Contents

I Ngspice User's Manual 27

1 Introduction 33

1.1 Simulation Algorithms 34

1.1.1 Analog Simulation 34

1.1.2 Matrix solvers 34

1.1.3 Device Models for Analog Simulation 35

1.1.4 Digital Simulation 35

1.1.5 Mixed-Signal Simulation 35

1.1.6 Mixed-Level Simulation (Electronic and TCAD) 36

1.2 Supported Analyses 37

1.2.1 DC Analysis 38

1.2.2 AC Small-Signal Analysis 38

1.2.3 Transient Analysis 38

1.2.4 Pole-Zero Analysis 39

1.2.5 Small-Signal Distortion Analysis 39

1.2.6 Sensitivity Analysis 39

1.2.7 Noise Analysis 40

1.2.8 Periodic Steady State Analysis 40

1.3 Analysis at Different Temperatures 40

1.3.1 Introduction 40

1.3.2 Controlling the temperature 42

1.4 Convergence 43

1.4.1 Voltage convergence criterion 43

1.4.2 Current convergence criterion 44

1.4.3 Convergence failure 44

2 Circuit Description 45

2.1 General Structure and Conventions . . . 45
2.1.1 Input file structure . . . 45
2.1.2 Syntax check . . . 45
2.1.3 Circuit elements (device instances) . . . 46
2.1.4 Some naming conventions . . . 47
2.1.5 Topological constraints . . . 49
2.2 Dot commands . . . 49
2.3 Basic lines . . . 51
2.3.1 .TITLE line . . . 51
2.3.2 .END Line . . . 51
2.3.3 Comments . . . 52
2.3.4 End-of-line comments . . . 52
2.3.5 Continuation lines . . . 52
2.4 .MODEL Device Models . . . 53
2.5 .SUBCKT Subcircuits . . . 54
2.5.1 .SUBCKT Line . . . 54
2.5.2 .ENDS Line . . . 55
2.5.3 Subcircuit Calls . . . 55
2.6 .GLOBAL . . . 56
2.7 .INCLUDE . . . 56
2.8 .LIB . . . 56
2.9 .PARAM Parametric netlists . . . 57
2.9.1 .param line . . . 57
2.9.2 Brace expressions in circuit elements: . . . 58
2.9.3 Subcircuit parameters . . . 58
2.9.4 Symbol scope . . . 59
2.9.5 Syntax of expressions . . . 59
2.9.6 Reserved words . . . 63
2.9.7 A word of caution on the three ngspice expression parsers . . . 63
2.10 .FUNC . . . 63
2.11 .CSPARAM . . . 64
2.12 .TEMP . . . 64
2.13 .IF Condition-Controlled Netlist . . . 65
2.14 Parameters, functions, expressions, and command scripts . . . 66
2.14.1 Parameters . . . 66
2.14.2 Nonlinear sources . . . 66
2.14.3 Control commands, Command scripts . . . 66

3 Circuit Elements and Models 69

3.1 About netlists, device instances, models and model parameters 69

3.2 General options 71

3.2.1 Paralleling devices with multiplier m 71

3.2.2 Instance and model parameters 73

3.2.3 Model binning 73

3.2.4 Initial conditions 73

3.3 Elementary Devices 74

3.3.1 Resistors 74

3.3.2 Semiconductor Resistors 76

3.3.3 Semiconductor Resistor Model (R) 76

3.3.4 Resistors, dependent on expressions (behavioral resistor) 78

3.3.5 Resistor with nonlinear r2_cmc or r3_cmc models 78

3.3.6 Capacitors 79

3.3.7 Semiconductor Capacitors 80

3.3.8 Semiconductor Capacitor Model (C) 80

3.3.9 Capacitors, dependent on expressions (behavioral capacitor) 81

3.3.10 Inductors 83

3.3.11 Inductor model 83

3.3.12 Coupled (Mutual) Inductors 85

3.3.13 Inductors, dependent on expressions (behavioral inductor) 85

3.3.14 Capacitor or inductor with initial conditions 86

3.3.15 Switches 87

3.3.16 Switch Model (SW/CSW) 88

4 Voltage and Current Sources 91

4.1 Independent Sources for Voltage or Current 91

4.1.1 Pulse 92

4.1.2 Sinusoidal 93

4.1.3 Exponential 94

4.1.4 Piece-Wise Linear 94

4.1.5 Single-Frequency FM 95

4.1.6 Amplitude modulated source (AM) 95

4.1.7 Transient noise source 96

4.1.8 Random voltage source 97

4.1.9 External voltage or current input 98
4.1.10 Arbitrary Phase Sources 98
4.1.11 RF Port 99
4.2 Linear Dependent Sources 99
4.2.1 Gxxxx: Linear Voltage-Controlled Current Sources (VCCS) 99
4.2.2 Exxxx: Linear Voltage-Controlled Voltage Sources (VCVS) 100
4.2.3 Fxxxx: Linear Current-Controlled Current Sources (CCCS) 100
4.2.4 Hxxxx: Linear Current-Controlled Voltage Sources (CCVS) 100
4.2.5 Polynomial Source Compatibility 101

5 Non-linear Dependent Sources (Behavioral Sources) 103
5.1 Bxxxx: Nonlinear dependent source (ASRC) 103
5.1.1 Syntax and usage 103
5.1.2 Special B-Source Variables time, temper, hertz 107
5.1.3 par(*'expression'*) 107
5.1.4 Piecewise Linear Function: pwl 107
5.2 Exxxx: non-linear voltage source 110
5.2.1 VOL 110
5.2.2 VALUE 111
5.2.3 TABLE 111
5.2.4 POLY 111
5.2.5 LAPLACE 111
5.2.6 FREQ 112
5.2.7 AND/OR/NAND/NOR 113
5.3 Gxxxx: non-linear current source 113
5.3.1 CUR 113
5.3.2 VALUE 114
5.3.3 TABLE 114
5.3.4 POLY 114
5.3.5 LAPLACE 114
5.3.6 FREQ 115
5.3.7 Example 115
5.4 Debugging a behavioral source 115
5.5 POLY Sources 116
5.5.1 E voltage source, G current source 117
5.5.2 F voltage source, H current source 117

6 Transmission Lines 119
6.1 Lossless Transmission Lines . . . 119
6.2 Lossy Transmission Lines . . . 120
6.2.1 Lossy Transmission Line Model (LTRA) . . . 120
6.3 Uniform Distributed RC Lines . . . 122
6.3.1 Uniform Distributed RC Model (URC) . . . 122
6.4 KSPICE Lossy Transmission Lines . . . 123
6.4.1 Single Lossy Transmission Line (TXL) . . . 124
6.4.2 Coupled Multiconductor Line (CPL) . . . 124

7 Device Models 127
7.1 Instance lines and .model lines . . . 127
7.2 Junction Diodes . . . 128
7.2.1 Diode Model (D) . . . 128
7.2.2 Diode Equations . . . 132
7.2.3 Diode models available via OpenVAF/OSDI . . . 136
7.3 BJT . . . 137
7.3.1 Bipolar Junction Transistors (BJTs) . . . 137
7.3.2 BJT Models (NPN/PNP) . . . 138
7.3.3 Gummel-Poon Models . . . 138
7.3.4 VBIC Model . . . 144
7.3.5 HICUM level 2 Model . . . 145
7.3.6 BJT models available via OpenVAF/OSDI . . . 146
7.4 JFETs . . . 147
7.4.1 Junction Field-Effect Transistors (JFETs) . . . 147
7.4.2 JFET Models (NJF/PJF) . . . 148
7.4.3 Basic model statement . . . 148
7.4.4 JFET level 1 model with Parker Skellern modification . . . 148
7.4.5 JFET level 2 Parker Skellern model . . . 150
7.5 MESFETs . . . 152
7.5.1 MESFET devices . . . 152
7.5.2 MESFET Models (NMF/PMF) . . . 152
7.5.3 Model by Statz e.a. . . . 152
7.5.4 Model by Ytterdal e.a. . . . 153
7.5.5 hfet1 and hfet2 . . . 153

7.6 MOSFETs 154
7.6.1 MOSFET devices 154
7.6.2 MOSFET models (NMOS/PMOS) 155
7.6.3 BSIM Models 160
7.6.4 BSIMSOI models (levels 10, 58, 55, 56, 57) 164
7.6.5 SOI3 model (level 60) 164
7.6.6 HiSIM models of the University of Hiroshima 164
7.6.7 MOS models available via OpenVAF/OSDI 164
7.7 Power MOSFET model (VDMOS) 165

8 Mixed-Mode and Behavioral Modeling with XSPICE 173

8.1 Code Model Element & .MODEL Cards 173
8.1.1 Syntax 173
8.1.2 Examples 177
8.1.3 Search path for file input 178
8.1.4 Code model location and assessment 178
8.2 Analog Models 179
8.2.1 Gain 179
8.2.2 Summer 180
8.2.3 Multiplier 181
8.2.4 Divider 182
8.2.5 Limiter 183
8.2.6 Controlled Limiter 185
8.2.7 PWL Controlled Source 186
8.2.8 PWL Time Controlled Source with optional edge smoothing 188
8.2.9 Filesource (PWL sourced from file) 191
8.2.10 Multi_input_PWL_block 192
8.2.11 Analog Switch 193
8.2.12 Alternative Analog Switch 195
8.2.13 Zener Diode 196
8.2.14 Current Limiter 197
8.2.15 Hysteresis Block 200
8.2.16 Differentiator 202
8.2.17 Integrator 203
8.2.18 S-Domain Transfer Function 204

8.2.19 PWL Transfer Function . . . 207
8.2.20 Slew Rate Block . . . 209
8.2.21 Inductive Coupling . . . 210
8.2.22 Magnetic Core . . . 211
8.2.23 Controlled Sine Wave Oscillator . . . 215
8.2.24 Controlled Triangle Wave Oscillator . . . 216
8.2.25 Controlled Square Wave Oscillator . . . 217
8.2.26 Controlled One-Shot . . . 219
8.2.27 Capacitance Meter . . . 221
8.2.28 Inductance Meter . . . 222
8.2.29 Memristor . . . 222
8.2.30 2D table model . . . 223
8.2.31 3D table model . . . 225
8.2.32 Simple Diode Model . . . 227
8.2.33 Analog delay . . . 229
8.2.34 Potentiometer . . . 230
8.3 Hybrid Models . . . 232
8.3.1 Digital-to-Analog Node Bridge . . . 232
8.3.2 Analog-to-Digital Node Bridge . . . 233
8.3.3 Bidirectional Analog/Digital Node Bridge . . . 235
8.3.4 Controlled Digital Oscillator . . . 238
8.3.5 Node bridge from digital to real with enable . . . 239
8.3.6 A Z**-1 block working on real data . . . 240
8.3.7 A gain block for event-driven real data . . . 240
8.3.8 Node bridge from real to analog voltage . . . 241
8.3.9 Controlled PWM Oscillator . . . 241
8.4 Digital Models . . . 243
8.4.1 Buffer . . . 245
8.4.2 Inverter . . . 245
8.4.3 And . . . 246
8.4.4 Nand . . . 247
8.4.5 Or . . . 247
8.4.6 Nor . . . 248
8.4.7 Xor . . . 248
8.4.8 Xnor . . . 249

8.4.9 Tristate 249
8.4.10 Pullup 251
8.4.11 Pulldown 251
8.4.12 D Flip Flop 252
8.4.13 JK Flip Flop 254
8.4.14 Toggle Flip Flop 256
8.4.15 Set-Reset Flip Flop 258
8.4.16 D Latch 260
8.4.17 Set-Reset Latch 262
8.4.18 State Machine 264
8.4.19 Frequency Divider 267
8.4.20 RAM 268
8.4.21 Digital Source 271
8.4.22 LUT 272
8.4.23 General LUT 273
8.4.24 D_process 275
8.4.25 d_cosim 277
8.5 Predefined Node Types for event driven simulation 279
8.5.1 Digital Node Type 279
8.5.2 Real Node Type 279
8.5.3 Int Node Type 280
8.5.4 (Digital) Input/Output 280
8.6 Automatic insertion of bridging devices 280

9 Verilog-A Compact Device Models 283
9.1 Introduction 283
9.2 OSDI/OpenVAF 283
9.3 How to create and apply OpenVAF models 284
9.3.1 Preparing for simulation 284
9.3.2 OSDI/OpenVAF examples distributed with ngspice 286

10 Digital Device Models 287
10.1 U devices (basic digital building blocks) 287
10.1.1 General format 288
10.1.2 List of devices available in ngspice (basic types) 288
10.1.3 URC transmission line versus U devices 289

10.2 Support for standard digital devices 289
10.3 Digital devices defined by a Hardware Description Language 290
10.3.1 Using Verilator, Verilog, and code model d_cosim 290
10.3.2 Using independent processes (e.g. C coded), pipes, and code model d_process 291
10.3.3 Using Yosys to map digital Verilog onto basic code model cells 291

11 Analyses and Output Control (batch mode) 293

11.1 Simulator Variables (.options) 293
11.1.1 General Options 294
11.1.2 OP and DC Solution Options 295
11.1.3 AC Solution Options 297
11.1.4 Transient Analysis Options 297
11.1.5 ELEMENT Specific options 298
11.1.6 Transmission Lines Specific Options 299
11.1.7 Precedence of option and .options commands 299
11.2 Initial Conditions 299
11.2.1 .NODESET: Specify Initial Node Voltage Guesses 299
11.2.2 .IC: Set Initial Conditions 300
11.3 Analyses 300
11.3.1 .AC: Small-Signal AC Analysis 300
11.3.2 .DC: DC Transfer Function 302
11.3.3 .DISTO: Distortion Analysis 302
11.3.4 .NOISE: Noise Analysis 304
11.3.5 .OP: Operating Point Analysis 305
11.3.6 .PZ: Pole-Zero Analysis 306
11.3.7 .SENS: DC or Small-Signal AC Sensitivity Analysis 307
11.3.8 .SP S-Parameter Analysis 307
11.3.9 .TF: Transfer Function Analysis 308
11.3.10 .TRAN: Transient Analysis 309
11.3.11 Transient noise analysis (at low frequency) 309
11.3.12 .PSS: Periodic Steady State Analysis 313
11.4 Measurements after AC, DC and Transient Analysis 314
11.4.1 .meas(ure) 314
11.4.2 batch versus interactive mode 314

11.4.3 General remarks . . . 314
11.4.4 Input . . . 315
11.4.5 Trig Targ . . . 315
11.4.6 Find ... When . . . 317
11.4.7 `AVG|MIN|MAX|PP|RMS|MIN_AT|MAX_AT` . . . 318
11.4.8 Integ . . . 318
11.4.9 param . . . 319
11.4.10 par(*'expression'*) . . . 319
11.4.11 Deriv . . . 320
11.4.12 More examples . . . 320
11.5 Safe Operating Area (SOA) warning messages . . . 321
11.5.1 Resistor and Capacitor SOA model parameters . . . 322
11.5.2 Diode SOA model parameters . . . 322
11.5.3 BJT SOA model parameters . . . 323
11.5.4 MOS SOA model parameters . . . 324
11.5.5 VDMOS SOA model parameters . . . 325
11.6 Batch Output . . . 325
11.6.1 .SAVE: Name vector(s) to be saved in raw file . . . 325
11.6.2 .PRINT Lines . . . 326
11.6.3 .PLOT Lines . . . 327
11.6.4 .FOUR: Fourier Analysis of Transient Analysis Output . . . 327
11.6.5 .PROBE: Save device node currents, device power dissipation, or differential voltages between arbitrary nodes . . . 328
11.6.6 par(*'expression'*): Algebraic expressions for output . . . 332
11.6.7 .width . . . 333
11.7 Measuring current through device terminals . . . 333
11.7.1 Using the .probe command . . . 333
11.7.2 Adding a voltage source in series . . . 333
11.7.3 Using option 'savecurrents' . . . 334

12 Starting ngspice 335
12.1 Introduction . . . 335
12.2 Where to obtain ngspice . . . 335
12.3 Command line options for starting ngspice . . . 336
12.4 Starting options . . . 338

12.4.1 Batch mode . . . 338
12.4.2 Interactive mode . . . 338
12.4.3 Control mode (Interactive mode with control file or control section) . . 339
12.5 Standard configuration file spinit . . . 340
12.6 User defined configuration file .spiceinit . . . 342
12.7 Environmental variables . . . 343
12.7.1 Ngspice specific variables . . . 343
12.7.2 Common environment variables . . . 344
12.8 Memory usage . . . 344
12.9 Simulation time . . . 344
12.10Ngspice on multi-core processors using OpenMP . . . 345
12.10.1 Introduction . . . 345
12.10.2 Internals . . . 345
12.10.3 Some results . . . 346
12.10.4 Usage . . . 346
12.10.5 Literature . . . 347
12.11Server mode option -s . . . 347
12.12Pipe mode option -p . . . 349
12.13Ngspice control via input, output fifos . . . 350
12.14Compatibility . . . 351
12.14.1 Compatibility mode . . . 351
12.14.2 Missing functions . . . 352
12.14.3 Devices . . . 352
12.14.4 Controls and commands . . . 353
12.14.5 PSPICE Compatibility mode . . . 354
12.14.6 LTSPICE Compatibility mode . . . 355
12.14.7 LTSPICE/PSPICE Compatibility mode . . . 357
12.14.8 KiCad Compatibility mode . . . 357
12.14.9 Spectre Compatibility mode . . . 358
12.14.10HSPICE Compatibility mode . . . 358
12.15Tests . . . 358
12.16Tools for debugging a circuit netlist . . . 359
12.16.1 options and initial conditions . . . 359
12.16.2 set debug . . . 359
12.16.3 set ngdebug . . . 359
12.16.4 miscellaneous . . . 360
12.17Reporting bugs and errors . . . 360

13 Interactive Interpreter 361

13.1 Introduction . . . 361

13.2 Expressions, Functions, and Constants . . . 362

13.3 Plots . . . 366

13.4 Command Interpretation . . . 367

13.4.1 On the console . . . 367

13.4.2 Scripts . . . 367

13.4.3 Add-on to circuit file . . . 367

13.5 Commands . . . 368

13.5.1 Ac: Perform an AC, small-signal frequency response analysis . . . 368

13.5.2 Alias: Create an alias for a command . . . 369

13.5.3 Alter: Change a device or model parameter . . . 369

13.5.4 Altermod: Change model parameter(s) . . . 371

13.5.5 Alterparam: Change value of a global parameter . . . 372

13.5.6 Asciiplot: Plot values using old-style character plots . . . 373

13.5.7 Aspice*: Asynchronous ngspice run . . . 373

13.5.8 Bg_ctrl**: suspend running controls until bg_run has finished . . . 373

13.5.9 Bg_halt**: halt a run . . . 373

13.5.10 Bg_run**: Run analysis from the input file in the background thread . . . 374

13.5.11 Bug: Output URL for ngspice bug tracker . . . 374

13.5.12 Cd: Change directory . . . 374

13.5.13 Cdump: Dump the control flow to the screen . . . 374

13.5.14 Circbyline: Enter a circuit line by line . . . 375

13.5.15 Codemodel: Load an XSPICE code model library . . . 376

13.5.16 Compose: Compose a vector . . . 377

13.5.17 Cutout: Cut out a section of all vectors in a tran plot . . . 378

13.5.18 Dc: Perform a DC-sweep analysis . . . 378

13.5.19 Define: Define a function . . . 378

13.5.20 Deftype: Define a new type for a vector or plot . . . 379

13.5.21 Delete: Remove a trace or breakpoint . . . 379

13.5.22 Destroy: Delete an output data set . . . 379

13.5.23 Devhelp: information on available devices . . . 380

13.5.24 Diff: Compare vectors . . . 381

13.5.25 Display: List known vectors and types . . . 381

13.5.26 Echo: Print text . . . 381

13.5.27 Edit*: Edit the current circuit 382
13.5.28 Edisplay: Print a list of all the event nodes 382
13.5.29 Eprint: Print an event driven node 382
13.5.30 Eprvcd: Dump nodes in VCD format 382
13.5.31 Esave: Save a set of event node outputs 383
13.5.32 Fclose: close an open file handle 383
13.5.33 FFT: fast Fourier transform of vectors 383
13.5.34 Fopen: open a text file 385
13.5.35 Fourier: Perform a Fourier transform 385
13.5.36 Fread: read into a variable from a text file 386
13.5.37 Getcwd: Print the current working directory 387
13.5.38 Gnuplot: Graphics output via gnuplot 387
13.5.39 Hardcopy: Save a plot to a file for printing 387
13.5.40 Help: Print summaries of Ngspice commands 387
13.5.41 History: Review previous commands 388
13.5.42 Inventory: Print circuit inventory 390
13.5.43 Iplot*: Incremental plot 391
13.5.44 Jobs*: List active asynchronous ngspice runs 391
13.5.45 Let: Assign a value to a vector 391
13.5.46 Linearize: Interpolate to a linear scale 392
13.5.47 Listing: Print a listing of the current circuit 393
13.5.48 Load: Load rawfile data 393
13.5.49 Mc_source: Reload the circuit netlist from an internal storage 394
13.5.50 Meas: Measurements on simulation data 394
13.5.51 Mdump: Dump the matrix values to a file (or to console) 395
13.5.52 Mrdump: Dump the matrix right hand side values to a file (or to console) 395
13.5.53 Noise: Noise analysis 395
13.5.54 Op: Perform an operating point analysis 396
13.5.55 Option: Set a ngspice option 396
13.5.56 Plot*: Plot vectors on the display 397
13.5.57 Pre_<command>: execute commands prior to parsing the circuit 399
13.5.58 Pre_OSDI: load a *.osdi compact device model shared library 399
13.5.59 Print: Print values 399
13.5.60 Psd: power spectral density of vectors 400
13.5.61 Quit: Leave Ngspice 400

13.5.62 Rehash: Reset internal hash tables 401
13.5.63 Remcirc: Remove the current circuit 401
13.5.64 Remzerovec: Remove zero length vectors 401
13.5.65 Reset: Reset an analysis 401
13.5.66 Reshape: Alter the dimensionality or dimensions of a vector 402
13.5.67 Resume: Continue a simulation after a stop 402
13.5.68 Rspice*: Remote ngspice submission 403
13.5.69 Run: Run analysis from the input file 403
13.5.70 Rusage: Resource usage 403
13.5.71 Save: Save a set of outputs 404
13.5.72 Sens: Run a sensitivity analysis 406
13.5.73 Set: Set the value of a variable 406
13.5.74 Setcs: Set the value of a variable, case preserved 407
13.5.75 Setcirc: Change the current circuit 407
13.5.76 Setplot: Switch the current set of vectors 408
13.5.77 Setscale: Set the scale vector for the current plot 408
13.5.78 Setseed: Set the seed value for the random number generator 408
13.5.79 Settype: Set the type of a vector 409
13.5.80 Shell: Call the command interpreter 409
13.5.81 Shift: Alter a list variable 409
13.5.82 Show: List device state 410
13.5.83 Showmod: List model parameter values 410
13.5.84 Snload: Load the snapshot file 411
13.5.85 Snsave: Save a snapshot file 411
13.5.86 Source: Read a ngspice input file 412
13.5.87 Sp: S-Parameter Analysis 413
13.5.88 Spec: Create a frequency domain plot 413
13.5.89 Status: Display breakpoint information 414
13.5.90 Step: Run a fixed number of time-points 414
13.5.91 Stop: Set a breakpoint 414
13.5.92 Strcmp: Compare two strings 415
13.5.93 Strslice: Extract a substring 415
13.5.94 Strstr: Find a substring 415
13.5.95 Sysinfo: Print system information 415
13.5.96 Tf: Run a Transfer Function analysis 416

13.5.97 Trace: Trace nodes 417
13.5.98 Tran: Perform a transient analysis 417
13.5.99 Transpose: Swap the elements in a multi-dimensional data set 418
13.5.100Unalias: Retract an alias 418
13.5.101Undefine: Retract a definition 418
13.5.102Unlet: Delete the specified vector(s) 418
13.5.103Unset: Clear a variable 419
13.5.104Version: Print the version of ngspice 419
13.5.105Where: Identify troublesome node or device 420
13.5.106Wrdata: Write data to a file (simple table) 421
13.5.107Write: Write data to a file (Spice3f5 format) 421
13.5.108Wrnodev: Write node voltage values to a file (.ic=xx format) 422
13.5.109Wrs2p: Write scattering parameters to file (Touchstone® format) . . . 423
13.6 Control Structures 423
13.6.1 While - End 423
13.6.2 Repeat - End 424
13.6.3 Dowhile - End 425
13.6.4 Foreach - End 425
13.6.5 If - Then - Else 426
13.6.6 Label 426
13.6.7 Goto 426
13.6.8 Continue 427
13.6.9 Break 427
13.7 Internally predefined variables 427
13.8 Scripts 435
13.8.1 Variables 436
13.8.2 Vectors 437
13.8.3 Assessing vectors in subcircuits 437
13.8.4 Commands 438
13.8.5 control structures 438
13.8.6 Example script 'spectrum' 442
13.8.7 Example script for random numbers 444
13.8.8 Parameter sweep 445
13.8.9 Output redirection 445
13.9 Scattering parameters (S-parameters) 447

13.9.1 Intro . . . 447
13.9.2 S-parameter measurement basics . . . 447
13.9.3 Usage of .sp and sp . . . 449
13.9.4 Usage of the script . . . 449
13.10 Using shell variables . . . 449
13.11 MISCELLANEOUS . . . 450
13.12 Bugs . . . 450

14 Ngspice User Interfaces 451
14.1 MS Windows Graphical User Interface . . . 451
14.2 MS Windows Console . . . 454
14.3 Linux . . . 454
14.4 CygWin . . . 454
14.5 Error handling . . . 455
14.6 Output-to-file options . . . 455
14.6.1 Graphics files . . . 455
14.6.2 Tabulated files . . . 461
14.7 Gnuplot . . . 464
14.7.1 Using Gnuplot to produce 1D graphs of (electrical) simulation results . 464
14.7.2 Using gnuplot to produce 2D contour plots for Cider . . . 465
14.8 Integration with CAD software and ‘third party’ GUIs . . . 469
14.8.1 KiCad . . . 469
14.8.2 Xschem . . . 469
14.8.3 Qucs-S . . . 469
14.8.4 GNU Spice GUI . . . 469
14.8.5 XCircuit . . . 470
14.8.6 GEDA . . . 470
14.8.7 MSEspice . . . 470
14.8.8 GNU Octave . . . 470

15 ngspice as shared library or dynamic link library 471
15.1 Compile options . . . 471
15.1.1 How to get the sources . . . 471
15.1.2 Linux, MINGW, CYGWIN . . . 471
15.1.3 MS Visual Studio . . . 472
15.2 Linking shared ngspice to a calling application . . . 472

15.2.1 Linking during creating the caller 472
15.2.2 Loading at runtime 472
15.3 Shared ngspice API 472
15.3.1 structs and types defined for transporting data 472
15.3.2 Exported functions 474
15.3.3 Callback functions 477
15.4 General remarks on using the API 480
15.4.1 Loading a netlist 480
15.4.2 Running the simulation 482
15.4.3 Accessing data 482
15.4.4 Altering model or device parameters 483
15.4.5 Output 484
15.4.6 Error handling 484
15.5 Example applications 484
15.6 ngspice parallel 484
15.6.1 Go parallel! 485
15.6.2 Additional exported functions 486
15.6.3 Additional callback functions 487
15.6.4 Parallel ngspice example 488

16 TCLspice 489

16.1 tclspice framework 489
16.2 tclspice documentation 489
16.3 spicetoblt 489
16.4 Running TCLspice 490
16.5 examples 490
16.5.1 Active capacitor measurement 490
16.5.2 Optimization of a linearization circuit for a Thermistor 493
16.5.3 Progressive display 497
16.6 Compiling 498
16.6.1 Linux 498
16.6.2 MS Windows 498
16.7 MS Windows 32 Bit binaries 499

17 Example Circuits **501**
17.1 AC coupled transistor amplifier . . . 501
17.2 Differential Pair . . . 507
17.3 MOSFET Characterization . . . 507
17.4 RTL Inverter . . . 507
17.5 Four-Bit Binary Adder (Bipolar) . . . 508
17.6 Four-Bit Binary Adder (MOS) . . . 510
17.7 Transmission-Line Inverter . . . 511

18 Statistical circuit analysis **513**
18.1 Introduction . . . 513
17.4 RTL Inverter
18.2 Using random param(eters) . . . 513
18.3 Behavioral sources (B, E, G, R, L, C) with random control . . . 515
18.4 ngspice control language . . . 516
18.5 Monte-Carlo Simulation . . . 517
18.5.1 Varying model or instance parameters . . . 518
18.5.2 Using the ngspice control language . . . 518
18.6 Data evaluation with Gnuplot . . . 520

19 Circuit optimization with ngspice **523**
19.1 Optimization of a circuit . . . 523
19.2 ngspice optimizer using ngspice scripts . . . 524
19.3 ngspice optimizer using tclspice . . . 524
19.4 ngspice optimizer using a Python script . . . 524
19.5 ngspice optimizer using ASCO . . . 524
19.5.1 Three stage operational amplifier . . . 525
19.5.2 Digital inverter . . . 526
19.5.3 Bandpass . . . 528
19.5.4 Class-E power amplifier . . . 528

20 Notes **529**
20.1 Glossary . . . 529
20.2 Acronyms and Abbreviations . . . 530
20.3 To Do . . . 531