 = +ve, -1 = -ve

Returns an array of values specifying the horizontal location(s) where the specified vector (argument 1) crosses the given y value (argument 2) in the direction specified by argument 4. If the vector never crosses the given value, an empty result is returned. The sampled input vector is interpolated to produce the final result. Interpolation order is specified by argument 3.

### 15.5.51 XY(real, real)

Returns a vector with y-values of argument 1 and x-values of argument 2. This function provides a means of creating X-Y plots using the .GRAPH control. See *Simulator Reference Manual/Command Reference/.GRAPH* for details.

### 15.5.52 YFromX(real, real [, real])

- Arg1 Input vector
- Arg2 X value
- Arg3 Interpolation order

Returns an array of values (usually a single value) specifying the vertical value of the specified vector (argument 1) at the given x value (argument 2). If the given x-value is out of range an empty result is returned. The sampled input vector is interpolated to produce the final result. Interpolation order is specified by argument 3.

## Chapter 16

# Monte Carlo, Sensitivity and Worst-case

Monte Carlo analysis is a procedure to assess manufacturing yields by repeating simulation runs with varying applied random variations to part parameters.

Sensitivity analysis repeats runs while perturbing a single parameter for each step. This allows the sensitivity to that parameter of any number of measurements to be evaluated. This makes it possible to identify components and parameters that may require a tight tolerance to maintain a particular specification, or in some cases identify instabilities in the design.

Worst-case analysis attempts to find the combination of component and parameter variances which will lead to the worst possible result. It assigns each component or parameter with a value that is either at the positive extreme or the negative extreme. The decision as to which to use is obtained from the results of a prior sensitivity analysis using the sign of each sensitivity value.

Both Monte Carlo and worst-case may be used to assess production yield and reliability. For many applications, Monte Carlo produces the more realistic results but rarely locates the extremes that are theoretically possible even if statistically unlikely.

It should be noted that worst-case analysis is not guaranteed to locate the worst possible result. The algorithm assumes that the relationship between component or parameter variation and the measured result is linear. This is almost never the case in practice.

The implementation of Monte Carlo analysis in SIMetrix has been designed to be quick to set up for simple cases while still providing the required flexibility for more advanced requirements as might be required for integrated circuit design.

In this chapter we cover the aspects of setting up Monte Carlo, Sensitivity and Worst-case analyses from the front end. This includes setting device tolerances in the schematic, setting up and running the simulation and analysing the results.

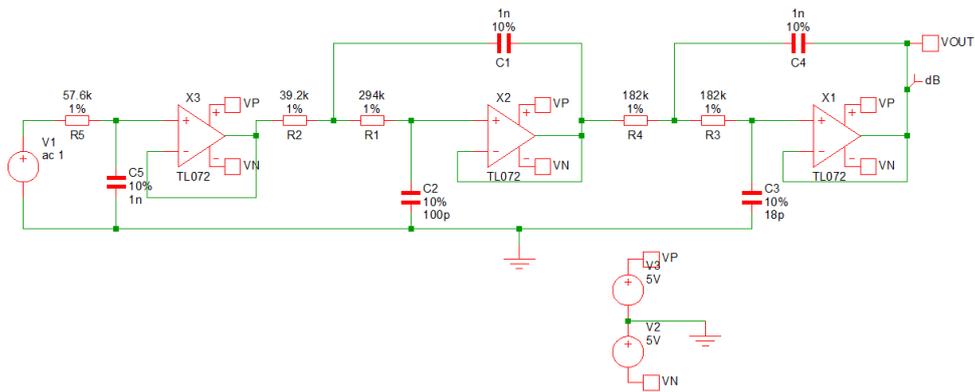
This chapter covers Monte Carlo analysis for SIMetrix (SPICE) simulations. Monte Carlo analysis is also available for SIMPLIS simulations, see [“Multi-step and Monte Carlo Analyses” on page 277](#).

Setting model tolerances is not covered here but in the Monte Carlo Analysis chapter in the *Simulator Reference Manual/Monte Carlo, Sensitivity and Worst-case/Specifying Tolerances*.

## 16.1 A Monte Carlo Example

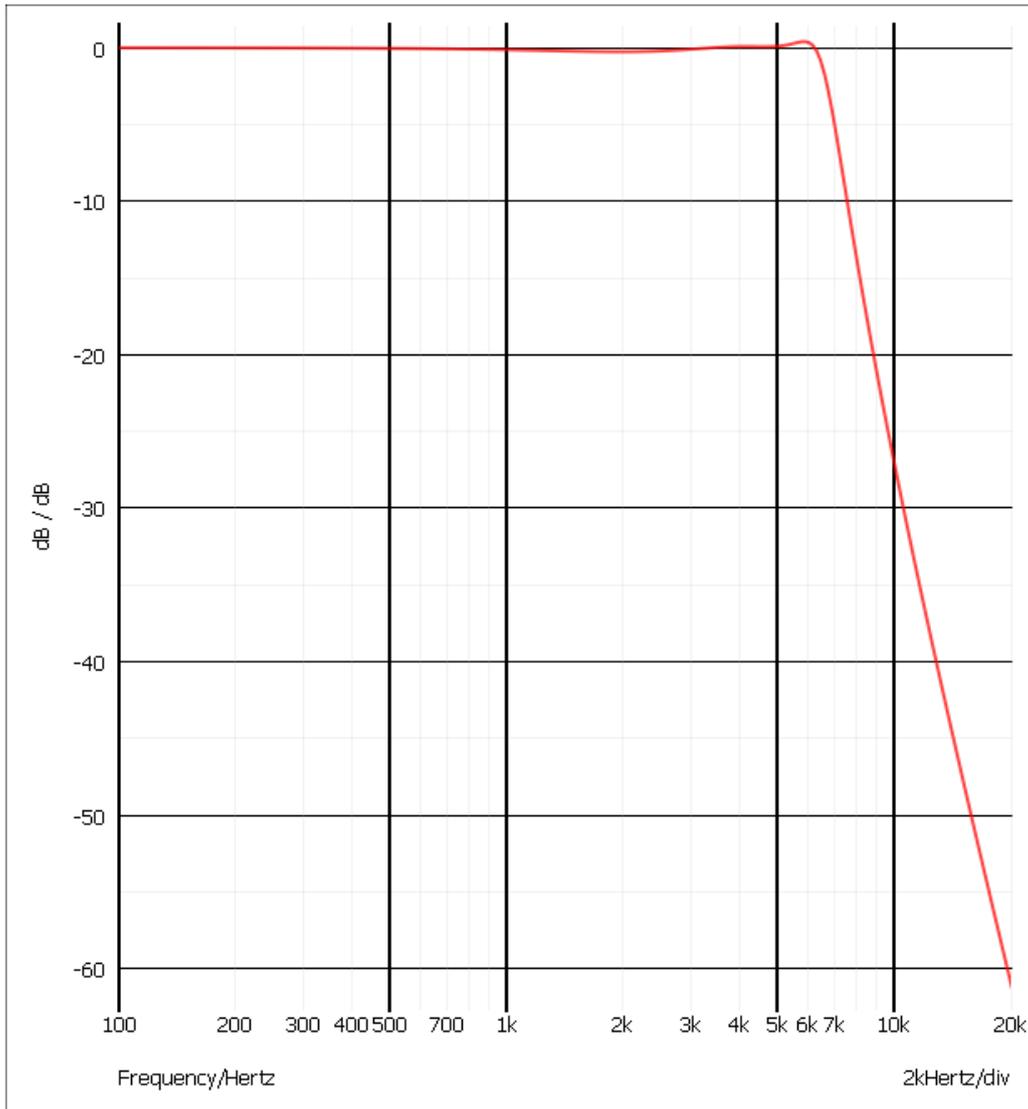
Consider the following active filter circuit.

## 16.1. A Monte Carlo Example



This circuit can be found in `EXAMPLES/MonteCarlo/cheb2.sxsch`

The circuit is a 5th order low-pass 7kHz Chebyshev filter with a 0.23dB passband ripple specification. Its nominal response is:



This circuit is to be used in an application that requires the gain of the amplifier to remain within 3dB of the dc value from 0 to 6kHz. A 0.23dB ripple specification therefore seems a reasonable choice. Clearly though the tolerance of the capacitors and resistors may upset this. To investigate, a Monte Carlo analysis may be performed. The standard part tolerances are 10% for capacitors and 1% for resistors. With the example circuit the tolerances are already applied but the procedure for doing this is as follows:

1. Select menu item **Monte Carlo | Set All Resistor Tolerances**
2. Enter 1%. (The % is recognised)
3. Select menu item **Monte Carlo | Set All Capacitor Tolerances**
4. Enter 10%.

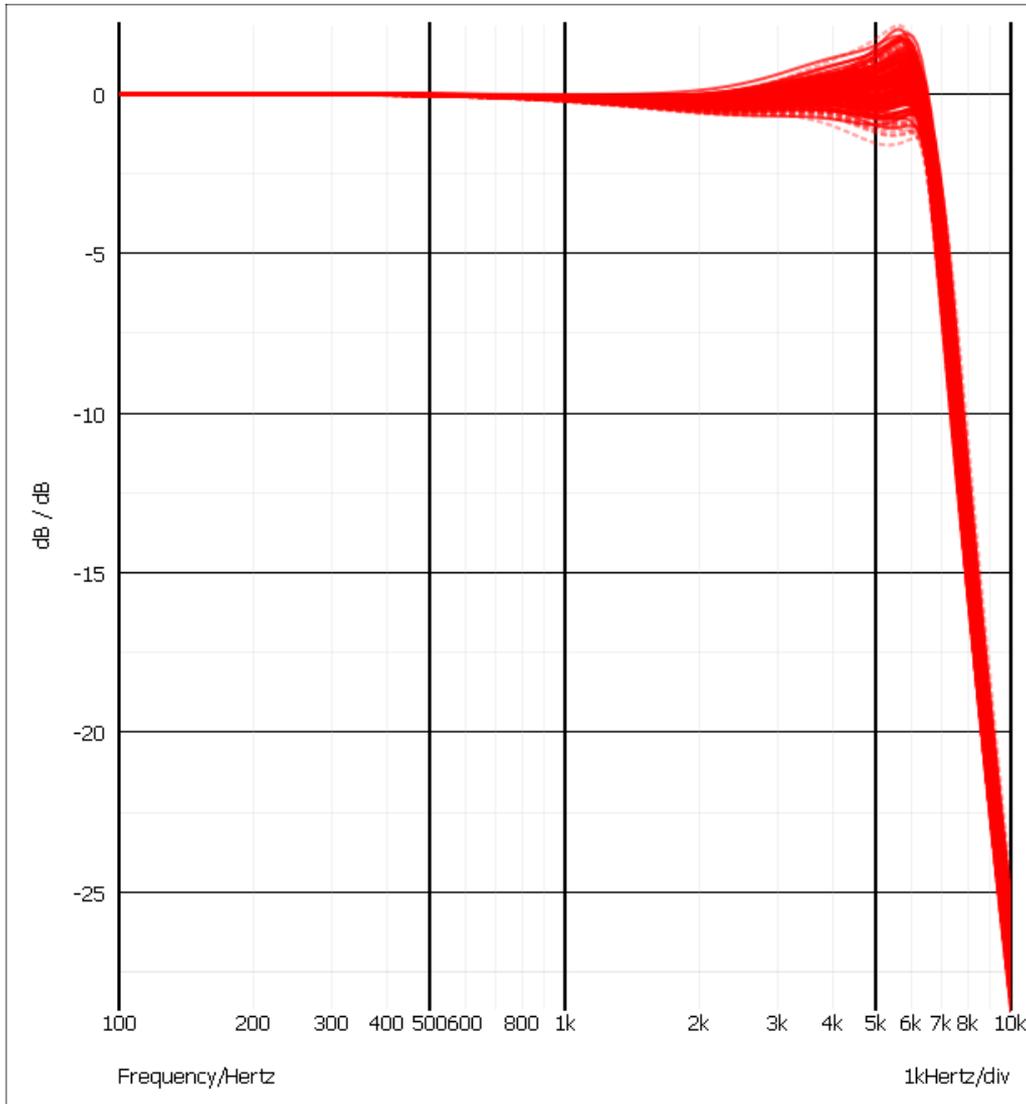
The example circuit has already been set up to run 100 steps of Monte Carlo. To view the settings:

1. Select menu **Simulator | Choose Analysis...**
2. Note in the section Monte Carlo and Multi-step Analysis the Enable multi-step box is checked.
3. Press the Define... button.

4. Note that in the Sweep Mode section, Monte Carlo is selected and in the Step Parameters section. Number of steps has been set to 100.

Start the analysis in the usual way. It should take less than 1 second to run.

The analysis will be repeated 100 times. Now plot the output of the filter in the usual way (**Probe AC/Noise | dB - Voltage...**). The result is the following:



As can be seen, the specification is generally met. In the next section [“Sensitivity/Worst-case Example” on page 415](#) we will perform a worst-case analysis on the circuit which will show that there are extreme combinations of values that do not meet the spec.

The SIMetrix Monte Carlo analysis implementation has many more features such as:

- Random variation of device model parameters.
- Support for matched devices.
- Log file creation
- Seed selection to allow repeated runs with same randomly applied values.

## 16.2 Sensitivity/Worst-case Example

The same circuit used for the Monte Carlo example may also be set up to run a Sensitivity and Worst-case analysis.

1. Open the schematic EXAMPLES/MonteCarlo/cheb2.sxsch if not already
2. Select menu Simulator | Choose Analysis...
3. In the AC sheet, in the Monte Carlo and multi-step analysis group, click the Define... button.
4. Click on Sensitivity/Worst case in the Sweep mode group.
5. Enter "minimum(mag(vout),0,6k)" as the first entry in the Sensitivity measurement functions group.
6. Check the Run worst case analysis button.
7. Ok dialog boxes, then run simulation in the usual way. You should now see this:

**Define Multi Step Analysis**

**Sweep mode**

- Device
- Parameter
- Model parameter
- Temperature
- Frequency
- Monte Carlo
- Snapshot
- Sensitivity/Worst case

**Step parameters**

Start value: 1k  Decade

Stop value: 2k  Linear

Number of steps: 100  List

Group digital curves Define List...

Number of Cores: 1

Number of physical cores: 48

Number of cores allowed by license: 16

**Sensitivity parameters**

**Sensitivity measurement functions**

minimum(mag(vout),0,6k) Functions...

For multi-step sensitivity analysis, the measurement functions must return a scalar value from a vector. E.g. Mean1(VOUT) returns the mean of the signal at node VOUT

At least one measurement function should be supplied. Up to two may be entered here. More can be added manually in the F11 window

Run worst case analysis

Perturbation: 100%

**Parameters**

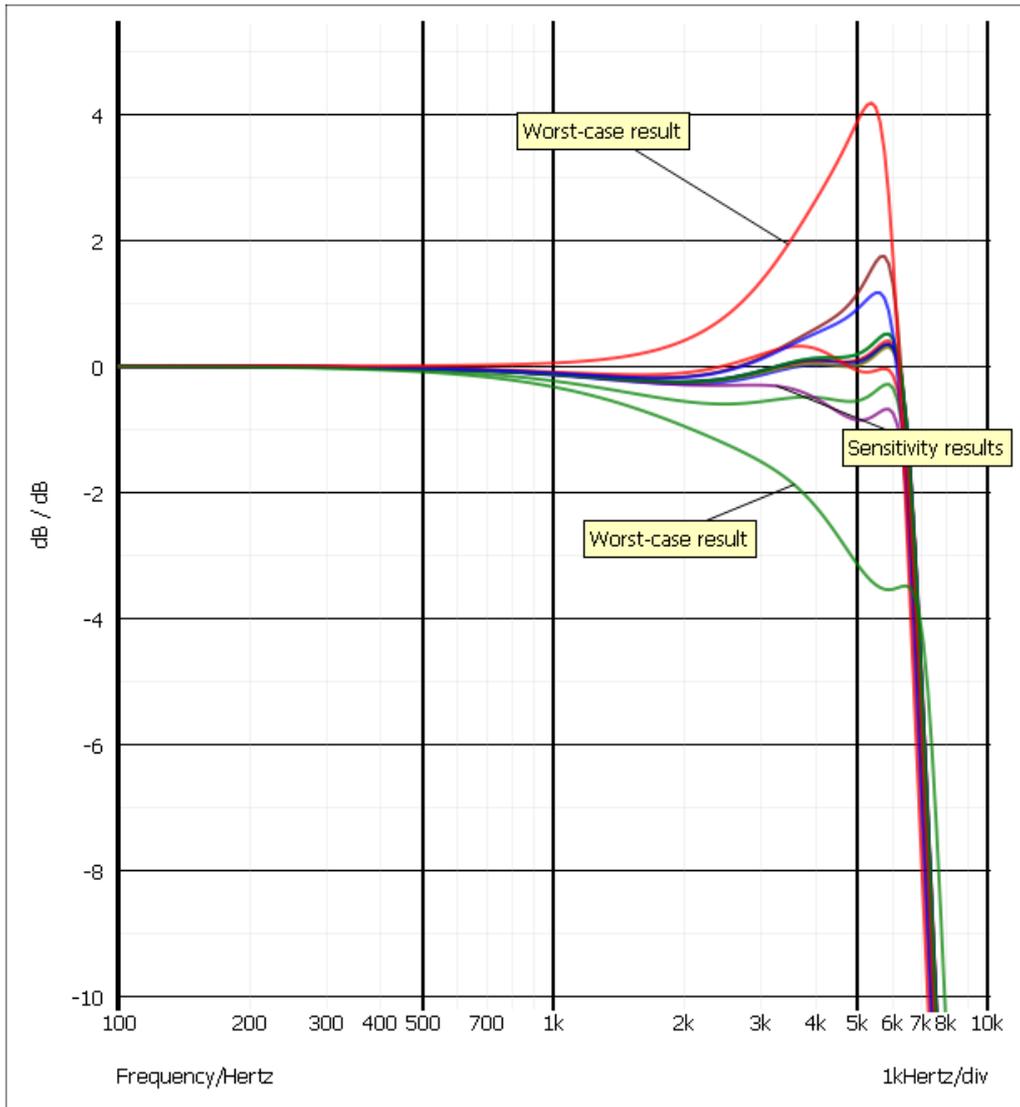
Device name:

Parameter name:

Enable Monte Carlo log file

Ok Cancel Help

Run the analysis in the usual way. You should see a graph like the following:



The outlying green and red curves are the worst-case results while the remaining curves are the sensitivity results. Each of the sensitivity curves is the result of running the simulation with a single component value perturbed so that it is set to its highest value within the tolerance specification. The worst-case curves are the result of running the simulation with all components perturbed at one or other tolerance extreme. Which extreme to use is determined from an analysis of the sensitivity results so as to maximise and minimise the sensitivity measurement function.

Note that the results show deviation from our 3dB specification. So, although our previous Monte Carlo analysis showed that typical production yields should be within our specification, there are some extreme combinations of components that will lead to a result substantially outside the specification.

To resolve this we can tighten the specification of some components and to do this we first examine the sensitivity report. The report details the results of evaluating the sensitivity measurement functions along with the component values used.

To view the sensitivity report, select menu **Simulator | Reports | Sensitivity**. You should see the following:

1	2	3	4	5	6	7	8
Run ID	Results			Perturbation			
Case #	Reference	Minimum(mag(vout), 0, 6000)	Delta	Norm	Lot item	Nominal	Value
1	nominal	0.9706977	0	0		0	0
2	Ref: C5	0.9330613	-0.0376365	-0.403365	C5	1e-09	1.1e-09
3	Ref: R5	0.967381	-0.00331675	-0.342859	R5	57600	58176
4	Ref: R1	0.9706362	-6.15196e-05	-0.00633807	R1	294000	296940
5	Ref: R3	0.9716089	0.000911131	0.0937755	R3	182000	183820
6	Ref: C2	0.9058439	-0.0648538	-0.715949	C2	1e-10	1.1e-10
7	Ref: C4	0.9792253	0.00852758	0.0870849	C4	1e-09	1.1e-09
8	Ref: R2	0.9721105	0.00141274	0.145327	R2	39200	39592
9	Ref: C1	0.9844947	0.013797	0.140143	C1	1e-09	1.1e-09
10	Ref: R4	0.9716222	0.000924451	0.0951451	R4	182000	183820
11	Ref: C3	0.9785062	0.00780844	0.0797996	C3	1.8e-11	1.98e-11

The **Delta** column shows the deviation from the nominal for each case. Double-click the **Delta** label in the header. This will sort the results in order of magnitude and so show which cases had the greatest effect. Note that C2 and C5 are at the top of the list. We will now tighten the tolerance on these parts to 2%. To do this:

1. Select C2 and C5
2. Select menu Mote Carlo | Set Selected Component Tolerances...
3. Enter 2% then Ok to close
4. Re-run simulation

You should see that the result is now within specification. Changing R5 and C5 to 576K and 100p respectively does not significantly affect the result and means that there is only one tight-tolerance capacitor value in the circuit of 100p.

## 16.3 Part Tolerance Specification

In this section we will only cover the simple case of how to specify tolerances on devices at the schematic level for use with Monte Carlo, sensitivity and worst-case analyses. SIMetrix has much more comprehensive features for specifying tolerances. For complete documentation on tolerance specification please refer to the *Simulator Reference Manual/Monte Carlo, Sensitivity and Worst-case/Specifying Tolerance*.

### 16.3.1 Setting Device Tolerances

To select individual device tolerances proceed as follows:

1. Select part or parts whose tolerances you wish to be the same. (You can individually select parts by holding the **control** key down and left clicking on each).

2. Select menu **Monte Carlo | Set Selected Component Tolerances...** and enter tolerance in the dialog box. You may use the '%' symbol here if you wish, so 5% and 0.05 have the same effect. (Note: this is the only place that '%' is recognised - you can't use it netlists or models).

If all the resistors or all the capacitors in a circuit are to have the same tolerance, select either **Monte Carlo | Select All Capacitor Tolerances** or **Monte Carlo | Select All Resistor Tolerances**.

Device tolerances can be applied to the following parts:

- Capacitors
- Resistors
- Inductors
- Fixed voltage sources
- Fixed current sources
- Voltage controlled voltage sources
- Voltage controlled current sources
- Current controlled voltage sources
- Current controlled current sources
- Lossless transmission lines (applied to Z0 parameter)

Device tolerance will be ignored for other devices.

### 16.3.2 Model Tolerances

Tolerances may be applied to .MODEL statements using a distribution function. For example:

```
.MODEL NPN1 NPN IS=1.5e-15 BF={180*GAUSS(0.5)}
```

The GAUSS function above returns a random number base around 1.0 with a  $3\sigma$  Gaussian distribution of 0.5. (Gaussian distribution with standard deviation of 0.5/3).

There are many other distribution functions including a user defined function which can be used for any distribution. In addition you can use parameter definitions to define random variables to create matched devices and parameters. Full details may be found in the *Simulator Reference Manual/Monte Carlo, Sensitivity and Worst-case/Specifying Tolerances*.

### 16.3.3 Matching Devices

Some devices such as resistor networks are constructed in a manner that their tolerances track. Such devices often have two specifications one is an absolute tolerance and the other a matching tolerance. A thin film resistor network might have an absolute tolerance of 1% but a matching tolerance of 0.05%. This means that the resistors will vary over a +/-1% range but will always be within +/-0.05% of each other.

To specify matched devices for Monte Carlo analysis two pieces of information are required. Firstly, the parts that are matched to each other must be identified and secondly their matching tolerances need to be specified.

#### To Identify Matched Devices

- Select the parts you wish to match to each other. (Use **control** key to select multiple parts.)
- Select menu item **Monte Carlo | Match Selected Devices**
- You must now supply a lot name which must be unique. You can use any alphanumeric name.

## Matching Tolerances

To specify device match tolerances, proceed as follows:

- Select the parts you wish to match to each other. (Use **control** key to select multiple parts.)
- Select menu item **Monte Carlo | Set Match Tolerances**
- Enter the desired tolerance.

If using device tolerance parameters, note that any absolute tolerance specified must be the same for all devices within the same lot. Any devices with the same lot name but different absolute tolerance will be treated as belonging to a different lot. For example if a circuit has four resistors all with lot name RN1 but two of them have an absolute tolerance of 1% and the other two have an absolute tolerance of 2%, the 1% resistors won't be matched to the 2% resistors. The 1% resistors will however be matched to each other as will the 2% resistors. This does not apply to match tolerances. It's perfectly OK to have devices with different match tolerances within the same lot.

### 16.3.4 Random Distribution

The default distribution for device tolerances is Gaussian with the tolerance representing a  $3\sigma$  spread. This can be changed to rectangular using two simulator options. These are:

MC_ABSOLUTE_RECT	If set absolute tolerances will have a rectangular distribution.
MC_MATCH_RECT	If set matching tolerances will have a rectangular distribution.

Distributions can be specified on a per part basis or even a per parameter basis by using distribution functions in an expression. See *Simulator Reference Manual/Monte Carlo, Sensitivity and Worst-case/Specifying Tolerances* for details.

## 16.4 Running Monte Carlo

### 16.4.1 Overview

There are actually two types of Monte Carlo analyses. These are:

1. Single step Monte Carlo sweep
2. Multi step Monte Carlo run.

1. above is applicable to AC, DC, Noise and Transfer Function analyses. 2. can be applied to the same analyses in addition to transient analysis.

An example of 1. can be seen in the figure in [“Monte Carlo” on page 233](#). This was a run where the gain at a single frequency was calculated 1000 times with the Monte Carlo tolerances applied. This used AC analysis with the Monte Carlo sweep mode - one of the six modes available. Only a single curve is created hence the name *single step*.

An example of 2 is the example at the beginning of this chapter. Here a complete frequency sweep from 1kHz to 100kHz was repeated 100 times creating 100 curves.

## 16.4.2 Setting up a Multi Step Monte Carlo Run

1. Select schematic menu **Simulator | Choose Analysis....** Select the AC, DC, Noise, Transient or TF tab as required.
2. Define the analysis as required.
3. In the Monte Carlo and Multi-step Analysis section, check the Enable multi-step box then press the Define... button. This will open:-

**Define Multi Step Analysis**

**Sweep mode**

- Device
- Parameter
- Model parameter
- Temperature
- Frequency
- Monte Carlo
- Snapshot
- Sensitivity/Worst case

**Step parameters**

Start value: 1k  Decade

Stop value: 2k  Linear

Number of steps: 10  List

Group digital curves

Number of Cores: 1

Number of physical cores: 48

Number of cores allowed by license: 16

**Sensitivity parameters**

**Sensitivity measurement functions**

For multi-step sensitivity analysis, the measurement functions must return a scalar value from a vector. E.g. Mean1(VOUT) returns the mean of the signal at node VOUT

At least one measurement function should be supplied. Up to two may be entered here. More can be added manually in the F11 window

Run worst case analysis

Perturbation: 100%

**Parameters**

Device name:

Parameter name:

Enable Monte Carlo log file

4. In the Sweep mode section, select Monte Carlo.
5. In the Step Parameters section, enter the number of steps required.

6. You can set the **Number of Cores** to a number large than 1 if your system and license permit it. This will substantially speed up the run. For more information see [“Using Multiple Cores for Multi-step Analyses”](#) on page 262.

### 16.4.3 Setting up a Single Step Monte Carlo Sweep

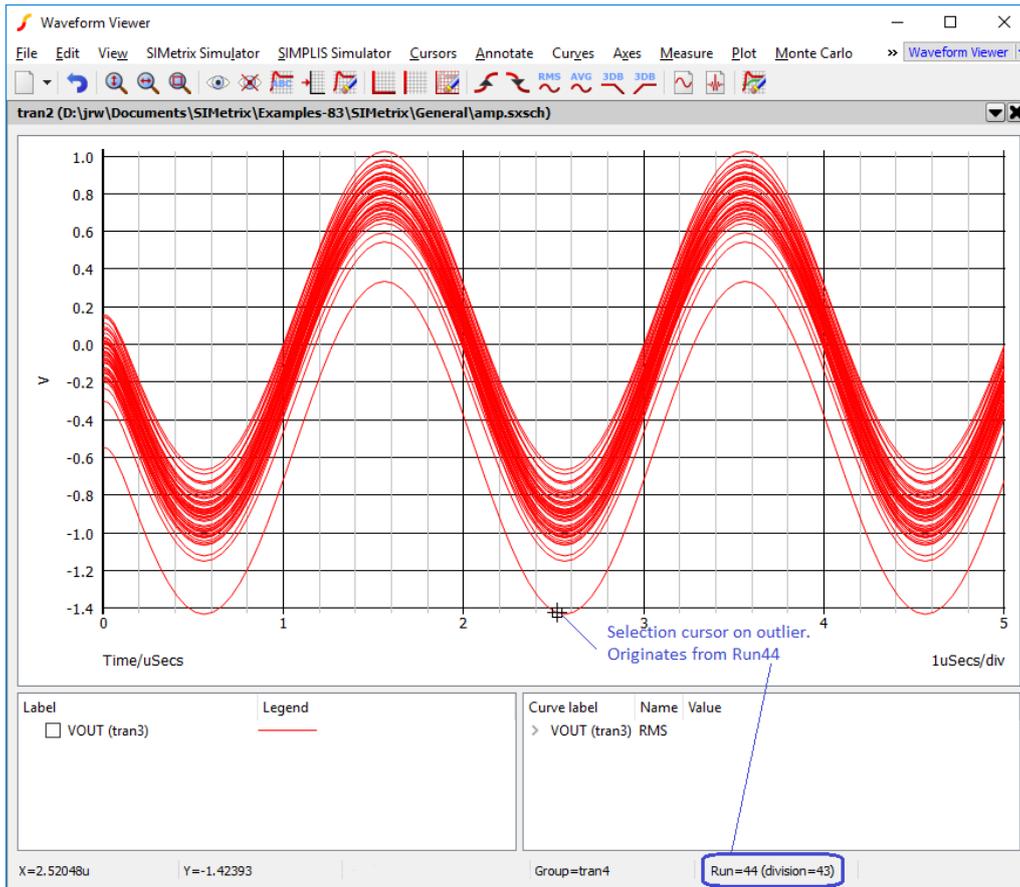
1. Select schematic menu **Simulator | Choose Analysis....** Select the AC, DC, Noise or TF tab as required.
2. In the **Sweep Parameters** section, press the Define... button.
3. In the Sweep Mode section select Monte Carlo.
4. In the Parameters section enter the required value for the Number of points.
5. For AC, Noise and TF, you must also supply a value for Frequency.

### 16.4.4 Setting the Seed Value

The random variations are created using a pseudo random number sequence. The sequence can be seeded such that it always produces the same sequence of numbers for a given seed. In Monte Carlo analysis, the random number generator is seeded with a new value at the start of each run and this seed value is displayed in the log file.

Using the seed value you can repeat a run that perhaps produced unexpected results. Obtain the seed used for that run, then repeat with the seed value but doing just a single run. You will then be able to probe around the circuit and plot the results for just that run. The procedure to do this is as follows:

1. Identify the Monte Carlo step that you wish to repeat. The picture below shows how to obtain the *Run number* from a Monte Carlo plot.



2. Open the Monte Carlo log report using menu **Simulator | Reports | Monte Carlo**. In the *Seed Table* locate the run number obtained in step 1. From this read the *seed value*.
3. Select schematic menu **Simulator | Choose Analysis....**
4. Select **Options** tab and enter the seed value in the Monte Carlo section.

The first run of each Monte Carlo analysis will use the same random values as the run from which you obtained the seed value in the log file. Note this assumes that only changes in values are made to the circuit. Any topology change will upset the sequence.

## 16.5 Running Sensitivity and Worst-case

### 16.5.1 Overview

As with Monte Carlo, there are both Multi-step and single-step variants of sensitivity and worst-case analyses. Single-step can be applied to AC, DC, Noise and transfer function analyses; multi-step can be applied to the same analyses in addition to transient analysis.

The example shown above (“[Sensitivity/Worst-case Example](#)” on page 415) is an example of a multi-step analysis and is the most commonly used.

Single-step analysis can be used when the measurement being performed can be done on a single point. For example, the gain of a linear amplifier at a single frequency can be obtained from a single point of an AC analysis. Similarly the, DC offset of an amplifier can be evaluated from a single point of a DC

analysis. If these are the quantities being studied then a single-step sensitivity analysis may be used with major speed advantages.

## 16.5.2 Setting up a Multi-step Sensitivity Analysis

1. Select schematic menu **Simulator | Choose Analysis....** Select the AC, DC, Noise, Transient or TF tab as required.
2. Define the analysis as required.
3. In the Monte Carlo and Multi-step Analysis group, check the Enable multi-step box then press the Define... button. This will open:-

**Define Multi Step Analysis**

**Sweep mode**

Device  
 Parameter  
 Model parameter  
 Temperature  
 Frequency  
 Monte Carlo  
 Snapshot  
 Sensitivity/Worst case

**Step parameters**

Start value: 1k  
 Stop value: 2k  
 Number of steps: 10  
 Group digital curves  
 Decade  
 Linear  
 List  
 Define List...  
 Number of Cores: 1  
 Number of physical cores: 48  
 Number of cores allowed by license: 16

**Sensitivity parameters**

**Sensitivity measurement functions**

Functions...

For multi-step sensitivity analysis, the measurement functions must return a scalar value from a vector. E.g. Mean1(VOUT) returns the mean of the signal at node VOUT

At least one measurement function should be supplied. Up to two may be entered here. More can be added manually in the F11 window

Run worst case analysis  
 Perturbation: 100%

**Parameters**

Device name:   
 Parameter name:   
 Enable Monte Carlo log file

Ok Cancel Help

4. In the Sweep mode group, select Sensitivity/Worst-case.
5. In the Sensitivity parameters group, enter at least one sensitivity measurement function. For multi-step analyses, this must be a function that returns a scalar value from a vector. For example, the function `mean1()` returns the mean of its argument.
6. You can set the Number of Cores to a number large than 1 if your system and license permit it. This will usually substantially speed up the run except for small circuits. For more information see [“Using Multiple Cores for Multi-step Analyses”](#) on page 262.

### 16.5.3 Setting up a Single Step Sensitivity Analysis

1. Select schematic menu **Simulator | Choose Analysis....** Select the AC, DC, Noise or TF tab as required.
2. In the **Sweep Parameters** group, press the Define... button.
3. In the Sweep Mode group select Sensitivity/Worst-case.
4. In the Sensitivity parameters enter at least one sensitivity measurement function. For single step, the analysis needs to obtain a measurement for each simulation point, so you must enter an expression that returns a vector from the sweep. This is usually simply the value of a node voltage or current. For AC analyses, you must apply a complex-real conversion function, e.g. `db()`, `mag()`, `phase()`, `real()` or `imag()`.
5. For AC, Noise and TF, you must also supply a value for Frequency.
6. If you wish a worst-case analysis to be performed after the sensitivity analysis, check the Run worst-case analysis box.

## 16.6 Analysing Monte-Carlo Results

### 16.6.1 Plots

Plots of Monte Carlo analyses are performed in exactly the same manner as for normal runs. When you probe a circuit point, curves for each run in the MC analysis will be created.

In common with other multi-step analyses, the curves generated for multi-step Monte Carlo are grouped and operations performed on them treat the group as a single entity. You can break the group using menu **Curves | Ungroup Selected Curves**.

### Identifying Curves

Sometimes it is useful to know exactly which run a particular curve is associated with. To do this, move the cursor so that it is on a curve of interest. The selection cursor (small square box) should locate itself on the curve. In the status box at the bottom of the graph window, you should see a label in the form `Run=nnn (division=ddd)` where `nnn` and `ddd` are integers. (`ddd` is always 1 less than `nnn`). The run number can be cross-referenced with the entry in the log file to find information about the run that created the curve. You can also obtain the seed value used so that the run can be repeated. See [“Setting the Seed Value”](#) on page 422 and [“Log File”](#) on page 426 for more details.

### Plotting a single Curve

If you wish to plot a single curve in a Monte Carlo set, you must obtain the division number (always 1 less than the run number) then use the **Probe | Add Curve...** menu to plot an indexed expression. We use an example to explain the process.

Using the Chebyshev filter example, let's suppose that we wish to plot the curve of the filter output created by run 49 (division 48) alone without the remaining curves. Proceed as follows.

1. Run the chebyshev filter example as explained at the beginning of this chapter.
2. Select menu **Probe | Add Curve...**
3. Click on the output of the filter. You should see C4\_P entered in the Y expression box.
4. You must now modify the expression you have entered to give it an index value. For the simple case of a single voltage or current just append it with

```
[division-index]
```

where *division-index* is the run number less 1. In this example the run number is 49 so we enter 48 for the index. You should now have:

```
C4_P [48]
```

displayed in the Y expression box.

5. Close box. You should see a single curve plotted.

An alternative method of plotting single curves is given in [“Setting the Seed Value” on page 422](#).

## 16.6.2 Creating Histograms

See [“Performance Analysis and Histograms” on page 374](#).

## 16.6.3 Log File

Multi-step Monte Carlo analyses generate a log file that documents the values of all tolerated parts. It will also record the seed value used for each step. To open the log file, select menu **Monte Carlo | View Log File** or **Simulator | Reports | Monte Carlo**. The actual log data is stored in an XML data file and if there are multiple cores used, each core generates its own file. The above menus will compile an HTML report using those files as its input data and then display it. For a long run on a large circuit, this process may take a few seconds.

# 16.7 Analysing Sensitivity and Worst-case Results

## 16.7.1 Plots

Plots of sensitivity and worst-case analyses are performed in exactly the same manner as for normal runs. When you probe a circuit point, curves for each run in the analysis will be created. For multi-step sensitivity analysis, there will be one curve for each sensitivity case. For multi-step worst case analysis, there will be two curves for each sensitivity measurement function provided. One curve maximises the sensitivity function and the other minimise the sensitivity function.

In common with other multi-step analyses, the curves generated are grouped and operations performed on them treat the group as a single entity. You can break the group using menu **Curves | Ungroup Selected Curves**.

## 16.7.2 Reports

### Sensitivity Analysis Interactive Table

The sensitivity analysis interactive table may be opened using menu **Simulator | Reports | Sensitivity (Interactive Table)** . This will show something like this:

1	2	3	4	5	6	7	8
Run ID		Results			Perturbation		
Case #	Reference	lpbw(vout, 3)	Delta	Norm	Lot item	Nominal	Value
1	nominal	3458324	0	0		0	0
2	Lot: inres	3442031.9	-16292	-0.236663	R6	100	102
					R5	1000	1020
					R4	1000	1020
					R3	100	102
3	Ref: R8	3464251.1	5927.19	0.0855479	R8	10000	10200
4	Ref: Q2	3447958.3	-10365.7	-0.00601266	Q2.bf	480	720
5	Ref: Q6	3455446.4	-2877.51	-0.00166549	Q6.bf	330	495
6	Ref: R12	3458754	430.048	0.0248672	R12	1000	1005
7	Lot: outres	3451503.6	-6820.3	-0.0988019	R13	1000	1020
					R12	1000	1020
					R10	1000	1020
8	Ref: Q1	3457300.2	-1023.72	-0.000592208	Q1.bf	480	720
9	Ref: Q5	3457343.9	-980.072	-0.000566951	Q5.bf	330	495
10	Ref: R14	3462109.2	3785.25	0.0546669	R14	100	102
11	Ref: R2	3452312.9	-6011.03	-0.087058	R2	1000	1020
12	Ref: Q4	3456305.2	-2018.8	-0.00116818	Q4.bf	480	720
13	Ref: R3	3463133.2	4809.3	0.277743	R3	100	100.5
14	Ref: R4	3450492.6	-7831.39	-0.453929	R4	1000	1005
15	Ref: R13	3460025	1701.09	0.0983284	R13	1000	1005
16	Ref: Q7	3458505.5	181.515	0.000104967	Q7.bf	480	720
17	Ref: R7	3461302.4	2978.47	0.0430253	R7	100	102
18	Ref: R1	3437957.2	-20366.8	-0.296204	R1	1000	1020
19	Ref: R10	3454487.3	-3836.67	-0.222127	R10	1000	1005
20	Ref: R9	3439920.3	-18403.6	-0.267501	R9	1000	1020
21	Ref: R11	3455429.7	-2894.29	-0.0418803	R11	100	102
22	Ref: R5	3453121.6	-5202.36	-0.301314	R5	1000	1005
23	Ref: Q3	3449306.2	-9017.73	-0.00522872	Q3.bf	480	720
24	Ref: R6	3462487.3	4163.37	0.240484	R6	100	100.5

Double-clicking the headings will sort the values in order of magnitude or alphabetically in the case of the Reference column. Sorting is inactive for the Perturbation columns.

The report provides the results of evaluating the sensitivity measurement functions along with perturbation data. In the example above there are 8 columns, made up of 2 columns identifying the case, 3 columns for perturbation data and 3 columns for measurement function data. In general there will always be 5 columns plus 3 for each measurement function supplied. The following table explains the purpose of each column:

Case #	Case number. This can also be read as a cursor readout from the graph allowing cross-referencing between the graph and the this report
Reference	A label identifying exactly what was perturbed in this case. This may be a single component value but may also be a model parameter (e.g. case 4) or a group of matched items referenced by their lot name (e.g. case 2 and 7)
Measurement function	The actual column heading will be the expression used. This is the result of evaluating the given sensitivity measurement function
Delta	The difference between the measurement function value for this case and that obtained from the nominal case. Double clicking the header will sort the table according to the magnitude of the value in this column
Norm	Normalised sensitivity A value that indicates the sensitivity of the item being perturbed to the measurement result, independent of the level of perturbation. The value is calculated from $\Delta/\text{measurement} \times \text{nominal-value}/\text{perturbation}$ . This value can be used to identify components that are unduly sensitive and which could indicate a design flaw. Note that this value does not have much meaning for measurement results that are nominally zero such as the offset of a differential amplifier. Double clicking the header will sort the table according to the magnitude of the value in this column
Lot item	Single item being perturbed. This may be the same as the reference column if only a single item is being perturbed in the case. But if there are items grouped together in a lot, each item will be listed. In case 2 in the example above, there are 4 resistors perturbed together as these are 4 matched devices in a network.
Nominal Value	Nominal value of single item being perturbed Actual value used in this case

## Sensitivity Analysis HTML Report

The menu **Simulator | Reports | Sensitivity (HTML)** will create and display a sensitivity HTML report. This shows the same information as the interactive table but its sort order is fixed. The HTML file is created by this process and will be available in the current working directory with the name of sensitivity.html.

The styles used for the HTML file are defined in a .CSS file. You can specify your own if desired. See [“File Location Options” on page 451](#).

## Worst-case Analysis Report

The sensitivity analysis report may be opened using menu **Simulator | Reports | Worst-case**.

This shows the values used for each worst-case analysis as well as the results of calculating the sensitivity measurement functions.

The styles used for the HTML file are defined in a .CSS file. You can specify your own if desired. See [“File Location Options” on page 451](#).

# Chapter 17

## Verilog-HDL Simulation

### 17.1 Overview

The Verilog-HDL feature provides the ability to simulate Verilog digital designs included in analog circuits. The SIMetrix implementation uses an external Verilog simulator to achieve this and communicates with that simulator using the standard VPI programming interface. Two open-source Verilog simulators are supplied for this purpose and these are installed with the SIMetrix installer. In principle, any VPI compliant Verilog simulator may be used, however, only the two open source programs supplied are tested and supported.

Interfacing an analog simulator with a digital HDL simulator is a challenging task and imposes some trade-offs between timing precision and simulation speed. In particular, circuits where a digital section sits inside an analog feedback loop are especially demanding. As SIMetrix is an analog tool for analog designers, we have focussed on providing maximum accuracy. This imposes the need for rapid communication between the analog simulator and the Verilog simulator and for this we have developed our own inter-process communication method as the standard system supplied techniques were too slow.

### 17.2 Documentation

In this chapter we show how to use the Verilog-HDL simulation feature using the schematic editor. Reference documentation for the underlying simulator device that implements the Verilog-HDL interface can be found in *Simulator Reference Manual/Analog Device Reference/Verilog-HDL Interface (VSXA)*.

### 17.3 Supported Verilog Simulators

We supply two alternative open source Verilog simulators, namely GPL Cver (Pragmatic C Software) and Icarus Verilog (Stephen Williams). These simulators are installed and configured ready to use.

The configuration of the external simulator is user definable and other VPI compliant simulators can be setup.

### 17.4 Basic Operation

To support Verilog designs, SIMetrix has a new device called VSXA. A VSXA device is defined by a .MODEL statement and this in turn specifies a Verilog design file. The Verilog file is expected to contain a

top level module definition and this module defines the external connections to the analog system via Verilog ports.

Any number of Verilog devices (i.e. VSXA instances) can be placed in a SIMetrix netlist/schematic. The actual design presented to the Verilog simulator will be a single Verilog definition, but SIMetrix handles the task of creating this from the user's individual Verilog design files and schematic/netlist interconnection of VSXA instances.

See Simulator Reference Manual/Analog Device Reference/Verilog-HDL Interface (VSXA)simref:AnalogDeviceReference:Verilog-HDLInterface(VSXA) for more details about the VSXA device.

## 17.5 Using Verilog-HDL in SIMetrix Schematics

### 17.5.1 Creating Schematic Symbols

The SIMetrix schematic editor provides a feature that will create and place a schematic symbol from a Verilog file. This feature reads the Verilog file and determines the inputs and outputs along with the names of the ports. It also reads any parameters defined. From this information it creates a symbol with inputs on the left and outputs on the right. It also creates an edit facility to edit any parameters defined in the Verilog module.

To create a schematic symbol from a Verilog design, proceed as follows:

1. In the schematic editor, select menu **Help | Construct Verilog-HDL Symbol**.
2. Navigate to the Verilog design file. SIMetrix expects the file extension `.v` or `.vl`. Select the file then close.
3. You should see an image of the symbol ready to place. Place in the usual way.
4. If there are errors in the Verilog file, you will see a message in the form:

```
"*** ERROR *** Cannot parse verilog design file 'filename'. For
details see log file 'filename.log' Cannot parse Verilog-HDL file.
No symbol created."
```

The log file should list details of the error. This file is generated by the GPL Cver Verilog simulator and will contain additional information that can obscure the desired error message. Verilog errors must be rectified before SIMetrix can create a symbol.

### 17.5.2 Editing Parameters

The symbol creation feature described above builds the necessary functionality in the symbol to allow GUI editing of the device's parameters. To use this, just edit the schematic instance in the usual way by double clicking or selecting followed by **F7**.

You will see a dialog box showing a number of parameters. The first parameters starting with 'Voltage input logic zero threshold' and ending with 'Threshold time tolerance' along with the check boxes 'Disable output of non-analog vectors' and 'Disable Module Cache' are built-in parameters that are defined for all Verilog devices. Any parameters defined within the Verilog definition will be shown in addition to these and listed after 'Threshold time tolerance'.

## 17.6 Module Cache

### 17.6.1 Operation

Before starting a simulation and also when creating a symbol from a Verilog design, SIMetrix needs to gather some information about each Verilog module used in the circuit. It does this by starting a Verilog simulation then interrogating the Verilog simulator via VPI. This process can take some time if there are many Verilog modules in the circuit. To speed things up, SIMetrix caches the information obtained for future use.

The cache mechanism calculates the MD5 checksum of the Verilog file and stores this with the cached information in the cache file. When the cached information is required, SIMetrix calculates the MD5 checksum of the Verilog file and looks to see whether there is a cache item with that MD5 value. If there is, it will use the cached data. If not it will retrieve the information via the Verilog simulator.

For more information about the Module cache, see Simulator Reference Manual/Analog Device Reference/Verilog-HDL Interface (VSXA)simref:AnalogDeviceReference:Verilog-HDLInterface(VSXA)

## 17.7 Simulation Options

There are three Verilog simulation options available through the user interface. These can be accessed from the Choose Analysis dialog box as follows:

1. Select menu **Simulator | Choose Analysis...**
2. Select the Options tab
3. See options under Verilog-HDL Options

### 17.7.1 Verilog Simulator

This option allows you to select the Verilog simulator used for the main simulation. There will be a default choice of 'CVER' or 'Icarus'.

Note that the Verilog simulator is also used to enumerate the ports and parameters of a Verilog module separately from the main simulation. This task is always performed by 'GPL Cver' regardless of the simulator setting.

### 17.7.2 Timing Resolution

Verilog simulations use 64bit integer values throughout and this includes time. To convert to real time, the value of each time 'tick' needs to be defined. This is the timing resolution defined here.

The default value is 1fs and there is no benefit in changing this unless the simulation runs for longer than  $2^{64} \times 1\text{fs}$ . This is approximately 18000 seconds.

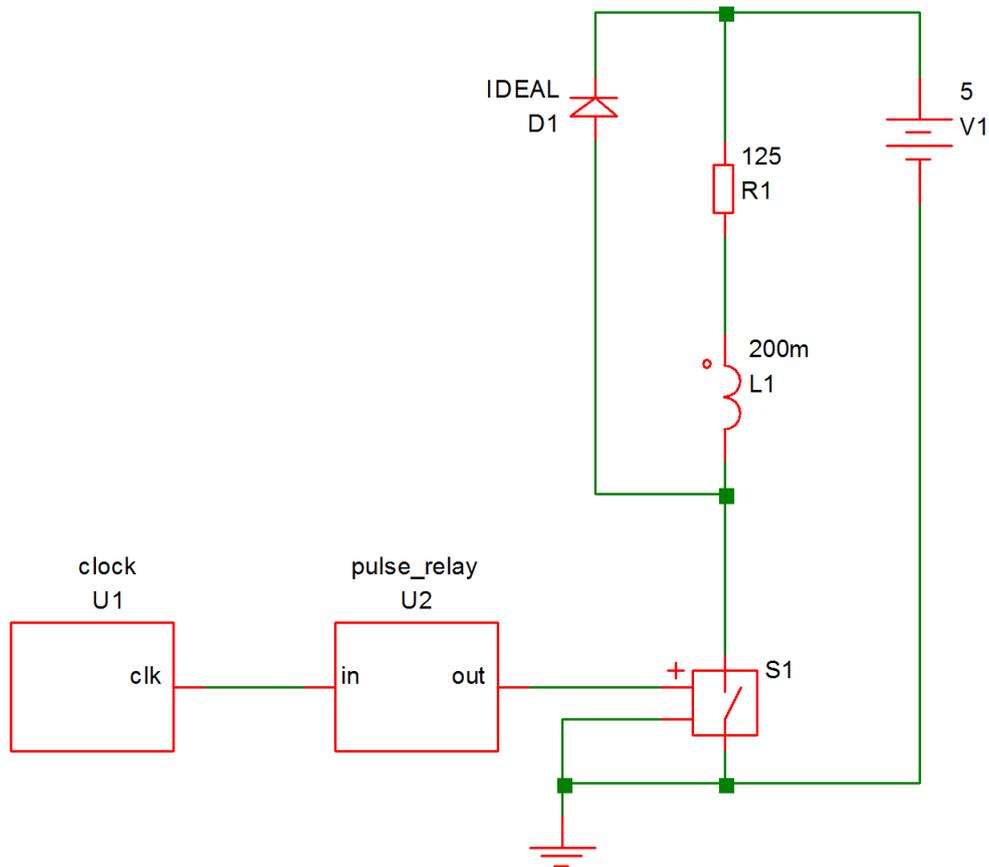
Note that the timescale setting used to define the values of delays etc. within each module, is not affected by this setting.

### 17.7.3 Open Console for Verilog Process

If you set this check box, a console window will open for the run and any messages generated by the Verilog simulation will be displayed in that window.

## 17.8 Tutorial

To demonstrate the basic features of Verilog simulation, we will work through the trivial example shown below:



This circuit pulses a relay for 10mS every 100mS driven by a 100kHz clock. The relay coil is modelled by L1 and R1 while D1 is a freewheel diode. S1 is the relay driver and is controlled by the output of U2. This is a simple counter implemented using the following Verilog code:

```

module pulse_relay(in, out) ;
  input in ;
  output out ;

  integer count ;
  reg out ;

  parameter divide_ratio=10000 ;
  parameter real duty = 0.1 ;

  always @(posedge in)
  begin
    count = count + 1 ;

    if (count==divide_ratio)

```

```

        count = 0 ;

        if (count>divide_ratio*(1-duty))
        begin
            out = 1 ;
        end else begin
            out = 0 ;
        end
    end

    initial
        count = 0 ;
endmodule

```

The Verilog files as well as a completed working schematic can be found in Examples/Verilog-HDL/Tutorial.

### 17.8.1 Procedure

The following assumes that you are already familiar with the basics of entering a schematic and running a simulation.

#### Enter Schematic

1. Open a new empty schematic sheet.
2. Immediately save the empty sheet to:  
Examples/Verilog-HDL/Tutorial/relay-driver.sxsch  
In general it is strongly advised to save the schematic sheet before using the automatic Verilog symbol generation scheme that we are about to demonstrate. This is to ensure that the file system paths of the schematics and Verilog files are kept correctly synchronised.
3. Select menu **Verilog | Construct Verilog-HDL Symbol**. You should see the file clock.v listed. If so select it then click **Open**. If you don't see the file, make sure you saved the schematic to the correct location in step 2 above.  
If you find that this menu is not present, then this means that the Verilog simulation facility is not available with your version of SIMetrix. You will probably need to upgrade your license, contact sales or support for assistance.
4. You should see an image of a symbol appear. Place on schematic in the usual way.
5. Repeat step 3 for pulse\_relay.v
6. Double click the pulse\_relay device (probably U2). You will see a dialog box showing a number of parameters. We aren't going to change any settings, this is just to point out this feature. You should see two parameters at the bottom of the top section called 'divide\_ratio' and 'duty'. These are obtained from the Verilog file pulse\_relay.v.
7. Connect the rest of the circuit as shown in the diagram above. S1 is a regular switch from menu **Place | Analog Functions | Switch** . Make sure you don't forget the ground symbol.

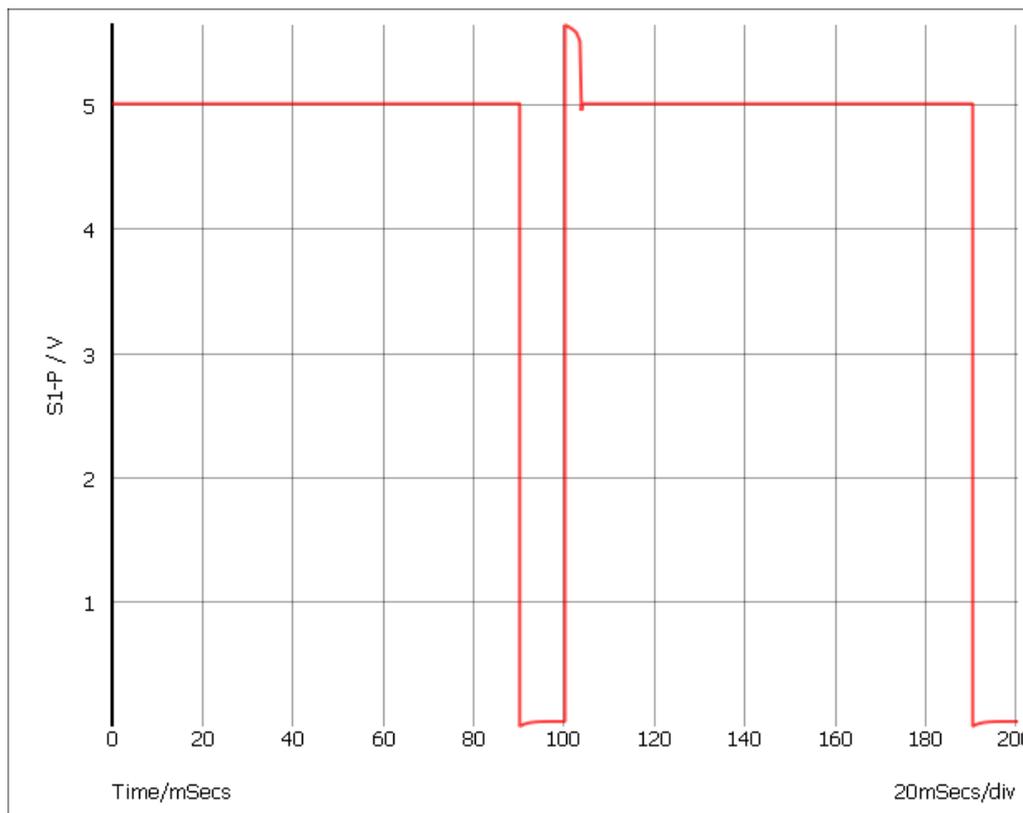
#### Set up Simulation

1. Set up a 200mS transient analysis in the usual way

2. Select Gear integration using menu **Simulator | Choose Analysis** then click on **Advanced Options...** and select **Gear Integration** under the Integration Method group. We do this to tidy up the response of the circuit, this is by no means essential.

## Run Simulation

1. Run the simulation in the usual way. It should take about 1-2 seconds maybe a little longer on an older machine.
2. Plot the voltage on the relay coil. You should see something like this:



You will notice that at  $t=0$ , the voltage is between 4 and 5 volts suggesting that the switch is not fully turned on or off. This is because the output of U2 starts in the unknown state. The unknown state is translated to a high-impedance which leaves the output in a near floating state. To calculate the DC operating point, SIMetrix takes the port values after the first Verilog event which is the state after executing the init block. In the Verilog design the output is the port but you will notice in pulse\_relay.v that is not defined in the init block.

3. Modify the pulse\_relay.v to add an initial definition for the port as follows:

```
initial
begin
    count = 0 ;
    out = 0 ;
end
```

4. Rerun the simulation and notice the change in the result at the start of the simulation.

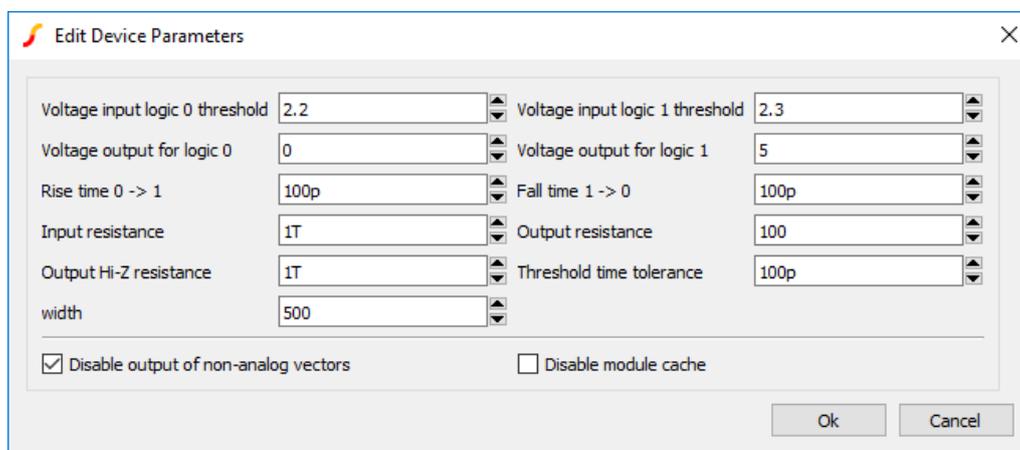
## Internal Verilog Nodes

Have a look at the connection between U1-clk and U2-in. This connects two Verilog signals but does not connect to any analog part. Because of this, it is implemented within the Verilog simulator and does not interact with the analog simulator.

Although the node is not connected to the analog simulator, its data is sent to SIMetrix so that it can be plotted. Try plotting this node now; you will notice that you get a digital plot with no analog detail.

Although, SIMetrix does retrieve the data for internal verilog nodes that interconnect VSXA instances, in circuits where the Verilog digital signals are much higher speed than the analog signals - such as this example - there is a speed penalty for doing so. For this reason there is a facility to disable this. To demonstrate, proceed as follows:

1. Note the time that the last run took using menu **Simulator | Show Statistics**.
2. Double click U1 (the instance of clock.v). You should see a dialog box like this:



3. Check the **Disable output of non-analog vectors** box then click **Ok**.
4. Rerun simulation and note the new simulation time. You will probably see in the region of a 2-3 times speed up. You may conclude from this that the facility to retrieve pure digital data is too expensive to be worthwhile, but this will only be the case where the digital signal are considerable higher speed than the analog signals. In this circuit the analog pulses are running at 10Hz whereas the digital pulses are running at 100kHz - 10000 times as fast.
5. As a further exercise, you may like to see what happens when this node is connected to an analog part. Try connecting a 1pF capacitor to ground and run the simulation. You will find that the simulation runs maybe 100 times slower. This is because analog time steps is now being performed for this high frequency signal. With just the 10Hz output to deal with, the analog simulator needed to perform only around 200 timepoints. Now it has to work at 100KHz it needs 1 million or so.

## Multi-step Run

As a final exercise, we will show how it is possible to perform multi-step runs while varying a parameter sent to a Verilog device. We will run a 3 step simulation while varying the DUTY parameter of the pulse\_relay device. Proceed as follows:

1. Double click U2
2. Set the 'duty' parameter to:

```
{duty}
```

3. Open the choose analysis dialog box.
4. In the Transient sheet, select Enable multi-step then click Define.
5. In Sweep mode select Parameter. Set Start value and Stop value to 100m and 500m respectively. Set Number of steps to 3.
6. In Parameters set Parameter name to duty.
7. Ok dialog boxes. If you carried out step 5 in the previous section [“Internal Verilog Nodes” on page 436](#), remember to remove the capacitor and restore the connection between U1 and U2 to an internal Verilog node.
8. Run simulation then plot relay drive as before. You should see three curves.

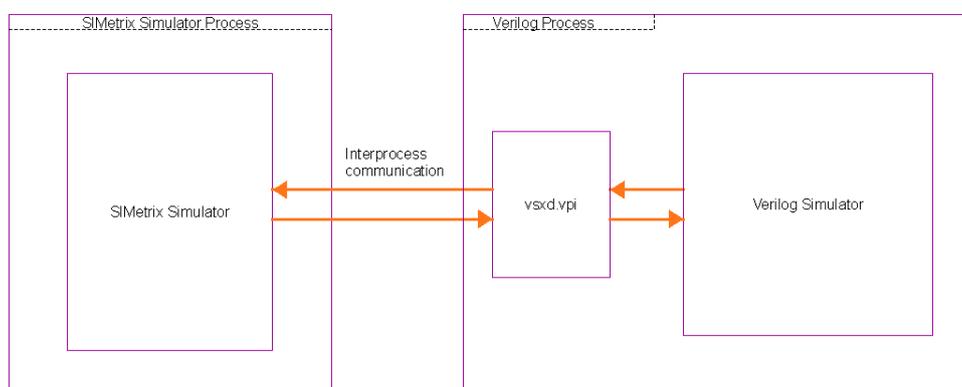
## 17.9 Verilog Simulator Interface

In this section we describe some details of how the Verilog interface works.

### 17.9.1 VPI

SIMetrix communicates with the Verilog simulator using the *Verilog Procedural Interface* or VPI. VPI allows an external program to communicate with a Verilog simulator.

The diagram below shows the program structure.



vsxd.vpi is a DLL/shared library that is provided as part of SIMetrix. This is loaded by the external Verilog simulator and runs in the Verilog process’s memory space. vsxd.vpi uses the VPI interface to send events to the Verilog simulator, to respond to events generated by the Verilog simulator and to interrogate the Verilog simulator about details of the user’s Verilog module.

vsxd.vpi is able to respond to events during the Verilog simulation that cause a change on an output port and send those changes to the SIMetrix simulator.

Conversely the SIMetrix simulator detects when there are changes on an input port and notifies vsxd.vpi which then notifies the Verilog simulator.

vsxd.vpi is also able to enumerate the ports and parameters of a Verilog module and report this information back to SIMetrix. This process is performed before a simulation starts in order to build the final Verilog top level design.

## 17.9.2 Interface Configuration

In order for the interface to be setup, SIMetrix needs to know how to drive the Verilog simulator. Also the Verilog simulator needs to know about vsxd.vpi.

The interface is configured by the VerilogHDL.ini file which is located in the support folder. VerilogHDL.ini uses the standard inifile format. Each supported simulator is defined in a section within VerilogHDL.ini and the section must at a minimum define the following keys 'Name', 'Script' and 'Path'. 'Name' defines a display name for the simulator that will be meaningful to the user. 'Script' defines a script name that is used to launch the Verilog simulator. (More on the launch script below). 'Path' is passed to the launch script and would usually be the path where the main binary executable for the Verilog simulator is located.

## 17.9.3 Launch Script

The launch script is a SIMetrix script that is called by SIMetrix. It is responsible for starting the Verilog process usually via the 'Shell' function. The launch script knows about the command line syntax for the Verilog simulator.

For more details about the launch script, see the script used to launch the GPL cver simulator.

## 17.9.4 Verilog Simulation Preparation

SIMetrix netlists can instantiate any number of verilog designs, both multiple instances of the same Verilog module and multiple module designs. In order for the Verilog simulator to be able to handle these multiple instances, it needs to be presented with a top level module that defines the interconnections between them. This top level module is generated automatically by SIMetrix on each simulation run and is called (by default) vsx\_root.v.

# Chapter 18

## Sundry Topics

### 18.1 Saving and Restoring Sessions

#### 18.1.1 Overview

You can save the current session for later restoration. This is useful in the situation where you are in the middle of editing schematics or studying simulation results, but you need to interrupt this work maybe at the end of a working day. While in some situations you might simply be able to leave your computer switched on and logged in, or maybe use a “Hibernate” mode, these methods are not always practical or indeed reliable.

The SIMetrix save session feature will save the current state of all open schematics, all open graphs and any simulation data so that it can be restored at a later time.

#### 18.1.2 Saving a Session

Select menu **File | Save Session**.

#### 18.1.3 Restoring a Session

You can only restore a session if all graphs and schematics are closed and there is no current simulation data loaded. This is the normal state when SIMetrix has just been started. If you wish to restore a session when SIMetrix has been in use since first starting, you can either shut down and restart, or close all windows and graphs then delete all data groups. To delete all data groups, select menu **Simulator | Manage Data Groups...**, then click on Delete All Groups and Ok.

To restore the session, select menu **File | Restore Session**

#### 18.1.4 Where is Session Data Stored?

Session data is stored in the following directory:

*application\_data/session*

where *application\_data* is the SIMetrix application data directory. See [“Application Data Directory” on page 444](#) for details.

## 18.2 Symbolic Path Names

### 18.2.1 Overview

Some file system path names used by SIMetrix may be defined using a symbolic constant. Such paths are of the form:

*%symbol%path*

Where *symbol* is the name of the constant and *path* is any sequence of characters valid for a path name. The actual path is resolved by substituting *%symbol%* with the value of *symbol*.

Symbolic paths make it easy to move files to new locations as only the values for their symbols need to be changed in order for SIMetrix to be able to continue to find them.

### 18.2.2 Definition

There are two types of symbolic constant. These are *system constants* and *user constants*. System constants are pre-defined while user constants can be arbitrarily defined by the user. There are currently 7 system constants. These are:

STARTPATH	Full path of the current working directory from where SIMetrix was launched.
DOCSPATH	Full path of the <i>My Documents</i> folder.
EXEPATH	Full path of the location of the SIMetrix binary SIMetrix.exe.
APPDATAPATH	Full path of the Application Data directory.
TEMPPATH	Path of temporary directory
SXAPPDATAPATH	Path of the SIMetrix application data directory. See <a href="#">“Application Data Directory”</a> on page 444 for details.
SHAREPATH	Path of the root support directory used for various support files used by SIMetrix such as model and symbol libraries.
LIBPATH	Path of plugins directory in SIMetrix installation
SXDOCSPATH	Path of SIMetrix directory in <i>My Documents</i>
COMMON_APPDATAPATH	Path of common application data directory. (Typically C:\ProgramData)

User constants must be defined in the configuration file. See [“Configuration Settings”](#) on page 443 for more information. User constants are defined in the [Locations] section of the file. Currently these must be added by hand using a text editor.

The format used is as follows:

```
[Locations]
symbol_definitions
```

Where *symbol\_definitions* is any number lines of the form:

```
symbolname=symbolvalue
```

*symbolvalue* may be any sequence of characters that are valid for a system path name and may contain spaces. There is no need to enclose it in quotation marks even if the value contains spaces. Nested definitions to any level are permitted. That is *symbolvalue* may also itself use other symbolic constants. Recursive definitions won't raise an error but will not be meaningful.

UNC paths (e.g. \\server\c\project) may be used for *symbolvalue*.

Comments may be added to the project file prefixed with a semi-colon.

### 18.2.3 Configuration File Example

The following shows examples of symbolic path name definitions in the configuration file. Lines such as these may be placed anywhere in the file, but we recommend that they are placed at the end.

```
; Project file
[Locations]
Project=c:\Projects\proj1
Cells=%PROJECT%\Cells
```

### 18.2.4 Using Symbolic Names

Symbolic path constants may be used in the applications listed below. In all cases a mechanism called *automatic path matching* is used which means that to use symbolic paths, all you need to do is define the values in the project file then carry on working as before. The automatic path matching algorithm attempts to match a user symbol or one of the EXEPATH or DOCSPATH system symbols to a part of the path being processed. If a match is found, the path name will be stored with the symbolic value.

#### Component paths

If a component is placed using the full path option, the automatic path matching mechanism described above will be invoked. For example suppose the user symbol CELLS has the value C:\Projects\Proj1\Cells and the component with path C:\Projects\Proj1\Cells\celllib1\inv.sxsch is placed *using the full path method*. The actual value of the SCHEMATIC\_PATH property will become %CELLS%\celllib1\inv.sxsch. The matching of C:\Projects\Proj1\Cells to %CELLS% is performed automatically.

Note that automatic path matching will not be invoked for components placed using the relative path method.

#### Global model library file paths

Model files installed globally can use symbolic paths. The automatic path matching mechanism described above will be invoked when models are installed. So if the model file C:\SPICELIB\OnSemi\\*.mod and the symbol MODELLIB has the value C:\SPICELIB, the model file path will be saved as %MODELLIB%\OnSemi\\*.mod.

#### Path option variables

- StartupDir
- ScriptDir
- BiScriptDir
- TempDataDir
- DefaultLib
- SymbolsDir

Automatic path matching is invoked whenever these values are set or modified.

## Symbol file locations

Schematic symbol file paths may be stored using symbolic constants. Automatic path matching is invoked whenever a library is installed.

## Notes for Windows

The automatic path matching system will correctly match a drive based path (e.g. h:\projects\proj1) with its mapped UNC path (e.g. \\server1\c\projects\proj1) provided the drive based path points to a network share and not a local drive. For example if the project file contains the entry:

```
Project=\\server1\c\projects\proj1
```

and \\server1\c is mapped to the H: drive then the file H:\Projects\proj1\cell23.sxcmp will be stored as %Project%\cell23.sxcmp. However, if you are actually running SIMetrix from the machine server1 and \\server1\c is the share name for the local C: drive then C:\Projects\proj1\cell23.sxcmp will not be recognised as equivalent to %Project%\cell23.sxcmp. This limitation is due to security restrictions in Windows.

## 18.3 SIMetrix Command Line Parameters

A number of command line parameters may be supplied to the SIMetrix binary (SIMetrix.exe) when starting the program. The full syntax is as follows:

```
SIMetrix(.exe) [schematic_file] [/s startup_script] [/i] [/n]
  [/c config_location] [/r product_code]
```

<i>schematic_file</i>	Path of a schematic file usually with extension .xsch. This file will be opened immediately.
<i>/s startup_script</i>	Name of script file or command that will be executed immediately after SIMetrix starts.
<i>/i</i>	If specified, the <i>schematic_file</i> or/and <i>startup_script</i> will be opened/run in an existing instance of SIMetrix if there is one. That is, a new instance will not be started unless none are already running.
<i>/n</i>	Now a legacy option. Originally inhibited the display of the splash screen. The splash screen was removed from version 5.1 so this option now does nothing.
<i>/c config_location</i>	This identifies where SIMetrix stores its configuration settings. <i>config_location</i> should be of the form:

```
PATH;file_pathname
```

*file\_pathname* identifies the location of a file to store the configuration settings. You may use any of the system constants defined in [“Definition” on page 440](#) in this definition of *file\_pathname*. E.g. %EXEPATH% for the executable directory. See [“Configuration Settings” on page 443](#) for details of configuration settings. The ‘REG;’ syntax available with earlier versions is no longer supported.

*/r product\_code*

Code that defines which product to request from the license server. By default, SIMetrix attempts to checkout the highest spec product available from the license server. If this option is specified the product listed in the following table will be checked out if it is available. This can be used to leave features that you are not using available to other users.

DP	SIMetrix Classic
DX	SIMetrix Pro
UX	SIMetrix Elite
SP	SIMetrix/SIMPLIS
SP1	SIMetrix/SIMPLIS Pro
SP2	SIMetrix/SIMPLIS Elite
DP-DVM	SIMetrix Classic with DVM
DX-DVM	SIMetrix Pro with DVM
UX-DVM	SIMetrix Elite with DVM
SP-DVM	SIMetrix/SIMPLIS with DVM
SP1-DVM	SIMetrix/SIMPLIS Pro with DVM
SP2-DVM	SIMetrix/SIMPLIS Elite with DVM
VH1	SIMetrix/SIMPLIS VH Pro
VH2	SIMetrix/SIMPLIS VH Elite
VH1-DVM	SIMetrix/SIMPLIS VH Pro with DVM
VH2-DVM	SIMetrix/SIMPLIS VH Elite with DVM

### 18.3.1 Using startup.ini

Start-up parameters can also be specified in a file called startup.ini. This must be located in the same directory as SIMetrix.exe. The format of the file is as follows:

```
[StartUp]
settings
```

*settings* can be any combination of the following:

```
StartupScript=startup_script (equivalent to /s on command line)
UsePrevInst= (equivalent to /i on command line)
InhibitSplash= (equivalent to /n on command line)
ConfigLoc=config_location (equivalent to /c on command line)
Features=features (equivalent to /f on command line)
```

## 18.4 Configuration Settings

### 18.4.1 Overview

SIMetrix, in common with most applications, needs to store a number of values that affect the operation of the program. These are known as configuration settings. Included among these are the locations of

installed symbol libraries, installed model libraries, font preferences, colour preferences and default window positions.

## 18.4.2 Default Configuration Location

By default, SIMetrix stores configuration settings in a single file. This file is located at:

`simetrix_app_data_dir\config\Base.sxprj`

See “Application Data Directory” on page 444 for location of `simetrix_app_data_dir`.

## 18.4.3 Application Data Directory

SIMetrix stores a number of files in its *application data directory*. The directory is at one of these locations:

`sys_application_data_dir\SIMetrix Technologies\SIMetrixxx` (full production versions)

`sys_application_data_dir\STI_STL\Elementsxxx` (SIMetrix/SIMPLIS Elements)

where

`xx` is a three digit code representing the SIMetrix version number, eg. 800 for version 8.00.

`sys_application_data_dir` is a system defined location.

The following table shows typical locations for all supported Windows systems:

Operating System	Path
Windows 7, Windows 10	C:\Users\username\AppData\Roaming

`username` is the log on name currently being used. The above are only typical locations on English language versions of Windows. The user or system administrator may move them and also the names used may be different for non-English versions of Windows.

With full versions of SIMetrix, you can locate the SIMetrix application data directory by typing the following command at the command line:

```
Show TranslateLogicalPath(`\%sxappdatapath\%')
```

## 18.4.4 Specifying Other Locations for Config Settings

You can specify alternative locations for the configuration settings. This can be done with the `/c` switch on the command line or `ConfigLoc` setting in the `startup.ini` file. See “SIMetrix Command Line Parameters” on page 442 for more details.

## 18.5 Options

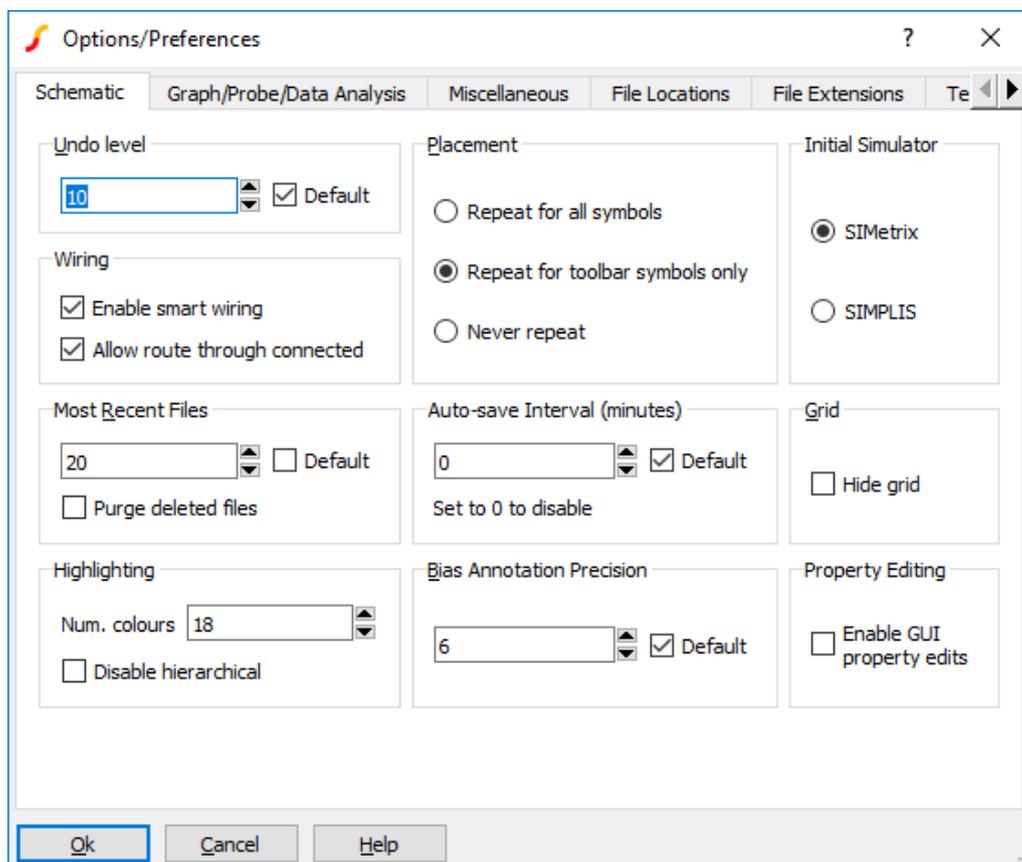
## 18.5.1 Overview

There are a number of options affecting all aspects of SIMetrix. Many of these may be viewed and adjusted using the Options dialog box, others can only be accessed from the command line using the Set and Unset commands.

## 18.5.2 Using the Options Dialog

This is invoked with the menu **File | Options | General....** The following sections explain the options available.

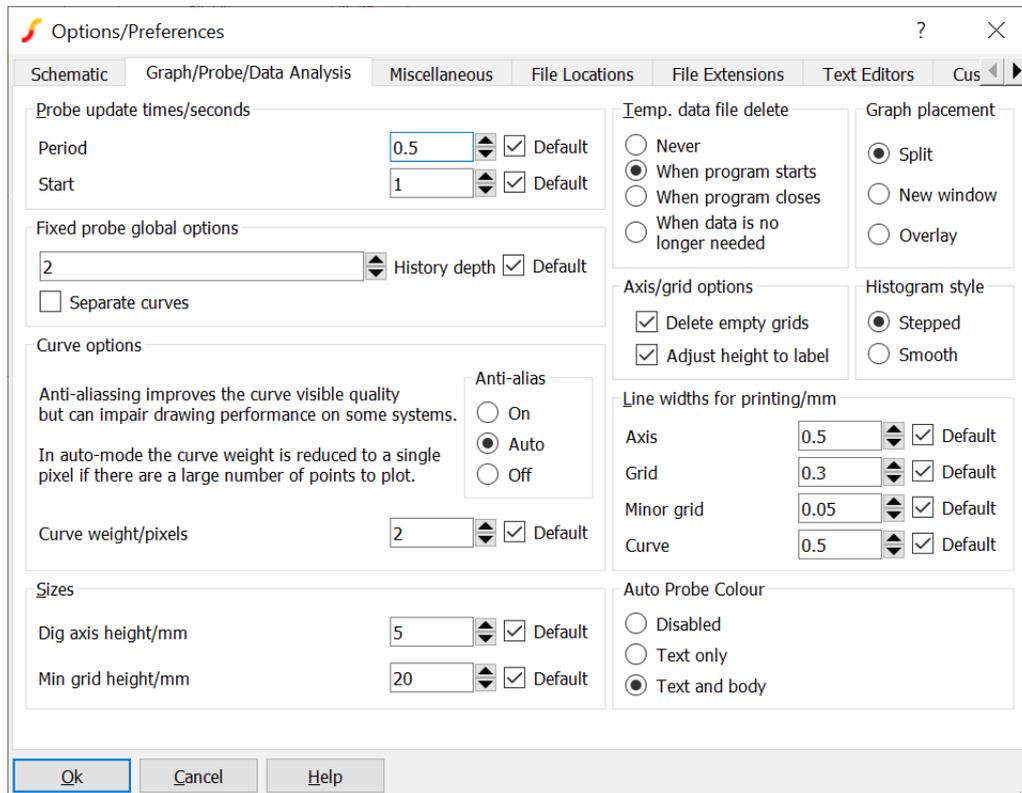
### Schematic Sheet



Undo Level	Number of levels of schematic undo. See <a href="#">“Creating a Schematic” on page 32.</a>
Most Recent Files	Controls how many recently used files are displayed in the File   Reopen menu
Bias Annotation Precision	Controls the precision of the schematic voltage and current bias annotation markers.

Auto-save interval (minutes)	Enter a value to enable schematic auto-save. SIMetrix will automatically save a backup copy of your schematic at the interval specified. If SIMetrix aborts abnormally, perhaps due to a power failure, you will be prompted to recover the auto-saved schematic when you next start SIMetrix.
Hierarchy Highlighting	Check the Disabled box to disable highlighting through the hierarchy. With deep and complex hierarchies, net highlighting can be time-consuming. Checking this box will restrict highlighting to the current schematic only.
Property Editing	Enable GUI Property Edits: If checked, allows property text (i.e. labels) to be moved using mouse actions. Labels themselves can be edited by double clicking. See <a href="#">“Notes on Property Text Position” on page 60</a> before checking this option.
Placement	When to auto-repeat placement of schematic parts. If autorepeat is enabled, a new symbol to be placed is automatically displayed after each placement. This speeds the placement of many instances of the same device.
Wiring	<p>Enable smart wiring enables the smart wiring algorithm. See <a href="#">“Wiring” on page 60</a> for details about smart wiring.</p> <p>Allow route through connected is an option for the smart wiring algorithm that allows it select routes that pass through existing wires that are already connected to the source or target destinations.</p>
Grid	Option to hide the schematic’s grid.
Initial Simulator	This is only relevant for SIMetrix/SIMPLIS products. Sets the initial simulation mode when opening a new schematic. If most or all of your work is with SIMPLIS, check SIMPLIS. This will save time switching to SIMPLIS mode for all new schematics.

## Graph/Probe/Data Analysis



### Probe update times

Plots created from fixed probes are updated on a regular basis. This controls how frequently and when it starts.

**Period:** Update period in seconds.

**Start:** Start delay in seconds.

### Fixed probe global options

**Default history depth:** Sets the default history depth for fixed probes. This is number of curves from previous simulations that remain after a simulation. If set to 1, previous results are deleted. If set to zero all results are retained. History depth can also be set individually for each probe. See [“Probe Options Sheet” on page 291](#) for details.

## Curve options

Anti-alias	Anti-aliasing is a technique of improving the visual quality of the curve. For curves with weight of 2 pixels or more, the processing time required can be significant and for this reason anti-aliasing may be switched off. In AUTO mode anti-aliasing is enabled but the curve weight will reduce to one pixel for complex curves with many points.
Curve weight	Thickness of displayed curves. Curves display much quicker if this value is set to 1 but are clearer (but can lose detail) if set to 2.

## Sizes

Digital Axis Height	Sets height of axes (in mm) used to plot digital traces.
Min grid height	When a grid is added to a graph window, existing grids are reduced in height to accommodate the new one. But they won't be reduced to a height lower than specified by this setting. When this limit is reached, the vertical space will be increased by allowing the window to scroll.

Temporary data file delete	<p>Simulation data is stored in temporary data files. The data is not read into system memory until it is needed - say - to plot a graph. However, the location in the file of the various vectors is always in memory so that the data can be extracted from the file as rapidly as possible. It is this latter location data that is destroyed when a simulation run gets out of date. The file containing the data gets deleted at a time set by the above options, not necessarily when the data is no longer needed. As long as the file exists, the data can be recovered by calling <b>File   Data   Load...</b> or <b>File   Data   Load Temporary Data...</b> which re builds the location data. The options described below control when these data files are deleted.</p> <p><b>Never</b> Temporary files are never deleted but will be overwritten in subsequent sessions. Not recommended unless you only ever do short simulations.</p> <p><b>When SIMetrix starts</b> All temporary files are deleted when SIMetrix starts.</p> <p><b>When SIMetrix closes</b> All temporary files are deleted when SIMetrix is shut down. While using SIMetrix, you can recover earlier simulation runs. Normally, only the 3 most recent are kept but earlier ones can be recovered from the TEMPDATA directory using <b>File   Data   Load...</b></p> <p><b>When data is no longer needed</b> This is the most aggressive delete method and is recommended if you do many long runs or/and have limited disc space. By default, the 3 most recent runs are kept but with the other options above, the data files are not deleted when the data is not needed but links to the data in them are released. (See explanation below). If this option is set, the data files are deleted as soon as they become out of date, optimising use of disc space at the expense of not being able to recover old data.</p>
Graph placement	<p>Specifies how new graph sheets are created.</p> <p><b>Split</b> Splits current view to create a new graph view on the right hand side</p> <p><b>New window</b> Creates a new graph main window if one does not already exist</p> <p><b>Overlay</b> Overlays the graph sheet over existing views</p>
Axis/grid options	<p><b>Delete empty grids:</b> If checked, grids will be deleted when the last curve is deleted from the grid</p>
Histogram style	<p>Controls the curve style used for histogram displays:</p> <p><b>Stepped</b> Displays a flat line for the width of each bin. Similar to a bar graph.</p> <p><b>Smooth</b> Joins the centre of each bin with a straight line to display a continuous curve.</p>

## Line widths for printing/mm

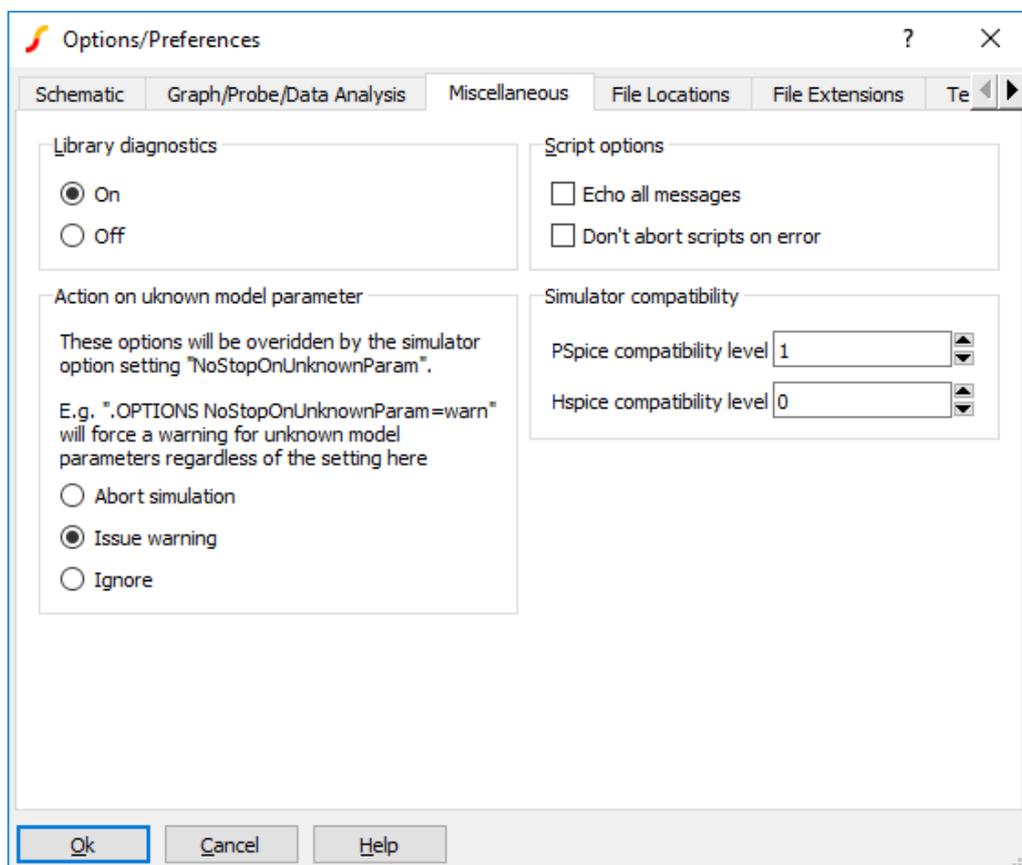
Axis line width	Width in mm of printed axis.
Grid line width	Width in mm of printed grid lines.
Minor grid	Width in mm of printed minor grid lines.
Curve line width	Width in mm of printed curves.

## Auto probe colour

Options for auto probe colouring. See [“Auto probe colouring” on page 322](#)

Disabled	Auto probe colouring is switched off
Text only	Only the text attached to probes is coloured
Text and body	The probe’s text and symbol are coloured

## Miscellaneous



## Library Diagnostics

Determines whether messages are displayed when models are found in the library.

Action on unknown model parameter	Specifies action to be taken when an unknown model parameter is encountered. There are three options:
Abort simulation	an error will be raised and the simulation will be aborted.
Issue warning	a warning message will be output to the list file but the simulation will continue normally.
Ignore	no warning or error will be raised and the simulation will continue normally.
Script Options	
Echo all messages	If set, all script lines will be displayed in the message window. This will result in a great deal of output and will slow down the whole program operation. Only set this if you are debugging your own scripts.
Don't abort scripts on error	Normally scripts abort if an error is detected. Check this box to disable this behaviour.
Simulator compatibility	
PSpice compatibility level	Can have values of 0, 1 or 2. The default is 1. A value of 2 gives the highest level of compatibility with PSpice but may introduce changes of behaviour that are not compatible with older versions of SIMetrix or other simulators. A value of 0 retains compatibility with SIMetrix versions 8.1 and earlier. Refer to <i>Simulator Reference Manual/Command Reference/.OPTIONS</i> for full details.
Hspice compatibility level	Can have values of 0 or 1. If you are using PDKs (process design kit) designed for Hspice, set this option to 1. Leave this option at its default setting of zero if you are not using PDKs for IC design. Refer to <i>Simulator Reference Manual/Command Reference/.OPTIONS</i> for full details.

## File Locations

Locations in your files system of various files and folders needed for correct operation of SIMetrix.

Built-in Scripts	This is the first location that SIMetrix searches for scripts. Much of the user interface is implemented with scripts and these are all internal to the program. These can be overridden by placing scripts of the same name at this location. This allows modification of the UI. Changing this setting has no effect until you restart SIMetrix.
------------------	--

Scripts	Location of script directory. This directory is searched for any scripts you run. Only change this setting if you are actually moving the script directory. Changing this setting has no effect until you restart SIMetrix.
Start up	Current working directory on start up.
Start up Script	Name of script that is automatically run on startup. You can place custom menu or key definitions in this file.
System Symbols Location	Directory location where the standard symbols are located.
Temp Data	Location of temporary simulation data files. Changing this setting has no effect until you restart SIMetrix. Note that this should always be a local directory. That is, it must not be on a remote network partition.
User Symbols Location	Directory where user symbol libraries are expected to be located. Note, you can place symbol libraries anywhere. This directory is simply a location that some UI functions use as a default.
Netlist Editor	Text editor used to edit netlists, models and simulation list files. The default is the built-in text editor.
Script Editor	Text editor used to edit scripts. The default is the built-in text editor.
Verilog-A Editor	Text editor used to edit Verilog-A files. The default is the built-in text editor.
Verilog-HDL Editor	Text editor used to edit Verilog-HDL files. The default is the built-in text editor.
Sensitivity Report Style Sheet	.CSS style sheet file that defines the style for sensitivity analysis reports
Worst-case Report Style Sheet	.CSS style sheet file that defines the style for worst-case analysis reports
Monte Carlo Log Style Sheet	.CSS style sheet file that defines the style for Monte Carlo logs

## File Extensions

Defines extensions used for the various files used by SIMetrix. These setting control what file extensions are listed for editors of the various file types and, where relevant, what action will be taken if a file of that type is dragged and dropped into the SIMetrix environment.

Many SIMetrix file types use five letter extensions beginning with .sx. This is to help prevent clashes with other applications.

For each setting, the supported extensions are separated by a semi-colon. The first in the list is the default. So, for example, the default schematic extension is sxsch so when you save a schematic without giving an extension, it will automatically be given the extension sxsch.

Schematic Files	Extensions used for schematic files.
Data Files	Extensions used for simulation data files.
Text files	Supported extensions for text files.
Symbol Files	Extension used for binary schematic symbol files. This is used by the drag and drop system to detect Symbol Library files being dropped into the command shell.

Logic Def. Files	Files used for logic definitions for the digital simulator's arbitrary logic block. If the extension is omitted in the model (FILE parameter) this will be used.
Scripts	Default extension for scripts if called without an extension.
Device models	SPICE model files. Used by drag and drop system to detect model file types. Note the drag and drop system will detect model files with extensions not in this list and it is not usually necessary to specify model file types here.
Netlist Files	Files used for simulation netlists.
Schematic Components	Extensions used by schematic component files.
Verilog-A Files	Verilog-A Source files.
Verilog-HDL Files	Verilog-HDL Source files.

## Text Editors

Defines colour syntax highlighting for the various text editors.

## Customisation

Defines colours for various UI elements. Note that schematic elements are defined using schematic styles.

Elements are divided into these groups:

Command Shell	Command shell text and background colours. Command shell input is the command line at the top, while command shell output is the message window below it
File View	Colour elements of the file viewer
Part Selector	Colour elements of the part selector
Graph	Colour elements for graphs. This includes both text and graphic objects. Note that all text items are based on the Graph Text Colour object. Changing this object will also change the colour of the objects: Caption Colour, Free Text Colour, Text Box Text Colour, Legend Box Text Colour, Curve Marker Text Colour and Cursor Dimension Text
Curves	Curve colours are chosen using a sequence number. The first curve on a sheet uses sequence 1, the second sequence 2 etc. New curves use the first unused sequence number. The colours associated with each sequence number can be defined here.

### 18.5.3 Using the Set and Unset commands

All options have a name and many also have a value. These are set using the [Set](#) command and can be cleared with [Unset](#). When an option is cleared it is restored to its default value. A complete listing of available options with possible values is given below. Note that option settings are persistent. This means that their values are stored either in the .INI file or in the system registry (see [“Configuration Settings” on page 443](#)) and automatically restored at the start of each subsequent SIMetrix session.

## List of Options

Upper and lower case letters have been used for the option names in the following listing only for clarity. Option names and their values are not in fact case sensitive. Many of the options described below are supported by the Options dialog box in which case they are noted accordingly.

## Unsupported Options

Some options in the following list are marked as ‘unsupported’. This means that they may be withdrawn in the future or their functionality changed.

See also

- [“Toolbar Buttons” on page 470](#)
- [“File Extensions” on page 469](#)
- [“Text Editor Colour Options” on page 466](#)

Name	Type	Description	User interface support
700Extensions	Boolean	Schematic symbols for the 700 series semi-custom arrays are enabled if this is set.	
ActionOnMissingModel	string	Default=AutoAssociate Action to be performed when attempting to place a model from the library, but that model has not been associated. If set to AutoAssociate, will attempt to auto-associate by simulating the model	
AllowTempDataDrives	String	List of drives separated by commas which are allowed to be used for simulation data without triggering a warning message. Drive should be specified with ‘:’ suffix - e.g Y:. A warning message will be issued for any drive that is reported as connected through a network if used for simulation data storage. Only use this setting to override drives that are reported as network drives but in fact are connected locally.	
AlphaBlendRatio	Numeric	Default=75 Value of 0 to 100 controls the opacity of plotted curves. 0 is totally transparent (and therefore invisible). 100 is fully opaque	No
AnnoMinSuffix	Numeric	Default = 1 Minimum suffix used for automatic generation of schematic part references	

Name	Type	Description	User interface support
AutoStartWire	Boolean	Default=False Only effective if AutoWireEnabled is False. Selects mode whereby a wire is started when the cursor is brought close to a pin or wire termination. This mode is on automatically when AutoWireEnabled is True	Options dialog
AutoWireEnabled	Boolean	Default=True Smart wiring is on if this is True	Options dialog
AAllowRouteThruConnected	Boolean	Default=True Controls whether the smart wire algorithm is allowed to route wires through existing wires that connect to the destination or target	Options dialog
AxisPrintWidth	Numeric	Default = 0.5 Width of printed axis in mm. See also CurvePrintWidth and GridPrintWidth	Options dialog
AutoSaveInterval	Numeric	Default=0 meaning disabled. Interval in minutes for which the session is automatically saved in order to recover from an unexpected program exit	Options dialog
BiasAnnoPrecision	Numeric	Default=6 Precision of values displayed for schematic DC bias annotation	Options dialog
BiScriptDir	Text	This is the first location that SIMetrix searches for scripts. See File Locations section of Options dialog above for more info.	Options dialog
BreakWireOperations	Text	Default=AllOps. See <a href="#">“Breaking Wires” on page 59</a> . If set to PlaceOnly, only placing a new part will lead to a wire being broken; other editing operations such as move and copy will leave the wire untouched	
BreakWireParts	Text	Default=AllParts. See <a href="#">“Breaking Wires” on page 59</a> . If set to TwoTermOnly, only two terminal parts will break wires.	
BuildAssociations	Text	Default=ask See <a href="#">“Auto Configuration Options” on page 470</a>	

Name	Type	Description	User interface support
BuildModelLibs	Text	Default=ifempty See “Auto Configuration Options” on page 470	
BuildPreferenceSettings	Text	Default=askmigrate See “Auto Configuration Options” on page 470	
BuildSymbolLibs	Text	Default=ifempty See “Auto Configuration Options” on page 470	
CachePathSymbols unsupported	Boolean	Default=false If true, SIMetrix will cache symbolic paths	
ComponentExtension	Text	Default=sxcmp File extension for schematic component files	Options dialog
CursorDisplay	Text	Default = <i>Graph</i> . Controls initial graph cursor readout display. <i>Graph</i> Display on graph only <i>StatusBar</i> Display on status bar only <i>Both</i> Display on both graph and status bar	Options dialog
CurveNoAntiAlias	Boolean	Default = false. If false, curves are plotted using anti-aliasing. This improves the visible quality of the curve. However, anti-alias processing for curves with more than one pixel width is slow. As a consequence curves of medium or high complexity are forced to single pixel if anti-aliasing is enabled. Setting this option to true will degrade plotted quality but will allow curves of 2 or more pixels to be plotted for all levels of complexity	No
CurvePrintWidth	Numeric	Default = 0.5mm Width of printed graph curves in mm. See also GridPrintWidth.	Options dialog
CurveWeight	Numeric	Default = 1 Sets the line width in pixels of graph curves. Note that although widths greater than 1 are clearer they normally take considerably longer to draw. This does however depend on the type of adapter card and display driver you are using.	Options dialog

Name	Type	Description	User interface support
DataExtension	Text	Default=sxdat;dat Default file extension for data files	Options dialog
DataGroupDelete	Text	Default = <i>OnStart</i> Determines when temporary simulation data is deleted. Possible values, <i>Never</i> , <i>OnStart</i> , <i>OnClose</i> and <i>OnDelete</i> . See “ <a href="#">Graph/Probe/Data Analysis</a> ” on page 447 for details.	Options dialog
DataHistoryDepth	Numeric	Default=4. Controls the number of previous results stored in each curve created by a fixed probe	No
DefaultLib	Text	Default=%SHAREPATH%/SymbolLibs/default.sxslb Name and location of default symbol library	
DefaultPersistence		Default=0 Sets the number of curves that are kept for graph fixed probes. ‘0’ means that all curves are kept. ‘1’ means that only 1 is kept at a time.	Options dialog
DevConfigFile	Text	Default=%SHAREPATH%/DeviceConfig.cfg Name and location of device configuration file. See <i>Simulator Reference Manual</i> for details	
DigAxisHeight	Numeric	Default = 8.0mm Height of digital axis in mm. (Screens are typically 75pixels/inch)	Options dialog
DisableCurveDataCompression	bool	Default=FALSE Set to TRUE to disable curve point compression. This generally speeds up graph drawing but can introduce minor artefacts	No
DisplaySimProgressMessage	Boolean	Default=False If true, a message will be displayed in the command shell indicating the start and end of a simulation	
EchoOn	Boolean	When set, all commands are echoed to the command shell message window. This is used primarily for script debugging.	Options dialog
EnableFileChangeWatcher	bool	Default=TRUE The GUI monitors changes to files and prompts the user to reload when changes are detected. Set this to FALSE to disable	No
EnableFileChangeWatcher FileView	bool	Default=TRUE The file viewer monitors changes to files and updates if changes are detected. Set to FALSE to disable this behaviour	No

Name	Type	Description	User interface support
EnableLWEditMenu	bool	Default=FALSE Enables the schematic popup menu <b>Edit MOS Length/Width...</b> which edits the length, width and multiplier (M) for MOSFETs used in IC design	
ExportRawFormat	Text	Default = SPICE3 Possible values, SPICE3, SPECTRE and OTHER. Controls format of raw output. See <a href="#">“Exporting SPICE3 Raw Files” on page 370</a>	No
ForceGlobalHash	Boolean	Default=false  In the simulator, any node name found in a subcircuit that starts with a ‘#’ accesses a top level node of the same name but without the ‘#’. E.g. #VCC in a subcircuit connects to VCC at the top level.  If this option is set, the ‘#’ is not stripped, so #VCC in a subcircuit connects to #VCC at the top level.	
GraphExtension	Text	Default=sxgph  File extension used for graph files	Options dialog
GraphLegendMaximumHeightProportion	real	Default=0.25 Maximum height of the graph legend window as a proportion of the graph window height	
GraphPlacement	string	Default=split.  Selects how new graph windows are created. Has one of three values <i>Split</i> Splits current layout and places graph on the right side <i>NewWindow</i> Creates a new main window for graphs if one does not already exist <i>Overlay</i> Overlays graph windows over the top of existing windows	Options dialog
GridPrintWidth	Numeric	Default = 0.3mm Width of printed graph grid lines in mm. See also CurvePrintWidth	Options dialog
GroupPersistence	Numeric	Default = 3 Sets the number of groups that are kept before being deleted. See <a href="#">“Plotting the Results from a Previous Simulation” on page 338</a>	No

Name	Type	Description	User interface support
GuiEditPropertyEnabled	Boolean	Default=false If set, allows visible properties in the schematic to be edited using GUI actions.	Options dialog
HideSchematicGrid	Boolean	If set, the schematic grid will be suppressed	Options dialog
HistoAdvancedStats	bool	Default=FALSE Display advanced statistics in histogram plots	Performance analysis dialog
HighlightIncrement	Numeric	Default = 1 Highlighted graph curves are thicker than normal curves by the amount specified by this option	No
HistoCurveStyle	Text	Default=stepped Sets histogram curve style	Options dialog
HSpiceCompatibility	Numeric	Default=0 Value of 0 or 1. Global default value for simulator option (.OPTIONS) of the same name. See <a href="#">“Options Dialog Miscellaneous Options” on page 450</a>	Options dialog
InitSchematicSimulator	Text	Default=SIMetrix Simulator mode for new schematic. If set to SIMPLIS, all schematics will start in SIMPLIS mode. Otherwise they start in SIMetrix mode	Options dialog
InterpOrder	Numeric	Default=2 Sets interpolation order for the FFT calculation used for distortion measurements	
InterpPts	Numeric	Default=1024 Sets number of interpolated points for the FFT calculation used for distortion measurements	
InvertCursors	Boolean	Default=false If true, schematic and graph mouse cursors are modified to be suitable for use on a black background	
LibraryDiagnostics	Text	Default = <i>Full</i> Possible values, <i>Partial</i> , <i>None</i> and <i>Full</i> . Affects progress information displayed during model library searching.	Options dialog

Name	Type	Description	User interface support
LicenseDisableFastCheckout	Boolean	Default=false Affects network licenses only. If set, the license checkout process may take longer delaying the time it takes SIMetrix to start.	
LogicDefExtension	Text	Default=ldf File extension used for logic definition files	Options dialog
MaxHighlightColours	Numeric	Default=4 Maximum number of different colours used for schematic highlighting	
MeasurePrecision	Numeric	Default=8 Precision of measurements in the graph window	
MaxVectorBufferSize	Numeric	Default=32768 See the <i>Simulator Reference Manual</i> for a full explanation	
MinGridHeight	Numeric	Minimum allowed height of graph grid	Options dialog
MinGridWidth	Numeric	Minimum allowed width of graph grid	No
MinorGridPrintWidth	Numeric	Default=0.05 Print width in mm of graph's minor grid	Options dialog
ModelExtension	Text	Default=lb;lib;mod;cir File extensions used for model files	Options dialog
MRUSize	Numeric	Default = 4 Number of items in File   Reopen menu.	Options dialog
NewModelLifetime	Numeric	Default=30 Number of days that user installed models remain displayed in the '* Recently Added Models *' model library browser category	
NoEditPinNamesWarning unsupported	Boolean	Default=false If true, the warning given when using the Edit Pin Names button in the associate models dialog box is inhibited	

Name	Type	Description	User interface support
NoHierarchicalHighlighting	Boolean	Default=false  If set hierarchical schematic highlighting is disabled. That is, if you highlight some part of a hierarchical schematic, only that schematic will be highlighted with no propagation to parent or child schematic	
NoKeys	Boolean	If on, the default key definitions will be disabled. Note this will not take affect until the <i>next</i> session of SIMetrix.	Options dialog
NoMenu	Boolean	If on, the default menu definitions will be disabled and no menu bar will appear. This will not take affect until the next session of SIMetrix.	Options dialog
NonHighlightAlpha	Numeric	Default=25  When highlighting individual divisions in a grouped curve, the divisions that are not highlighted are displayed with a lower alpha value than the highlighted divisions. This is to make the highlighted divisions stand out more clearly. This option defines the akpha value for the divisions that are not highlighted	No
NoStopOnError	Boolean	If disabled, scripts and multi-command lines (i.e. several commands on the same line separated by ‘;’) are aborted if any individual command reports an error.	Options dialog
NoStopOnUnknownParam	Text	Specifies action to be taken in the event of an unknown parameter being encountered in a .MODEL statement. Choices are:  TRUE: No action taken, simulation continues normally FALSE: An error will be raised and the simulation will abort WARN: A warning will be displayed but the simulation will continue  This will be overridden by a .OPTIONS setting of the same name. Refer to <i>Simulator Reference Manual/Command Reference/.OPTIONS</i> for details	Options dialog
OmitAsciiRevision unsupported	Boolean	Default=false  If true, the revision value is not written to ASCII schematic files. For backward compatibility.	No

Name	Type	Description	User interface support
PartSelShowSimplisModels	Boolean	Default=False If true, model library parts will show directly in the part selector in SIMPLIS mode	No
PartSelShowSimetrixModels	Boolean	Default=True If true, model library parts will show directly in the part selector in SIMetrix mode	No
PassUnresTemplate unsupported	Boolean	Default=false If true, unresolved template values in netlists will be passed literally. Default behaviour is no output	No
Precision	Numeric	Default = 10 Precision of numeric values displayed using Show command.	No
PrintOptions	Text	Options set in print dialog	Print dialog
PrintWireWidth	Numeric	Default=5 Width in pixels of schematic wires when printed	No
ProbeFlushOnUpdate	Boolean	Default=false It is not usual to need to set this option. Simulation data is buffered for performance reasons but this buffering can interfere with the incremental updates needed for fixed probes. Usually SIMetrix deals with this problem automatically but this is not guaranteed to work in all cases. In such situations, fixed probes or .GRAPH statements may not incrementally update correctly. Setting this option may rectify this.	
ProbeStartDelay	Numeric	Default = 1 Delay after start of simulation run before fixed probe graphs are first opened.	Options dialog
ProbeUpdatePeriod	Numeric	Default = 0.5 seconds Update period for fixed probe graphs	Options dialog
PSpiceCompatibility	Numeric	Default=1 Value of 0, 1 or 2. Global default value for simulator option (.OPTIONS) of the same name. See <a href="#">“Options Dialog Miscellaneous Options” on page 450</a>	Options dialog
RebuildConfig	Boolean	Default=true See <a href="#">“Auto Configuration Options” on page 470</a>	No

Name	Type	Description	User interface support
RenderNumThreads	Numeric	Default=number of installed processor cores	No
RepeatPlace	Text	Number of threads used to draw multi-division curves Default = <i>Toolbar</i> Controls when schematic placement is repeated. Possible values, <i>Always</i> , <i>Toolbar</i> (toolbar symbols only) and <i>Never</i> .	Options dialog
SchematicReadOnly	Boolean	Default=false If set <i>all</i> schematics are opened in read only mode	No
SchemDoubleClickScript	Text	Default=on_schem_double_click/ne Script that is called when a double click action is detected in the schematic. Only active if SchematicEditMode=NoSnap	
ScriptDir	Text	Default=%SXDOCSPATH%/Scripts %SXDOCSPATH% is “My Documents/SIMetrix” Directory used to search for scripts and symbol files if not found in the current directory. Changes to this option do not take effect until next session.	Options dialog
ScriptExtension	Text	Default=sxscr File extension used for scripts	Options dialog
ShellCommandProcessor	Text	String used to launch command processor when /command supplied with “Shell” script command. See Shell command in the Script Reference manual for more details	
SHOWWELCOME PAGE	Boolean	Default=true. If set a welcome page will be displayed on first start	Welcome page
SimDataGroupDelete	Text	Default=Never Same as DataGroupDelete when simulator is run independently. I.e. not called from the front end	No
SIMPLISPath	Text	Default=%EXEPATH%/simplis.exe Path of SIMPLIS binary	No
SnapshotExtension	Text	Default=sxsnp File extension used for snapshot files	

Name	Type	Description	User interface support
StartUpDir	Text	Current directory set at start of session.	Options dialog
StartUpFile	Text	Default = Startup.SXSCR Script that is automatically run at start of each session.	Options dialog
StatusUpdatePeriod	Numeric	Default = 0.2 seconds Minimum delay in seconds between updates of simulator status window during run.	No
SymbolExtension	Text	Default=sxslb;slb File extension used for symbol files	
SymbolsDir	Text	Default=%SHAREPATH%/SymbolLibs Path of directory where system symbol libraries are located.	Option dialog
SystemStyle	Text	Default=LightStyle.qss. Sets the style sheet that defines the user interface style. May be set to DarkStyle.qss or custom generated style sheet placed in support\ui	<b>File   Options   Interface Colour...</b>
TempDataDir	Text	Default = %TEMPPATH%/SIMetrixTempData  See “ <a href="#">Default Configuration Location</a> ” on <a href="#">page 444</a> for definition of %TEMPPATH%  Directory where temporary simulation data files are placed.	Options dialog
TextExtension	Text	File extension used for text files	
TotalVectorBufferSize	Numeric	See the <i>Simulator Reference Manual</i> for a full explanation	
TitleBlockDefaultAuthor	Text	The default authour for title blocks	Title block interface
TitleBlockDefaultCompany	Text	The default company name for title blocks	Title block interface
TitleBlockDefaultLogoPath	Text	The path of the default logo for title blocks. Should be a bitmap file such as .jpg, .png, .bmp or .gif	Title block interface
ToolBarIconSize	Numeric	Default = 20. Size of toolbar icons in pixels. Takes effect on restart	No
TranscriptErrors	Boolean	Default=false  If true, incorrectly typed commands will be entered in the history box. (The drop down list in the command line that shows previously entered commands)	

Name	Type	Description	User interface support
UndoBufferSize	Numeric	Default = 10 Number of levels of schematic undo. See <a href="#">“Creating a Schematic” on page 32</a>	Options dialog
UpdateClosedSchematics	Boolean	Allows SIMetrix to write to closed schematic if required. See <a href="#">“Closed Schematics” on page 336</a>	
UpdateCurvesNoDeleteOld	Boolean	Default=false If true, old curves are not deleted when using the Update Curves feature.	Plot   Update Curve Settings
UpdateCurvesNoFixSelected	Boolean	Default=false If true update includes selected curves when using the Update Curves feature	Plot   Update Curve Settings
UseLocalHelp	Boolean	If set, the help system will use locally stored content. Note that the local content must be installed for this to work. Menu <b>Help   Select Help Source...</b> will automatically set this option and install local content if needed.	<b>Help   Select Help Source...</b>
UserScriptDir	Text	Alternative location for user scripts. See <i>Script Reference Manual/Script Language/Executing Scripts</i> for more information	No
UserSymbolsDir	Text	Path of directory where the user’s symbol libraries are stored.	Options dialog
UserSystemSymbolDir	Text	Default=%sxappdatapath%/SysSymbols Location of symbol libraries containing edits to system symbols	No
VideoBufferTotalSize	Numeric	Default=100000000 (100MBytes). The total memory allowed to be used for video buffers. The bitmap image of each schematic and graph window is buffered in memory. This allows for fast redraw when tabs are switched or when a window is exposed. Without this buffering, the windows need to be redrawn which in the case of graphs can be time consuming. The buffers consume memory and this setting specifies how much memory may be allowed to be used for the buffers (that is all buffers together, not each one). If all the buffers needed exceed this memory allowance, some windows will be released meaning that a redraw will be needed when that window needs to be displayed.	No

Name	Type	Description	User interface support
VIDataPath	Text	Default=%SXAPPDATAPATH%/veriloghdl Location of Verilog-HDL files. Currently only the cache data is stored here	
VIModuleCacheSize	Numeric	Default=1000 Maximum number of cache entries for Verilog-HDL module info cache	
WarnSubControls	Boolean	Default=false If true, a warning will be issued if unexpected simulator commands are found in subcircuits.	No

## Text Editor Colour Options

The following options set colours for text editor elements. All of these can be set in the Customisation sheet in the options dialog. See menu **File | Options | General...**

Option name	Editor	Element
AsciiFileEditorDoubleQuoteStringColour	Ascii File Editor	Double quoted string
AsciiFileEditorSingleQuoteStringColour	Ascii File Editor	Single quoted string
AsciiFileEditorSubTypesColour	Ascii File Editor	Sub types
AsciiFileEditorTypesColour	Ascii File Editor	Types
AsciiFileEditorValuesColour	Ascii File Editor	Values
LogicDefinitionEditorCommentColour	Logic Definition	comment colour
LogicDefinitionEditorKeywordColour	Logic Definition	keyword colour
NetlistEditorCommentColour	Netlist editor	comment colour
NetlistEditorDoubleQuoteStringColour	Netlist editor	Double quoted string colour
NetlistEditorInstanceColour	Netlist editor	Instance colour
NetlistEditorKeywordColour	Netlist editor	keyword colour
NetlistEditorSingleQuoteStringColour	Netlist editor	Single quoted string colour
NetlistEditorSubCircuitColour	Netlist editor	Subcircuit colour (.SUBCKT and .ENDS)
ScriptEditorBIScriptColour	Script Editor	Built-in script colour
ScriptEditorCommandColour	Script Editor	Command colour
ScriptEditorCommentColour	Script Editor	Comment colour

Option name	Editor	Element
ScriptEditorDoubleQuotedStringColour	Script Editor	double quoted string colour
ScriptEditorFunctionColour	Script Editor	Function colour
ScriptEditorKeywordColour	Script Editor	keyword colour
ScriptEditorStringColour	Script Editor	string colour
TextEditorBackgroundColour	Plain text editor	background colour
TextEditorFontColour	Plain text editor	Text colour
TextEditorHighlightColour	Plain text editor	highlighted colour
VerilogAEditorAnalogOperatorColour	Verilog-A editor	Analog operator colour
VerilogAEditorDoubleQuoteColour	Verilog-A editor	String colour
VerilogAEditorIntegerColour	Verilog-A editor	Integer colour
VerilogAEditorKeywordColour	Verilog-A editor	Keyword colour
VerilogAEditorMultilineCommentColour	Verilog-A editor	Multi-line comment colour
VerilogAEditorPreProcessorColour	Verilog-A editor	Pre-processor colour
VerilogAEditorRealColour	Verilog-A editor	Real value colour
VerilogAEditorRegularFunctionsColour	Verilog-A editor	Function colour
VerilogAEditorSingleLineCommentColour	Verilog-A editor	Singe line comment colour
VerilogAEditorSystemFunctionsColour	Verilog-A editor	System function colour
VerilogHDLEditorDoubleQuoteStringColour	Verilog-HDL editor	String colour
VerilogHDLEditorKeywordColour	Verilog-HDL editor	Keyword colour
VerilogHDLEditorMultipleLineCommentColour	Verilog-HDL editor	Multi-line comment colour
VerilogHDLEditorSingleLineCommentColour	Verilog-HDL editor	Single line comment colour

## Other Colour Options

The following options set colours for various elements. All of these can be set in the Customisation sheet in the options dialog. See menu **File | Options | General...**

Option name	Editor	Element
ShellInputActiveHighlightColour	Command Shell Input	Active highlight
ShellInputBackgroundColour	Command Shell Input	Background
ShellInputHighlightedTextColour	Command Shell Input	Highlighted text
ShellInputInactiveHighlightColour	Command Shell Input	Inactive highlight
ShellInputTextColour	Command Shell Input	Input text

<b>Option name</b>	<b>Editor</b>	<b>Element</b>
ShellOutputActiveHighlightColour	Command Shell Output	Active highlight
ShellOutputBackgroundColour	Command Shell Output	Background
ShellOutputHighlightedTextColour	Command Shell Output	Highlighted text
ShellOutputInactiveHighlightColour	Command Shell Output	Inactive highlight
ShellOutputTextColour	Command Shell Output	Input text
PartSelectorBackgroundColour	Part selector	Background colour
PartSelectorTextColour	Part selector	Text colour
GraphColourText	Graph	Text Colour (default graph text colour)
GraphColourSelectedText	Graph	Selected Text Colour
GraphColourCaption	Graph	Caption Colour
GraphColourFreeText	Graph	Free Text Colour
GraphColourTextBoxText	Graph	Text Box Text Colour
GraphColourTextBoxBG	Graph	Text Box Background
GraphColourLegendBox	Graph	Legend Box Text Colour
GraphColourCurveMarkerText	Graph	Curve Marker Text Colour
GraphColourCurveMarker	Graph	Curve Marker Line Colour
GraphColourRectangle	Graph	Zoom Box Colour
GraphColourCrossHair	Graph	Cursor Colour
GraphColourMajorGrid	Graph	Major Grid
GraphColourMinorGrid	Graph	Minor Grid
GraphColourBackground	Graph	Background
GraphColourAxis	Graph	Axis
GraphColourSelectedAxis	Graph	Selected Axis
GraphColourHighlightedCurve	Graph	Highlighted Curve
GraphColourCursorDimension	Graph	Cursor Dimension
GraphColourCursorDimensionText	Graph	Cursor Dimension Text
GraphColourCurve_1	Curves	Curve 1 Colour
GraphColourCurve_2	Curves	Curve 2 Colour
GraphColourCurve_3	Curves	Curve 3 Colour

Option name	Editor	Element
GraphColourCurve_4	Curves	Curve 4 Colour
GraphColourCurve_5	Curves	Curve 5 Colour
GraphColourCurve_6	Curves	Curve 6 Colour
GraphColourCurve_7	Curves	Curve 7 Colour
GraphColourCurve_8	Curves	Curve 8 Colour
GraphColourCurve_9	Curves	Curve 9 Colour
GraphColourCurve_10	Curves	Curve 10 Colour
GraphColourCurve_11	Curves	Curve 11 Colour
GraphColourCurve_12	Curves	Curve 12 Colour
GraphColourCurve_13	Curves	Curve 13 Colour
GraphColourCurve_14	Curves	Curve 14 Colour
GraphColourCurve_15	Curves	Curve 15 Colour
GraphColourCurve_16	Curves	Curve 16 Colour
GraphColourCurve_17	Curves	Curve 17 Colour
GraphColourCurve_18	Curves	Curve 18 Colour
GraphColourCurve_19	Curves	Curve 19 Colour
GraphColourCurve_20	Curves	Curve 20 Colour

## File Extension

The following options set default file extensions. See Options dialog for more details.

Option name	Default value	Description
AsciiFileEditorExtension	sxsch;sxslib;sxcmp	ASCII schematic and symbol files
ComponentExtension	sxcmp	Schematic component files
DataExtension	sxdat;dat	Data files
GraphExtension	sxgph	Graph binary files
LogicDefExtension	ldf	Arbitrary block logic definition files
ModelExtension	lb;lib;mod;cir	SPICE model files
NetlistExtension	net;cir;deck;out	Netlist files
SchematicExtension	sxsch;sch	Schematic files
ScriptExtension	sxscr;txt	Scripts
SnapshotExtension	sxsnp	Simulator snapshot files
SymbolExtension	lib	Binary symbol files
TextExtension	txt;net;cir;mod;ldf;sxscr;lib;lb;cat	Text files
VerilogAExtension	va;vams	Verilog-A files
VerilogHDLExtension	v	Verilog-HDL files

## Toolbar Buttons

The buttons displayed on each of the standard toolbars are defined with an option variable - that is one for each toolbar. The value of the option consists of a series of semi-colon delimited button names. A complete list of button names and full information concerning user defined toolbars can be found in the *Script Reference Manual/Applications/Creating Modifying Toolbars*. The toolbar option variable names are listed below.

Option name	Description
ComponentButtons	Schematic parts in SIMetrix mode
SIMPLISComponentButtons	Schematic parts in SIMPLIS mode
SchematicMainButtons	Schematic main toolbar
SchematicFileButtons	Schematic file operations toolbar
SchematicProbeButtons	Schematic probe toolbar
SymbolMainButtons	Symbol editor toolbar
SymbolFileButtons	Symbol editor file operations toolbar
GraphMainButtons	Graph window toolbar

## 18.6 Startup Auto Configuration

### 18.6.1 Overview

When SIMetrix is started for the first time, it automatically sets up its configuration to default values. Details of this process are provided in the following sections. There are a number of settings that can be made to control this process and these are also explained.

### 18.6.2 What is Set Up

During this phase, the following is set up:

1. Installs system supplied symbol libraries
2. Installs system supplied model libraries
3. Migrates configuration from earlier installed versions if available
4. Associates file extensions with the operating system.
5. Sets up default window positions according to the system screen resolution (only if preference settings are *not* migrated in 3.)
6. Define default values for various fonts (only if preference settings are *not* migrated in 3.)

### 18.6.3 Auto Configuration Options

Configuration settings are stored in a file called base.sxprj. See [“Configuration Settings” on page 443](#) for details of where this file is located. Auto configuration writes values to this file but will also read values from this file to decide how it will proceed. In the usual sequence of events for installing and setting up SIMetrix, this file will not actually exist when auto configuration occurs. In this case auto configuration uses default values for the settings it tries to read.

However, if you are a system administrator may wish to customise the way SIMetrix is configured when started by each user. In this case you may manually create a base.sxprj file or alternatively a common skeleton that SIMetrix will use to create this file. Your base.sxprj file can, if desired, be completely populated with all required settings and configured to disable auto configuration altogether. Alternatively, you can inhibit some of the auto configuration operations while allowing others to proceed normally.

There are five settings that control auto configuration. These must be placed in the [Options] section of the base.sxprj file. The settings are shown in the following table:

Option Name	Possible Values (Default in bold)	Description
RebuildConfig	<b>true</b> , false	Auto configuration proceeds if this is set to true. Auto configuration automatically sets this to false on completion
BuildPreferenceSettings	<b>askmigrate</b> , true, false	Build user preference settings askmigrate: ask the user whether he wants to migrate settings from an earlier version if available true: Build default values false: do nothing
BuildAssociations	<b>ask</b> , true, false	Build file associations ask: ask the user if he wants file association to be performed true: file associations performed unconditionally false: file associations not performed
BuildModelLibs	<b>ifempty</b> , merge, no	Install system model libraries ifempty: install libraries if there are no libraries currently installed merge: merge system libraries with currently installed libraries no: do not install system libraries
BuildSymbolLibs	<b>ifempty</b> , merge, no	Install system symbol libraries ifempty: install libraries if there are no libraries currently installed merge: merge system libraries with currently installed libraries no: do not install system libraries

The settings in the above table should be placed in the file in [Options] section in the form:

```
[Options]
name = value
```

For example:

```
[Options]
```

```
BuildSymbolLibs=merge
```

## Skeleton Configuration File

The skeleton configuration file, if it exists, will be copied to base.sxprj if base.sxprj does not exist.

The skeleton configuration file must be called skeleton.sxprj and be located in the same directory as the executable file SIMetrix.exe.

### 18.6.4 Installation - Customising

It isn't possible to customise the Windows install program. However, the SIMetrix installer doesn't do much more than simply uncompress files to the chosen location. It is therefore possible for you to create your own SIMetrix install process using a fresh install tree as a source image. You can then add your own files to this including the skeleton.sxprj file described above.

## 18.7 Colours and Fonts

### 18.7.1 Colours

Colours for various objects except in schematics may be defined using the Customisation sheet in the options dialog box. See [“Customisation” on page 453](#). Colours for schematic objects are controlled using styles. Refer to [“Schematic Styles” on page 78](#)

### 18.7.2 Fonts

Fonts for various elements of SIMetrix except schematics may be selected using the font selection dialog box. Fonts for schematic objects are controlled using styles. Refer to [“Schematic Styles” on page 78](#).

To open the font selection dialog box, select menu **File | Options | Font...** menu item. Select the item whose font you wish to change then press **Edit** to select new font. Items available are:

Font object name	Where font is used
Ascii File Editor	Text editor used for direct editing of schematic and symbol files
Associate Model Text	Model display window in associate model dialog box
Command Line	Command line at top of command shell
<b>F11</b> Window	Schematic <b>F11</b> window used for simulator commands
Graph	Graph windows
Graph Caption	Graph Caption objects placed using <b>Annotate   Add Caption</b>
Graph Free Text	Graph Free Text object placed using <b>Annotate   Add Free Text</b>
Graph Text Box	Graph Text Box object placed using <b>Annotate   Add Text Box</b>
Legend Box	Graph Legend Box object placed using <b>Annotate   Add Legend Box</b>
Message Window	Bottom part of command shell
Netlist Editor	Text editor used for editing netlists and model files
Script Editor	Text editor used for editing scripts

Font object name	Where font is used
Text Editor	Text editor used for editing general text files
Verilog-A Editor	Text editor used for editing Verilog-A source files
Verilog-HDL Editor	Text editor used for editing Verilog-HDL source files
View File Window	Window opened with ViewFile command

## 18.8 Startup Script

The startup script is executed automatically each time SIMetrix is launched. By default it is called startup.sxscr but this name can be changed in the options dialog box (**File | Options | General...**). The startup file may reside in the script directory (defined by ScriptDir option variable) or in a user script directory (defined by UserScriptDir option variable).

The most common use for the startup script is to define custom menus and keys but any commands can be placed there.

To edit the startup script, select the **File | Options | Edit Startup Script** menu item.

## 18.9 International Character Sets

SIMetrix schematic editor, waveform viewer and text editors support international character sets. This means that any language can be displayed in these tools. This section describes some of the issues with regard to International character support.

### 18.9.1 Character Encoding

The choice of character encoding scheme affects how international character sets are stored and displayed. The simplest encoding scheme is ASCII but this only supports a total of 95 printable characters. This is not adequate for displaying anything other than English text.

Older versions of SIMetrix used code page based encoding which required a language selection to be made at the system level. For example, a Japanese user would select the Japanese code page and would then be able to display Japanese characters in schematics and the like. While this worked reasonably well, a schematic drawn by a Japanese user would not display correctly in a system setup for a user whose native language is not Japanese.

Code page based encoding has been superseded by the UNICODE character encoding scheme. UNICODE is a scheme that supports all character sets worldwide and does not require the setting up of system-level code pages. There are a number of variants of UNICODE distinguished by the method of encoding its approximately 1 million characters to byte sequences. By far the most widely used is UTF-8.

UTF-8 uses 8 bit character encoding and is compatible with ASCII, that is, ASCII is a subset of UTF-8. A text file containing pure ASCII characters is also a valid UTF-8 file.

### 18.9.2 SIMetrix Character Encoding

SIMetrix version 8 and later uses UTF-8 for schematics, graphs, text editors and a few other things. This means that you can enter text in any language in a schematic and be sure that it will be displayed correctly on any system without needing a code page to be setup.

### 18.9.3 Compatibility with Older Versions

As noted previously, older versions of SIMetrix (version 7.2 and earlier) used code page based encoding and therefore are not compatible with UTF-8. However, we have developed some level of compatibility to allow interoperability between old and new versions. Note the following:

1. Schematic and graph files created with older versions will render international character sets correctly provided the system is setup with the correct code page. For example, if a schematic displayed correctly in version 7.2, it will display correctly in version 8 and later provided it is opened on the same system or a system with the same code page set up.
2. Version 7.2m or later have some UTF-8 support allowing schematics created with version 8 to display correctly in version 7.2. You must update to the most recent version of 7.2 to use this feature.

### 18.9.4 Text Editors

SIMetrix text editors will detect non-UTF-8 characters in a file and attempt to interpret them correctly. By default, the text editors will assume that the file is encoded using the local code-page. You can change this using the SetDefaultEncoding command. Refer to the Script Reference Manual for further information.

## 18.10 Installation and Licensing

Full installation and licensing instructions may be found at:

<http://www.simetrix.co.uk/app/users.htm>

### 18.10.1 Install CD

The install CD contains the installation files but also contains other useful material such as model documentation and script source. We no longer supply a physical CD, but the full CD content may be accessed from our web site. There is a web browseable version as well as a downloadable ISO file allowing you to burn your own CD.

The CD content may be found at our web site by navigating to:

<http://www.simetrix.co.uk/app/product-installation.htm>

From there you should be able to find the download links page for the version of interest. Be aware that a username and password is required to access the download links pages. Please contact [support@simetrix.co.uk](mailto:support@simetrix.co.uk) for assistance.

