SPICE3 Version 3f3 User's Manual

May, 1993

T. Quarles
A.R.Newton, D.O.Pederson, A.Sangiovanni-Vincentelli

Department of Electrical Engineering and Computer Sciences
University of California
Berkeley, Ca., 94720

Copyright 1990, 1991, 1992, 1993 Regents of the University of California. All rights reserved.
This manual may be reproduced in whole or in part without permission
for any purpose other than resale for profit, if this copyright notice
is included; personal, educational, or internal business use is
specifically permitted as long as the copyright notice is preserved.

TABLE OF CONTENTS

Release Notes ix
1. INTRODUCTION 1
1. TYPES OF ANALYSIS 1
1. DC Analysis 1
2. AC Small-Signal Analysis 2
3. Transient Analysis 2
4. Pole-Zero Analysis 2
5. Small-Signal Distortion Analysis 2
6. Sensitivity Analysis 3
7. Noise Analysis 3
2. ANALYSIS AT DIFFERENT TEMPERATURES 4
3. CONVERGENCE 6
2. CIRCUIT DESCRIPTION 7
1. GENERAL STRUCTURE AND CONVENTIONS 7
2. TITLE LINE, COMMENT LINES AND .END LINE 8
1. Title Line 8
2. .END Line 8
3. Comments 8
3. DEVICE MODELS 9
4. SUBCIRCUITS 10
1. .SUBCKT Line 10
2. .ENDS Line 10
3. Subcircuit Calls 11
5. COMBINING FILES: .INCLUDE LINES 12
3. CIRCUIT ELEMENTS AND MODELS 13
1. ELEMENTARY DEVICES 13
1. Resistors 13
2. Semiconductor Resistors 13
3. Semiconductor Resistor Model (R) 14
4. Capacitors 15
5. Semiconductor Capacitors 15
6. Semiconductor Capacitor Model (C) 16
7. Inductors 16
8. Coupled (Mutual) Inductors 17
9. Switches 17
10. Switch Model (SW/CSW) 18
2. VOLTAGE AND CURRENT SOURCES 19
1. Independent Sources 19
1. Pulse 20

2. Sinusoidal 21
3. Exponential 21
4. Piece-Wise Linear 22
5. Single-Frequency FM 23
2. Linear Dependent Sources 23
1. Linear Voltage-Controlled Current Sources 24
2. Linear Voltage-Controlled Voltage Sources 24
3. Linear Current-Controlled Current Sources 24
4. Linear Current-Controlled Voltage Sources 25
3. Non-linear Dependent Sources 25
3. TRANSMISSION LINES 27
1. Lossless Transmission Lines 27
2. Lossy Transmission Lines 27
3. Lossy Transmission Line Model (LTRA) 28
4. Uniform Distributed RC Lines (Lossy) 29
5. Uniform Distributed RC Model (URC) 30
4. TRANSISTORS AND DIODES 31
1. Junction Diodes 31
2. Diode Model (D) 32
3. Bipolar Junction Transistors (BJTs) 33
4. BJT Models (NPN/PNP) 33
5. Junction Field-Effect Transistors (JFETs) 35
6. JFET Models (NJF/PJF) 36
7. MOSFETs 37
8. MOSFET Models (NMOS/PMOS) 38
9. MESFETs 43
10. MESFET Models (NMF/PMF) 43
4. ANALYSES AND OUTPUT CONTROL 45
1. SIMULATOR VARIABLES (.OPTIONS) 45
2. INITIAL CONDITIONS 49
1. .NODESET: Specify Initial Node Voltage Guesses 49
2. .IC: Set Initial Conditions 49
3. ANALYSES 50
1. .AC: Small-Signal AC Analysis 50
2. .DC: DC Transfer Function 50
3. .DISTO: Distortion Analysis 51
4. .NOISE: Noise Analysis 52
5. .OP: Operating Point Analysis 53
6. .PZ: Pole-Zero Analysis 53
7. .SENS: DC or Small-Signal AC Sensitivity Analysis 54
8. .TF: Transfer Function Analysis 54
9. .TRAN: Transient Analysis 55

4. BATCH OUTPUT 56
1. .SAVE Lines 56
2. .PRINT Lines 56
3. .PLOT Lines 57
4. .FOUR: Fourier Analysis of Transient Analysis Output 58
5. INTERACTIVE INTERPRETER 59
1. EXPRESSIONS, FUNCTIONS, AND CONSTANTS 60
2. COMMAND INTERPRETATION 64
3. COMMANDS 65
1. Ac*: Perform an AC, small-signal frequency response analysis 65
2. Alias: Create an alias for a command 65
3. Alter*: Change a device or model parameter 65
4. Asciiplot: Plot values using old-style character plots 66
5. Aspice: Asynchronous spice run 66
6. Bug: Mail a bug report 66
7. Cd: Change directory 66
8. Destroy: Delete a data set 67
9. Dc*: Perform a DC-sweep analysis 67
10. Define: Define a function 67
11. Delete*: Remove a trace or breakpoint 68
12. Diff: Compare vectors 68
13. Display: List known vectors and types 68
14. Echo: Print text 68
15. Edit*: Edit the current circuit 69
16. Fourier: Perform a fourier transform 69
17. Hardcopy: Save a plot to a file for printing 69
18. Help: Print summaries of Spice3 commands 70
19. History: Review previous commands 70
20. Iplot*: Incremental plot 70
21. Jobs: List active asynchronous spice runs 70
22. Let: Assign a value to a vector 70
23. Linearize*: Interpolate to a linear scale 71
24. Listing*: Print a listing of the current circuit 71
25. Load: Load rawfile data 71
26. Op*: Perform an operating point analysis 72
27. Plot: Plot values on the display 72
28. Print: Print values 72
29. Quit: Leave Spice3 or Nutmeg 73
30. Rehash: Reset internal hash tables 73
31. Reset*: Reset an analysis 73
32. Reshape: Alter the dimensionality or dimensions of a vector 74
33. Resume*: Continue a simulation after a stop 74

34. Rspice: Remote spice submission 74
35. Run*: Run analysis from the input file 75
36. Rusage: Resource usage 76
37. Save*: Save a set of outputs 77
38. Sens*: Run a sensitivity analysis 77
39. Set: Set the value of a variable 77
40. Setcirc*: Change the current circuit 78
41. Setplot: Switch the current set of vectors 78
42. Settype: Set the type of a vector 78
43. Shell: Call the command interpreter 78
44. Shift: Alter a list variable 79
45. Show*: List device state 79
46. Showmod*: List model parameter values 80
47. Source: Read a Spice3 input file 80
48. Status*: Display breakpoint information 80
49. Step*: Run a fixed number of timepoints 81
50. Stop*: Set a breakpoint 81
51. Tf*: Run a Transfer Function analysis 81
52. Trace*: Trace nodes 82
53. Tran*: Perform a transient analysis 82
54. Transpose: Swap the elements in a multi-dimensional data set 82
55. Unalias: Retract an alias 83
56. Undefine: Retract a definition 83
57. Unset: Clear a variable 83
58. Version: Print the version of Spice 83
59. Where: Identify troublesome node or device 84
60. Write: Write data to a file 84
61. Xgraph: use the xgraph(1) program for plotting. 84
4. CONTROL STRUCTURES 85
1. While — End 85
2. Repeat — End 85
3. Dowhile — End 85
4. Foreach — End 86
5. If — Then — Else 86
6. Label 86
7. Goto 87
8. Continue 87
9. Break 87
5. VARIABLES 88
6. MISCELLANEOUS 92
7. BUGS 94
6. BIBLIOGRAPHY 97

A. APPENDIX A: EXAMPLE CIRCUITS 99
1. Circuit 1: Differential Pair 99
2. Circuit 2: MOSFET Characterization 99
3. Circuit 3: RTL Inverter 100
4. Circuit 4: Four-Bit Binary Adder 100
5. Circuit 5: Transmission-Line Inverter 102
B. APPENDIX B: MODEL AND DEVICE PARAMETERS 103
1. URC: Uniform R.C. line 104
2. ASRC: Arbitrary Source 104
3. BJT: Bipolar Junction Transistor 105
4. BSIM1: Berkeley Short Channel IGFET Model 108
5. BSIM2: Berkeley Short Channel IGFET Model 110
6. Capacitor: Fixed capacitor 114
7. CCCS: Current controlled current source 115
8. CCVS: Linear current controlled current source 115
9. CSwitch: Current controlled ideal switch 115
10. Diode: Junction Diode model 116
11. Inductor: Inductors 117
12. mutual: Mutual inductors 118
13. Isource: Independent current source 118
14. JFET: Junction Field effect transistor 119
15. LTRA: Lossy transmission line 120
16. MES: GaAs MESFET model 121
17. Mos1: Level 1 MOSfet model with Meyer capacitance model 123
18. Mos2: Level 2 MOSfet model with Meyer capacitance model 126
19. Mos3: Level 3 MOSfet model with Meyer capacitance model 129
20. Mos6: Level 6 MOSfet model with Meyer capacitance model 132
21. Resistor: Simple linear resistor 135
22. Switch: Ideal voltage controlled switch 135
23. Tranline: Lossless transmission line 136
24. VCCS: Voltage controlled current source 137
25. VCVS: Voltage controlled voltage source 137
26. Vsource: Independent voltage source 138

RELEASE NOTES

Spice3f is the six major release of Spice3. This release incorporates new features not available in Spice 3c or 3d, as well as several performance improvements. All of the features described here are believed to be fully functional. The development of SPICE and its algorithms is ongoing at Berkeley, and therefore not all of the intended capabilities of the program have been implemented in full yet.

Bugs in 3f2 fixed in 3f3:

```
Ascii (printer) plots in spice3f2 did not print bode plots
vs log of the frequency by default, as in spice2.
You had to explicity request the x-axis to be log; either
"plot vdb(2) xlog" (best) or "plot vdb(2) vs log10(frequency)" will do.
Now simply "plot vdb(2)" will work.

The on-line documentation has been brought up to date by converting this
manual into a format readable on-line.

Significant problems with AC sensitivities in 3f2 only have been fixed.

Multiple analyses and plots in spice2 emulation mode under 3f2 and earlier
generated misleading error messages.
This no longer happens in 3f3.
```

New features and bug fixes incorporated in Spice3f (the current release of Spice3):

```
Sensitivity analysis.
Added a parameter for fitting JFET models (parameter "B").
Fixed a discontinuity problem in MOS level 3
      (related to the "kappa" parameter).
Working "alter" command.
Improved "show" and "showmod" commands for operating
      point summary tables (like Spice2).
Working "trace" command.
Interactive "set" variable values now the same as ".options" settings.
Improved plotting, including implicitly transforming data for smith plots.
Added the function "deriv( )" (derivative) to the front-end.
Corrected an error affecting the specified initial conditions for
      some circuits.
Small signal behaviour of non-linear dependant sources fixed.
Miscellaneous bug fixes in the front end.
```

New features and bug fixes incorporated in Spice3e (the *previous* release of Spice3):

```
Lossy Transmission Lines.
Proper calculation of sheet resistance in the MOS models.
A new command ("where") to aid in debugging troublesome circuits.
Smith-chart plots working (see the "plot" command).
Arbitrary sources in subcircuits handled correctly.
Arbitrary source reciprocal calculations and DC biasing fixed.
Minor bug-fixes to the Pole-Zero analysis.
Miscellaneous bug fixes in the front end.
```

Some common problems remaining in Spice3f (note that this list is not complete):

```
Models defined within subcircuits are not always handled correctly.
If you have trouble, move the model definition outside of ".subckt"
and ".ends" lines.
```

```
Batch run data is not compacted if a "rawspice" data file is generated,
resulting in excessively large output files for some difficult inputs.

Sufficient detail is sometimes not preserved in transient analysis.
Providing a small value for the "TMAX" parameter (the fourth argument)
in the transient run command will solve this problem.

Convergence problems can sometimes be worked around by relaxing the
transient "TMAX" parameter.

The substrate node of the bipolar transistor (BJT) is modelled
incorrectly (this may actually be due to inherant numerical
problems with the model).  Do not use the substrate node; use
a semiconductor capacitor to model substrate effects.

Charge is not conserved in MOS devices based on the Meyer model.

Transient simulation of strictly resistive circuits (typical for
first runs or tests) allow a time step that is too large (e.g. a
sinusoidal source driving a resistor).  There is no integration error
to restrict the time step.  Use the "TMAX" parameter or include reactive
elements.

Deep nesting of subcircuits may exceed internal static buffers.

The PZ analysis can not be interrupted;  the sensitivity analysis
can not be continued (the interactive "resume" command) once interrupted.

There are many other small bugs, particularly in the front end.
```

Bug reporting:

Please send bug reports or suggested enhancements to Spice3 via electronic mail to:

```
spice@Berkeley.EDU  (Internet)
ucbvax!spice  (UUCP-net)
```

or by physical mail to "Spice3 Bug Report" at:

```
EECS/ERL Industrial Support Office
205 Cory Hall
U.C. Berkeley
Berkeley, CA   94720
```

Information on ordering Spice3 or other software distributed through the EECS/ERL Industrial Support Office should be directed to the above physical mail address or to "ilpsoftware@berkeley.edu" via electronic mail.

Include the version of Spice used, the system type and operating system version, the smallest input file demonstrating the problem, and the resulting (incorrect) output.

1. INTRODUCTION

SPICE is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analyses. Circuits may contain resistors, capacitors, inductors, mutual inductors, independent voltage and current sources, four types of dependent sources, lossless and lossy transmission lines (two separate implementations), switches, uniform distributed RC lines, and the five most common semiconductor devices: diodes, BJTs, JFETs, MESFETs, and MOSFETs.

The SPICE3 version is based directly on SPICE 2G.6. While SPICE3 is being developed to include new features, it continues to support those capabilities and models which remain in extensive use in the SPICE2 program.

SPICE has built-in models for the semiconductor devices, and the user need specify only the pertinent model parameter values. The model for the BJT is based on the integral-charge model of Gummel and Poon; however, if the Gummel- Poon parameters are not specified, the model reduces to the simpler Ebers-Moll model. In either case, charge-storage effects, ohmic resistances, and a current-dependent output conductance may be included. The diode model can be used for either junction diodes or Schottky barrier diodes. The JFET model is based on the FET model of Shichman and Hodges. Six MOSFET models are implemented: MOS1 is described by a square-law I-V characteristic, MOS2 [1] is an analytical model, while MOS3 [1] is a semi-empirical model; MOS6 [2] is a simple analytic model accurate in the short-channel region; MOS4 [3, 4] and MOS5 [5] are the BSIM (Berkeley Short-channel IGFET Model) and BSIM2. MOS2, MOS3, and MOS4 include second-order effects such as channel-length modulation, subthreshold conduction, scattering-limited velocity saturation, small-size effects, and charge-controlled capacitances.

1.1. TYPES OF ANALYSIS

1.1.1. DC Analysis

The dc analysis portion of SPICE determines the dc operating point of the circuit with inductors shorted and capacitors opened. The dc analysis options are specified on the .DC, .TF, and .OP control lines. A dc analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an ac small-signal analysis to determine the linearized, small-signal models for nonlinear devices. If requested, the dc small-signal value of a transfer function (ratio of output variable to input source), input resistance, and output resistance is also computed as a part of the dc solution. The dc analysis can also be used to generate dc transfer curves: a specified independent voltage or current source is stepped over a user-specified range and the dc output variables are stored for each sequential source value.

1.1.2. AC Small-Signal Analysis

The ac small-signal portion of SPICE computes the ac output variables as a function of frequency. The program first computes the dc operating point of the circuit and determines linearized, small-signal models for all of the nonlinear devices in the circuit. The resultant linear circuit is then analyzed over a user-specified range of frequencies. The desired output of an ac small- signal analysis is usually a transfer function (voltage gain, transimpedance, etc). If the circuit has only one ac input, it is convenient to set that input to unity and zero phase, so that output variables have the same value as the transfer function of the output variable with respect to the input.

1.1.3. Transient Analysis

The transient analysis portion of SPICE computes the transient output variables as a function of time over a user-specified time interval. The initial conditions are automatically determined by a dc analysis. All sources which are not time dependent (for example, power supplies) are set to their dc value. The transient time interval is specified on a .TRAN control line.

1.1.4. Pole-Zero Analysis

The pole-zero analysis portion of SPICE computes the poles and/or zeros in the small-signal ac transfer function. The program first computes the dc operating point and then determines the linearized, small-signal models for all the nonlinear devices in the circuit. This circuit is then used to find the poles and zeros of the transfer function.

Two types of transfer functions are allowed : one of the form (output voltage)/(input voltage) and the other of the form (output voltage)/(input current). These two types of transfer functions cover all the cases and one can find the poles/zeros of functions like input/output impedance and voltage gain. The input and output ports are specified as two pairs of nodes.

The pole-zero analysis works with resistors, capacitors, inductors, linear-controlled sources, independent sources, BJTs, MOSFETs, JFETs and diodes. Transmission lines are not supported.

The method used in the analysis is a sub-optimal numerical search. For large circuits it may take a considerable time or fail to find all poles and zeros. For some circuits, the method becomes "lost" and finds an excessive number of poles or zeros.

1.1.5. Small-Signal Distortion Analysis

The distortion analysis portion of SPICE computes steady-state harmonic and intermodulation products for small input signal magnitudes. If signals of a single frequency are specified as the input to the

circuit, the complex values of the second and third harmonics are determined at every point in the circuit. If there are signals of two frequencies input to the circuit, the analysis finds out the complex values of the circuit variables at the sum and difference of the input frequencies, and at the difference of the smaller frequency from the second harmonic of the larger frequency.

Distortion analysis is supported for the following nonlinear devices: diodes (DIO), BJT, JFET, MOSFETs (levels 1, 2, 3, 4/BSIM1, 5/BSIM2, and 6) and MESFETS. All linear devices are automatically supported by distortion analysis. If there are switches present in the circuit, the analysis continues to be accurate provided the switches do not change state under the small excitations used for distortion calculations.

1.1.6. Sensitivity Analysis

Spice3 will calculate either the DC operating-point sensitivity or the AC small-signal sensitivity of an output variable with respect to all circuit variables, including model parameters. Spice calculates the difference in an output variable (either a node voltage or a branch current) by perturbing each parameter of each device independently. Since the method is a numerical approximation, the results may demonstrate second order affects in highly sensitive parameters, or may fail to show very low but non-zero sensitivity. Further, since each variable is perturb by a small fraction of its value, zero-valued parameters are not analyized (this has the benefit of reducing what is usually a very large amount of data).

1.1.7. Noise Analysis

The noise analysis portion of SPICE does analysis device-generated noise for the given circuit. When provided with an input source and an output port, the analysis calculates the noise contributions of each device (and each noise generator within the device) to the output port voltage. It also calculates the input noise to the circuit, equivalent to the output noise referred to the specified input source. This is done for every frequency point in a specified range - the calculated value of the noise corresponds to the spectral density of the circuit variable viewed as a stationary gaussian stochastic process.

After calculating the spectral densities, noise analysis integrates these values over the specified frequency range to arrive at the total noise voltage/current (over this frequency range). This calculated value corresponds to the variance of the circuit variable viewed as a stationary gaussian process.

1.2. ANALYSIS AT DIFFERENT TEMPERATURES

All input data for SPICE is assumed to have been measured at a nominal temperature of 27°C, which can be changed by use of the TNOM parameter on the .OPTION control line. This value can further be overridden for any device which models temperature effects by specifying the TNOM parameter on the model itself. The circuit simulation is performed at a temperature of 27°C, unless overridden by a TEMP parameter on the .OPTION control line. Individual instances may further override the circuit temperature through the specification of a TEMP parameter on the instance.

Temperature dependent support is provided for resistors, diodes, JFETs, BJTs, and level 1, 2, and 3 MOSFETs. BSIM (levels 4 and 5) MOSFETs have an alternate temperature dependency scheme which adjusts all of the model parameters before input to SPICE. For details of the BSIM temperature adjustment, see [6] and [7].

Temperature appears explicitly in the exponential terms of the BJT and diode model equations. In addition, saturation currents have a built-in temperature dependence. The temperature dependence of the saturation current in the BJT models is determined by:

$$I_S(T_1) = I_S(T_0)\left(\frac{T_1}{T_0}\right)^{XTI} \exp\left(\frac{E_g q(T_1\ T_0)}{k\ (T_1 - T_0)}\right)$$

where k is Boltzmann's constant, q is the electronic charge, E_G is the energy gap which is a model parameter, and XTI is the saturation current temperature exponent (also a model parameter, and usually equal to 3).

The temperature dependence of forward and reverse beta is according to the formula:

$$\beta(T_1) = \beta(T_0)\left(\frac{T_1}{T_0}\right)^{XTB}$$

where T_1 and T_0 are in degrees Kelvin, and XTB is a user-supplied model parameter. Temperature effects on beta are carried out by appropriate adjustment to the values of β_F, I_{SE}, β_R, and I_{SC} (spice model parameters BF, ISE, BR, and ISC, respectively).

Temperature dependence of the saturation current in the junction diode model is determined by:

$$I_S(T_1) = I_S(T_0)\left(\frac{T_1}{T_0}\right)^{\frac{XTI}{N}} \exp\left(\frac{E_g q(T_1\ T_0)}{N\ k\ (T_1 - T_0)}\right)$$

where N is the emission coefficient, which is a model parameter, and the other symbols have the same meaning as above. Note that for Schottky barrier diodes, the value of the saturation current temperature exponent, XTI, is usually 2.

Temperature appears explicitly in the value of junction potential, ϕ (in spice PHI), for all the device models. The temperature dependence is determined by:

$$\phi(T) = \frac{kT}{q}\log_e\left(\frac{N_a N_d}{N_i(T)^2}\right)$$

where k is Boltzmann's constant, q is the electronic charge, N_a is the acceptor impurity density, N_d is the donor impurity density, N_i is the intrinsic carrier concentration, and E_g is the energy gap.

Temperature appears explicitly in the value of surface mobility, μ_0 (or UO), for the MOSFET model. The temperature dependence is determined by:

$$\mu_0(T) = \frac{\mu_0(T_0)}{\left(\frac{T}{T_0}\right)^{1.5}}$$

The effects of temperature on resistors is modeled by the formula:

$$R(T) = R(T_0)\,[1 + TC_1\,(T - T_0) + TC_2\,(T - T_0)^2]$$

where T is the circuit temperature, T_0 is the nominal temperature, and TC_1 and TC_2 are the first- and second-order temperature coefficients.

1.3. CONVERGENCE

Both dc and transient solutions are obtained by an iterative process which is terminated when both of the following conditions hold:

1) The nonlinear branch currents converge to within a tolerance of 0.1% or 1 picoamp (1.0e-12 Amp), whichever is larger.

2) The node voltages converge to within a tolerance of 0.1% or 1 microvolt (1.0e-6 Volt), whichever is larger.

Although the algorithm used in SPICE has been found to be very reliable, in some cases it fails to converge to a solution. When this failure occurs, the program terminates the job.

Failure to converge in dc analysis is usually due to an error in specifying circuit connections, element values, or model parameter values. Regenerative switching circuits or circuits with positive feedback probably will not converge in the dc analysis unless the OFF option is used for some of the devices in the feedback path, or the .NODESET control line is used to force the circuit to converge to the desired state.

2. CIRCUIT DESCRIPTION

2.1. GENERAL STRUCTURE AND CONVENTIONS

The circuit to be analyzed is described to SPICE by a set of element lines, which define the circuit topology and element values, and a set of control lines, which define the model parameters and the run controls. The first line in the input file must be the title, and the last line must be ".END". The order of the remaining lines is arbitrary (except, of course, that continuation lines must immediately follow the line being continued).

Each element in the circuit is specified by an element line that contains the element name, the circuit nodes to which the element is connected, and the values of the parameters that determine the electrical characteristics of the element. The first letter of the element name specifies the element type. The format for the SPICE element types is given in what follows. The strings XXXXXXX, YYYYYYY, and ZZZZZZZ denote arbitrary alphanumeric strings. For example, a resistor name must begin with the letter R and can contain one or more characters. Hence, R, R1, RSE, ROUT, and R3AC2ZY are valid resistor names. Details of each type of device are supplied in a following section.

Fields on a line are separated by one or more blanks, a comma, an equal ('=') sign, or a left or right parenthesis; extra spaces are ignored. A line may be continued by entering a '+' (plus) in column 1 of the following line; SPICE continues reading beginning with column 2.

A name field must begin with a letter (A through Z) and cannot contain any delimiters.

A number field may be an integer field (12, -44), a floating point field (3.14159), either an integer or floating point number followed by an integer exponent (1e-14, 2.65e3), or either an integer or a floating point number followed by one of the following scale factors:

$T = 10^{12}$	$G = 10^{9}$	$Meg = 10^{6}$	$K = 10^{3}$	$mil = 25.4^{-6}$
$m = 10^{-3}$	u (or μ) $= 10^{-6}$	$n = 10^{-9}$	$p = 10^{-12}$	$f = 10^{-15}$

Letters immediately following a number that are not scale factors are ignored, and letters immediately following a scale factor are ignored. Hence, 10, 10V, 10Volts, and 10Hz all represent the same number, and M, MA, MSec, and MMhos all represent the same scale factor. Note that 1000, 1000.0, 1000Hz, 1e3, 1.0e3, 1KHz, and 1K all represent the same number.

Nodes names may be arbitrary character strings. The datum (ground) node must be named '0'. Note the difference in SPICE3 where the nodes are treated as character strings and not evaluated as numbers, thus '0' and '00' are distinct nodes in SPICE3 but not in SPICE2. The circuit cannot contain a loop of voltage sources and/or inductors and cannot contain a cut-set of current sources and/or capacitors. Each node in the circuit must have a dc path to ground. Every node must have at least two connections except for transmission line nodes (to permit unterminated transmission lines) and MOSFET substrate nodes (which have two internal connections anyway).

2.2. TITLE LINE, COMMENT LINES AND .END LINE

2.2.1. Title Line

Examples:

```
POWER AMPLIFIER CIRCUIT
TEST OF CAM CELL
```

The title line must be the first in the input file. Its contents are printed verbatim as the heading for each section of output.

2.2.2. .END Line

Examples:

```
.END
```

The "End" line must always be the last in the input file. Note that the period is an integral part of the name.

2.2.3. Comments

General Form:

```
* <any comment>
```

Examples:

```
* RF=1K      Gain should be 100
* Check open-loop gain and phase margin
```

The asterisk in the first column indicates that this line is a comment line. Comment lines may be placed anywhere in the circuit description. Note that SPICE3 also considers any line with leading white space to be a comment.

2.3. DEVICE MODELS

General form:

```
.MODEL MNAME TYPE(PNAME1=PVAL1 PNAME2=PVAL2 ... )
```

Examples:

```
.MODEL MOD1 NPN (BF=50 IS=1E-13 VBF=50)
```

Most simple circuit elements typically require only a few parameter values. However, some devices (semiconductor devices in particular) that are included in SPICE require many parameter values. Often, many devices in a circuit are defined by the same set of device model parameters. For these reasons, a set of device model parameters is defined on a separate .MODEL line and assigned a unique model name. The device element lines in SPICE then refer to the model name.

For these more complex device types, each device element line contains the device name, the nodes to which the device is connected, and the device model name. In addition, other optional parameters may be specified for some devices: geometric factors and an initial condition (see the following section on Transistors and Diodes for more details).

MNAME in the above is the model name, and type is one of the following fifteen types:

R	Semiconductor resistor model
C	Semiconductor capacitor model
SW	Voltage controlled switch
CSW	Current controlled switch
URC	Uniform distributed RC model
LTRA	Lossy transmission line model
D	Diode model
NPN	NPN BJT model
PNP	PNP BJT model
NJF	N-channel JFET model
PJF	P-channel JFET model
NMOS	N-channel MOSFET model
PMOS	P-channel MOSFET model
NMF	N-channel MESFET model
PMF	P-channel MESFET model

Parameter values are defined by appending the parameter name followed by an equal sign and the parameter value. Model parameters that are not given a value are assigned the default values given below for each model type. Models, model parameters, and default values are listed in the next section along with the description of device element lines.

2.4. SUBCIRCUITS

A subcircuit that consists of SPICE elements can be defined and referenced in a fashion similar to device models. The subcircuit is defined in the input file by a grouping of element lines; the program then automatically inserts the group of elements wherever the subcircuit is referenced. There is no limit on the size or complexity of subcircuits, and subcircuits may contain other subcircuits. An example of subcircuit usage is given in Appendix A.

2.4.1. .SUBCKT Line

General form:

```
.SUBCKT subnam N1 <N2 N3 ...>
```

Examples:

```
.SUBCKT OPAMP 1 2 3 4
```

A circuit definition is begun with a .SUBCKT line. SUBNAM is the subcircuit name, and N1, N2, ... are the external nodes, which cannot be zero. The group of element lines which immediately follow the .SUBCKT line define the subcircuit. The last line in a subcircuit definition is the .ENDS line (see below). Control lines may not appear within a subcircuit definition; however, subcircuit definitions may contain anything else, including other subcircuit definitions, device models, and subcircuit calls (see below). Note that any device models or subcircuit definitions included as part of a subcircuit definition are strictly local (i.e., such models and definitions are not known outside the subcircuit definition). Also, any element nodes not included on the .SUBCKT line are strictly local, with the exception of 0 (ground) which is always global.

2.4.2. .ENDS Line

General form:

```
.ENDS <SUBNAM>
```

Examples:

```
.ENDS OPAMP
```

The "Ends" line must be the last one for any subcircuit definition. The subcircuit name, if included, indicates which subcircuit definition is being terminated; if omitted, all subcircuits being defined are terminated. The name is needed only when nested subcircuit definitions are being made.

2.4.3. Subcircuit Calls

General form:

```
XYYYYYYY N1 <N2 N3 ...> SUBNAM
```

Examples:

```
X1 2 4 17 3 1 MULTI
```

Subcircuits are used in SPICE by specifying pseudo-elements beginning with the letter X, followed by the circuit nodes to be used in expanding the subcircuit.

2.5. COMBINING FILES: .INCLUDE LINES

General form:

`.INCLUDE` *filename*

Examples:

```
.INCLUDE /users/spice/common/wattmeter.cir
```

Frequently, portions of circuit descriptions will be reused in several input files, particularly with common models and subcircuits. In any spice input file, the ".include" line may be used to copy some other file as if that second file appeared in place of the ".include" line in the original file. There is no restriction on the file name imposed by spice beyond those imposed by the local operating system.

3. CIRCUIT ELEMENTS AND MODELS

Data fields that are enclosed in less-than and greater-than signs ('< >') are optional. All indicated punctuation (parentheses, equal signs, etc.) is optional but indicate the presence of any delimiter. Further, future implementations may require the punctuation as stated. A consistent style adhering to the punctuation shown here makes the input easier to understand. With respect to branch voltages and currents, SPICE uniformly uses the associated reference convention (current flows in the direction of voltage drop).

3.1. ELEMENTARY DEVICES

3.1.1. Resistors

General form:

```
RXXXXXXX N1 N2 VALUE
```

Examples:

```
R1 1 2 100
RC1 12 17 1K
```

N1 and N2 are the two element nodes. VALUE is the resistance (in ohms) and may be positive or negative but not zero.

3.1.2. Semiconductor Resistors

General form:

```
RXXXXXXX N1 N2 <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <TEMP=T>
```

Examples:

```
RLOAD 2 10 10K
RMOD 3 7 RMODEL L=10u W=1u
```

This is the more general form of the resistor presented in section 6.1, and allows the modeling of temperature effects and for the calculation of the actual resistance value from strictly geometric information and the specifications of the process. If VALUE is specified, it overrides the geometric information and defines the resistance. If MNAME is specified, then the resistance may be calculated from the process information in the model MNAME and the given LENGTH and WIDTH. If VALUE is not specified, then MNAME and LENGTH **must** be specified. If WIDTH is not specified, then it is taken from the default

width given in the model. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

3.1.3. Semiconductor Resistor Model (R)

The resistor model consists of process-related device data that allow the resistance to be calculated from geometric information and to be corrected for temperature. The parameters available are:

name	parameter	units	default	example
TC1	first order temperature coeff.	Ω/°C	0.0	-
TC2	second order temperature coeff.	$\Omega/°C^2$	0.0	-
RSH	sheet resistance	Ω/□	-	50
DEFW	default width	meters	1e-6	2e-6
NARROW	narrowing due to side etching	meters	0.0	1e-7
TNOM	parameter measurement temperature	°C	27	50

The sheet resistance is used with the narrowing parameter and L and W from the resistor device to determine the nominal resistance by the formula

$$R = RSH\frac{L - NARROW}{W - NARROW}$$

DEFW is used to supply a default value for W if one is not specified for the device. If either RSH or L is not specified, then the standard default resistance value of 1k Ω is used. TNOM is used to override the circuit-wide value given on the .OPTIONS control line where the parameters of this model have been measured at a different temperature. After the nominal resistance is calculated, it is adjusted for temperature by the formula:

$$R(T) = R(T_0)\,[1 + TC1\,(T - T_0) + TC2\,(T - T_0)^2]$$

3.1.4. Capacitors

General form:

```
CXXXXXXX N+ N- VALUE <IC=INCOND>
```

Examples:

```
CBYP 13 0 1UF
COSC 17 23 10U IC=3V
```

N+ and N- are the positive and negative element nodes, respectively. VALUE is the capacitance in Farads.

The (optional) initial condition is the initial (time-zero) value of capacitor voltage (in Volts). Note that the initial conditions (if any) apply 'only' if the UIC option is specified on the .TRAN control line.

3.1.5. Semiconductor Capacitors

General form:

```
CXXXXXXX N1 N2 <VALUE> <MNAME> <L=LENGTH> <W=WIDTH> <IC=VAL>
```

Examples:

```
CLOAD 2 10 10P
CMOD 3 7 CMODEL L=10u W=1u
```

This is the more general form of the Capacitor presented in section 6.2, and allows for the calculation of the actual capacitance value from strictly geometric information and the specifications of the process. If VALUE is specified, it defines the capacitance. If MNAME is specified, then the capacitance is calculated from the process information in the model MNAME and the given LENGTH and WIDTH. If VALUE is not specified, then MNAME and LENGTH **must** be specified. If WIDTH is not specified, then it is taken from the default width given in the model. Either VALUE or MNAME, LENGTH, and WIDTH may be specified, but not both sets.

3.1.6. Semiconductor Capacitor Model (C)

The capacitor model contains process information that may be used to compute the capacitance from strictly geometric information.

name	parameter	units	default	example
CJ	junction bottom capacitance	F/meters^2	-	5e-5
CJSW	junction sidewall capacitance	F/meters	-	2e-11
DEFW	default device width	meters	1e-6	2e-6
NARROW	narrowing due to side etching	meters	0.0	1e-7

The capacitor has a capacitance computed as

CAP = CJ (LENGTH − NARROW) (WIDTH − NARROW) + 2 CJSW (LENGTH + WIDTH − 2 NARROW)

3.1.7. Inductors

General form:

```
LYYYYYYY N+ N- VALUE <IC=INCOND>
```

Examples:

```
LLINK 42 69 1UH
LSHUNT 23 51 10U IC=15.7MA
```

N+ and N- are the positive and negative element nodes, respectively. VALUE is the inductance in Henries.

The (optional) initial condition is the initial (time-zero) value of inductor current (in Amps) that flows from N+, through the inductor, to N-. Note that the initial conditions (if any) apply only if the UIC option is specified on the .TRAN analysis line.

3.1.8. Coupled (Mutual) Inductors

General form:

```
KXXXXXXX LYYYYYYY LZZZZZZZ VALUE
```

Examples:

```
K43 LAA LBB 0.999
KXFRMR L1 L2 0.87
```

LYYYYYYY and LZZZZZZZ are the names of the two coupled inductors, and VALUE is the coefficient of coupling, K, which must be greater than 0 and less than or equal to 1. Using the 'dot' convention, place a 'dot' on the first node of each inductor.

3.1.9. Switches

General form:

```
SXXXXXXX N+ N- NC+ NC- MODEL <ON><OFF>
WYYYYYYY N+ N- VNAM MODEL <ON><OFF>
```

Examples:

```
s1 1 2 3 4 switch1 ON
s2 5 6 3 0 sm2 off
Switch1 1 2 10 0 smodel1
w1 1 2 vclock switchmod1
W2 3 0 vramp sm1 ON
wreset 5 6 vclck lossyswitch OFF
```

Nodes 1 and 2 are the nodes between which the switch terminals are connected. The model name is mandatory while the initial conditions are optional. For the voltage controlled switch, nodes 3 and 4 are the positive and negative controlling nodes respectively. For the current controlled switch, the controlling current is that through the specified voltage source. The direction of positive controlling current flow is from the positive node, through the source, to the negative node.

3.1.10. Switch Model (SW/CSW)

The switch model allows an almost ideal switch to be described in SPICE. The switch is not quite ideal, in that the resistance can not change from 0 to infinity, but must always have a finite positive value. By proper selection of the on and off resistances, they can be effectively zero and infinity in comparison to other circuit elements. The parameters available are:

name	parameter	units	default	switch
VT	threshold voltage	Volts	0.0	S
IT	threshold current	Amps	0.0	W
VH	hysteresis voltage	Volts	0.0	S
IH	hysteresis current	Amps	0.0	W
RON	on resistance	Ω	1.0	both
ROFF	off resistance	Ω	1/GMIN*	both

*(See the .OPTIONS control line for a description of GMIN, its default value results in an off-resistance of 1.0e+12 ohms.)

The use of an ideal element that is highly nonlinear such as a switch can cause large discontinuities to occur in the circuit node voltages. A rapid change such as that associated with a switch changing state can cause numerical roundoff or tolerance problems leading to erroneous results or timestep difficulties. The user of switches can improve the situation by taking the following steps:

First, it is wise to set ideal switch impedances just high or low enough to be negligible with respect to other circuit elements. Using switch impedances that are close to "ideal" in all cases aggravates the problem of discontinuities mentioned above. Of course, when modeling real devices such as MOSFETS, the on resistance should be adjusted to a realistic level depending on the size of the device being modeled.

If a wide range of ON to OFF resistance must be used in the switches (ROFF/RON >1e+12), then the tolerance on errors allowed during transient analysis should be decreased by using the .OPTIONS control line and specifying TRTOL to be less than the default value of 7.0. When switches are placed around capacitors, then the option CHGTOL should also be reduced. Suggested values for these two options are 1.0 and 1e-16 respectively. These changes inform SPICE3 to be more careful around the switch points so that no errors are made due to the rapid change in the circuit.

3.2. VOLTAGE AND CURRENT SOURCES

3.2.1. Independent Sources

General form:

```
VXXXXXXX N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>
+     <DISTOF1 <F1MAG <F1PHASE>>> <DISTOF2 <F2MAG <F2PHASE>>>
IYYYYYYY N+ N- <<DC> DC/TRAN VALUE> <AC <ACMAG <ACPHASE>>>
+     <DISTOF1 <F1MAG <F1PHASE>>> <DISTOF2 <F2MAG <F2PHASE>>>
```

Examples:

```
VCC 10 0 DC 6
VIN 13 2 0.001 AC 1 SIN(0 1 1MEG)
ISRC 23 21 AC 0.333 45.0 SFFM(0 1 10K 5 1K)
VMEAS 12 9
VCARRIER 1 0 DISTOF1 0.1 -90.0
VMODULATOR 2 0 DISTOF2 0.01
IIN1 1 5 AC 1 DISTOF1 DISTOF2 0.001
```

N+ and N- are the positive and negative nodes, respectively. Note that voltage sources need not be grounded. Positive current is assumed to flow from the positive node, through the source, to the negative node. A current source of positive value forces current to flow out of the N+ node, through the source, and into the N- node. Voltage sources, in addition to being used for circuit excitation, are the 'ammeters' for SPICE, that is, zero valued voltage sources may be inserted into the circuit for the purpose of measuring current. They of course have no effect on circuit operation since they represent short-circuits.

DC/TRAN is the dc and transient analysis value of the source. If the source value is zero both for dc and transient analyses, this value may be omitted. If the source value is time-invariant (e.g., a power supply), then the value may optionally be preceded by the letters DC.

ACMAG is the ac magnitude and ACPHASE is the ac phase. The source is set to this value in the ac analysis. If ACMAG is omitted following the keyword AC, a value of unity is assumed. If ACPHASE is omitted, a value of zero is assumed. If the source is not an ac small-signal input, the keyword AC and the ac values are omitted.

DISTOF1 and DISTOF2 are the keywords that specify that the independent source has distortion inputs at the frequencies F1 and F2 respectively (see the description of the .DISTO control line). The keywords may be followed by an optional magnitude and phase. The default values of the magnitude and phase are 1.0 and 0.0 respectively.

Any independent source can be assigned a time-dependent value for transient analysis. If a source is assigned a time-dependent value, the time-zero value is used for dc analysis. There are five independent source functions: pulse, exponential, sinusoidal, piece-wise linear, and single-frequency FM. If parameters other than source values are omitted or set to zero, the default values shown are assumed. (TSTEP is

the printing increment and TSTOP is the final time (see the .TRAN control line for explanation)).

3.2.1.1. Pulse

General form:

```
PULSE(V1 V2 TD TR TF PW PER)
```

Examples:

```
VIN 3 0 PULSE(-1 1 2NS 2NS 2NS 50NS 100NS)
```

parameter	default value	units
V1 (initial value)		Volts or Amps
V2 (pulsed value)		Volts or Amps
TD (delay time)	0.0	seconds
TR (rise time)	TSTEP	seconds
TF (fall time)	TSTEP	seconds
PW (pulse width)	TSTOP	seconds
PER(period)	TSTOP	seconds

A single pulse so specified is described by the following table:

time	value
0	V1
TD	V1
TD+TR	V2
TD+TR+PW	V2
TD+TR+PW+TF	V1
TSTOP	V1

Intermediate points are determined by linear interpolation.

3.2.1.2. Sinusoidal

General form:

```
SIN(VO VA FREQ TD THETA)
```

Examples:

```
VIN 3 0 SIN(0 1 100MEG 1NS 1E10)
```

parameters	default value	units
VO (offset)		Volts or Amps
VA (amplitude)		Volts or Amps
FREQ (frequency)	1/TSTOP	Hz
TD (delay)	0.0	seconds
THETA (damping factor)	0.0	1/seconds

The shape of the waveform is described by the following table:

time	value
0 to TD	VO
TD to TSTOP	$VO + VA\, e^{-(t - TD)THETA} \sin(2\,\pi\, FREQ\,(t + TD))$

3.2.1.3. Exponential

General Form:

```
EXP(V1 V2 TD1 TAU1 TD2 TAU2)
```

Examples:

```
VIN 3 0 EXP(-4 -1 2NS 30NS 60NS 40NS)
```

parameter	default value	units
V1 (initial value)		Volts or Amps
V2 (pulsed value)		Volts or Amps
TD1 (rise delay time)	0.0	seconds
TAU1 (rise time constant)	TSTEP	seconds
TD2 (fall delay time)	TD1+TSTEP	seconds
TAU2 (fall time constant)	TSTEP	seconds

The shape of the waveform is described by the following table:

time	value
0 to TD1	V1
TD1 to TD2	$V1 + (V2 - V1)\left(1 - e^{\frac{-(t - TD1)}{TAU1}}\right)$
TD2 to TSTOP	$V1 + (V2 - V1)\left(-e^{\frac{-(t - TD1)}{TAU1}}\right) + (V1 - V2)\left(1 - e^{\frac{-(t - TD2)}{TAU2}}\right)$

3.2.1.4. Piece-Wise Linear

General Form:

```
PWL(T1 V1 <T2 V2 T3 V3 T4 V4 ...>)
```

Examples:

```
VCLOCK 7 5 PWL(0 -7 10NS -7 11NS -3 17NS -3 18NS -7 50NS -7)
```

Each pair of values (Ti, Vi) specifies that the value of the source is Vi (in Volts or Amps) at time=Ti. The value of the source at intermediate values of time is determined by using linear interpolation on the input values.

3.2.1.5. Single-Frequency FM

General Form:

```
SFFM(VO VA FC MDI FS)
```

Examples:

```
V1 12 0 SFFM(0 1M 20K 5 1K)
```

parameter	default value	units
VO (offset)		Volts or Amps
VA (amplitude)		Volts or Amps
FC (carrier frequency)	1/TSTOP	Hz
MDI (modulation index)		
FS (signal frequency)	1/TSTOP	Hz

The shape of the waveform is described by the following equation:

$$V(t)=V_O + V_A \sin\left(2 \pi FC\, t + MDI \sin(2 \pi FS\, t)\right)$$

3.2.2. Linear Dependent Sources

SPICE allows circuits to contain linear dependent sources characterized by any of the four equations

$i = g\, v$ $v = e\, v$ $i = f\, i$ $v = h\, i$

where g, e, f, and h are constants representing transconductance, voltage gain, current gain, and transresistance, respectively.

3.2.2.1. Linear Voltage-Controlled Current Sources

General form:

```
GXXXXXXX N+ N- NC+ NC- VALUE
```

Examples:

```
G1 2 0 5 0 0.1MMHO
```

N+ and N- are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. NC+ and NC- are the positive and negative controlling nodes, respectively. VALUE is the transconductance (in mhos).

3.2.2.2. Linear Voltage-Controlled Voltage Sources

General form:

```
EXXXXXXX N+ N- NC+ NC- VALUE
```

Examples:

```
E1 2 3 14 1 2.0
```

N+ is the positive node, and N- is the negative node. NC+ and NC- are the positive and negative controlling nodes, respectively. VALUE is the voltage gain.

3.2.2.3. Linear Current-Controlled Current Sources

General form:

```
FXXXXXXX N+ N- VNAM VALUE
```

Examples:

```
F1 13 5 VSENS 5
```

N+ and N- are the positive and negative nodes, respectively. Current flow is from the positive node, through the source, to the negative node. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the current gain.

3.2.2.4. Linear Current-Controlled Voltage Sources

General form:

```
HXXXXXXX N+ N- VNAM VALUE
```

Examples:

```
HX 5 17 VZ 0.5K
```

N+ and N- are the positive and negative nodes, respectively. VNAM is the name of a voltage source through which the controlling current flows. The direction of positive controlling current flow is from the positive node, through the source, to the negative node of VNAM. VALUE is the transresistance (in ohms).

3.2.3. Non-linear Dependent Sources

General form:

```
BXXXXXXX N+ N- <I=EXPR> <V=EXPR>
```

Examples:

```
B1 0 1 I=cos(v(1))+sin(v(2))
B1 0 1 V=ln(cos(log(v(1,2)^2)))-v(3)^4+v(2)^v(1)
B1 3 4 I=17
B1 3 4 V=exp(pi^i(vdd))
```

N+ is the positive node, and *N*- is the negative node. The values of the **V** and **I** parameters determine the voltages and currents across and through the device, respectively. If **I** is given then the device is a current source, and if **V** is given the device is a voltage source. One and only one of these parameters must be given.

The small-signal AC behavior of the nonlinear source is a linear dependent source (or sources) with a proportionality constant equal to the derivative (or derivatives) of the source at the DC operating point.

The expressions given for **V** and **I** may be any function of voltages and currents through voltage sources in the system. The following functions of real variables are defined:

abs	asinh	cosh	sin
acos	atan	exp	sinh
acosh	atanh	ln	sqrt
asin	cos	log	tan

The function "u" is the unit step function, with a value of one for arguments greater than one and a value of zero for arguments less than zero. The function "uramp" is the integral of the unit step: for an

input *x*, the value is zero if *x* is less than zero, or if *x* is greater than zero the value is *x*. These two functions are useful in sythesizing piece-wise non-linear functions, though convergence may be adversely affected.

The following standard operators are defined:

+ - * / ^ unary -

If the argument of log, ln, or sqrt becomes less than zero, the absolute value of the argument is used. If a divisor becomes zero or the argument of log or ln becomes zero, an error will result. Other problems may occur when the argument for a function in a partial derivative enters a region where that function is undefined.

To get time into the expression you can integrate the current from a constant current source with a capacitor and use the resulting voltage (don't forget to set the initial voltage across the capacitor). Non-linear resistors, capacitors, and inductors may be synthesized with the nonlinear dependent source. Non-linear resistors are obvious. Non-linear capacitors and inductors are implemented with their linear counter-parts by a change of variables implemented with the nonlinear dependent source. The following subcircuit will implement a nonlinear capacitor:

```
.Subckt nlcap   pos neg
* Bx: calculate f(input voltage)
Bx   1    0    v = f(v(pos,neg))
* Cx: linear capacitance
Cx   2    0    1
* Vx: Ammeter to measure current into the capacitor
Vx   2    1    DC 0Volts
* Drive the current through Cx back into the circuit
Fx   pos  neg  Vx 1
.ends
```

Non-linear inductors are similar.

3.3. TRANSMISSION LINES

3.3.1. Lossless Transmission Lines

General form:

```
TXXXXXXX N1 N2 N3 N4 Z0=VALUE <TD=VALUE> <F=FREQ <NL=NRMLEN>>
+                    <IC=V1, I1, V2, I2>
```

Examples:

```
T1 1 0 2 0 Z0=50 TD=10NS
```

N1 and N2 are the nodes at port 1; N3 and N4 are the nodes at port 2. Z0 is the characteristic impedance. The length of the line may be expressed in either of two forms. The transmission delay, TD, may be specified directly (as TD=10ns, for example). Alternatively, a frequency F may be given, together with NL, the normalized electrical length of the transmission line with respect to the wavelength in the line at the frequency F. If a frequency is specified but NL is omitted, 0.25 is assumed (that is, the frequency is assumed to be the quarter-wave frequency). Note that although both forms for expressing the line length are indicated as optional, one of the two must be specified.

Note that this element models only one propagating mode. If all four nodes are distinct in the actual circuit, then two modes may be excited. To simulate such a situation, two transmission-line elements are required. (see the example in Appendix A for further clarification.)

The (optional) initial condition specification consists of the voltage and current at each of the transmission line ports. Note that the initial conditions (if any) apply 'only' if the UIC option is specified on the .TRAN control line.

Note that a lossy transmission line (see below) with zero loss may be more accurate than than the lossless transmission line due to implementation details.

3.3.2. Lossy Transmission Lines

General form:

```
OXXXXXXX N1 N2 N3 N4 MNAME
```

Examples:

```
O23 1 0 2 0 LOSSYMOD
OCONNECT 10 5 20 5 INTERCONNECT
```

This is a two-port convolution model for single-conductor lossy transmission lines. N1 and N2 are the nodes at port 1; N3 and N4 are the nodes at port 2. Note that a lossy transmission line with zero loss may be more accurate than than the lossless transmission line due to implementation details.

3.3.3. Lossy Transmission Line Model (LTRA)

The uniform RLC/RC/LC/RG transmission line model (referred to as the LTRA model henceforth) models a uniform constant-parameter distributed transmission line. The RC and LC cases may also be modeled using the URC and TRA models; however, the newer LTRA model is usually faster and more accurate than the others. The operation of the LTRA model is based on the convolution of the transmission line's impulse responses with its inputs (see [8]).

The LTRA model takes a number of parameters, some of which must be given and some of which are optional.

name	parameter	units/type	default	example
R	resistance/length	Ω/unit	0.0	0.2
L	inductance/length	henrys/unit	0.0	9.13e-9
G	conductance/length	mhos/unit	0.0	0.0
C	capacitance/length	farads/unit	0.0	3.65e-12
LEN	length of line		no default	1.0
REL	breakpoint control	arbitrary unit	1	0.5
ABS	breakpoint control		1	5
NOSTEPLIMIT	don't limit timestep to less than line delay	flag	not set	set
NOCONTROL	don't do complex timestep control	flag	not set	set
LININTERP	use linear interpolation	flag	not set	set
MIXEDINTERP	use linear when quadratic seems bad		not set	set
COMPACTREL	special reltol for history compaction	flag	RELTOL	1.0e-3
COMPACTABS	special abstol for history compaction		ABSTOL	1.0e-9
TRUNCNR	use Newton-Raphson method for timestep control	flag	not set	set
TRUNCDONTCUT	don't limit timestep to keep impulse-response errors low	flag	not set	set

The following types of lines have been implemented so far: RLC (uniform transmission line with series loss only), RC (uniform RC line), LC (lossless transmission line), and RG (distributed series resistance and parallel conductance only). Any other combination will yield erroneous results and should not be tried. The length LEN of the line must be specified.

NOSTEPLIMIT is a flag that will remove the default restriction of limiting time-steps to less than the line delay in the RLC case. NOCONTROL is a flag that prevents the default limiting of the time-step based on convolution error criteria in the RLC and RC cases. This speeds up simulation but may in some cases reduce the accuracy of results. LININTERP is a flag that, when specified, will use linear interpolation instead of the default quadratic interpolation for calculating delayed signals. MIXEDINTERP is a flag that, when specified, uses a metric for judging whether quadratic interpolation is not applicable and if so

uses linear interpolation; otherwise it uses the default quadratic interpolation. TRUNCDONTCUT is a flag that removes the default cutting of the time-step to limit errors in the actual calculation of impulse-response related quantities. COMPACTREL and COMPACTABS are quantities that control the compaction of the past history of values stored for convolution. Larger values of these lower accuracy but usually increase simulation speed. These are to be used with the TRYTOCOMPACT option, described in the .OPTIONS section. TRUNCNR is a flag that turns on the use of Newton-Raphson iterations to determine an appropriate timestep in the timestep control routines. The default is a trial and error procedure by cutting the previous timestep in half. REL and ABS are quantities that control the setting of breakpoints.

The option most worth experimenting with for increasing the speed of simulation is REL. The default value of 1 is usually safe from the point of view of accuracy but occasionally increases computation time. A value greater than 2 eliminates all breakpoints and may be worth trying depending on the nature of the rest of the circuit, keeping in mind that it might not be safe from the viewpoint of accuracy. Breakpoints may usually be entirely eliminated if it is expected the circuit will not display sharp discontinuities. Values between 0 and 1 are usually not required but may be used for setting many breakpoints.

COMPACTREL may also be experimented with when the option TRYTOCOMPACT is specified in a .OPTIONS card. The legal range is between 0 and 1. Larger values usually decrease the accuracy of the simulation but in some cases improve speed. If TRYTOCOMPACT is not specified on a .OPTIONS card, history compaction is not attempted and accuracy is high. NOCONTROL, TRUNCDONTCUT and NOSTEPLIMIT also tend to increase speed at the expense of accuracy.

3.3.4. Uniform Distributed RC Lines (Lossy)

General form:

```
UXXXXXXX N1 N2 N3 MNAME L=LEN <N=LUMPS>
```

Examples:

```
U1 1 2 0 URCMOD L=50U
URC2 1 12 2 UMODL l=1MIL N=6
```

N1 and N2 are the two element nodes the RC line connects, while N3 is the node to which the capacitances are connected. MNAME is the model name, LEN is the length of the RC line in meters. LUMPS, if specified, is the number of lumped segments to use in modeling the RC line (see the model description for the action taken if this parameter is omitted).

3.3.5. Uniform Distributed RC Model (URC)

The URC model is derived from a model proposed by L. Gertzberrg in 1974. The model is accomplished by a subcircuit type expansion of the URC line into a network of lumped RC segments with internally generated nodes. The RC segments are in a geometric progression, increasing toward the middle of the URC line, with K as a proportionality constant. The number of lumped segments used, if not specified for the URC line device, is determined by the following formula:

$$N = \frac{\log\left[F_{max}\,\frac{R}{L}\,\frac{C}{L}\,2\pi L^2\left[\frac{(K-1)}{K}\right]^2\right]}{\log K}$$

The URC line is made up strictly of resistor and capacitor segments unless the ISPERL parameter is given a non-zero value, in which case the capacitors are replaced with reverse biased diodes with a zero-bias junction capacitance equivalent to the capacitance replaced, and with a saturation current of ISPERL amps per meter of transmission line and an optional series resistance equivalent to RSPERL ohms per meter.

	name	parameter	units	default	example	area
1	K	Propagation Constant	-	2.0	1.2	-
2	FMAX	Maximum Frequency of interest	Hz	1.0G	6.5Meg	-
3	RPERL	Resistance per unit length	Ω/m	1000	10	-
4	CPERL	Capacitance per unit length	F/m	1.0e-15	1pF	-
5	ISPERL	Saturation Current per unit length	A/m	0	-	-
6	RSPERL	Diode Resistance per unit length	Ω/m	0	-	-

3.4. TRANSISTORS AND DIODES

The area factor used on the diode, BJT, JFET, and MESFET devices determines the number of equivalent parallel devices of a specified model. The affected parameters are marked with an asterisk under the heading 'area' in the model descriptions below. Several geometric factors associated with the channel and the drain and source diffusions can be specified on the MOSFET device line.

Two different forms of initial conditions may be specified for some devices. The first form is included to improve the dc convergence for circuits that contain more than one stable state. If a device is specified OFF, the dc operating point is determined with the terminal voltages for that device set to zero. After convergence is obtained, the program continues to iterate to obtain the exact value for the terminal voltages. If a circuit has more than one dc stable state, the OFF option can be used to force the solution to correspond to a desired state. If a device is specified OFF when in reality the device is conducting, the program still obtains the correct solution (assuming the solutions converge) but more iterations are required since the program must independently converge to two separate solutions. The .NODESET control line serves a similar purpose as the OFF option. The .NODESET option is easier to apply and is the preferred means to aid convergence.

The second form of initial conditions are specified for use with the transient analysis. These are true 'initial conditions' as opposed to the convergence aids above. See the description of the .IC control line and the .TRAN control line for a detailed explanation of initial conditions.

3.4.1. Junction Diodes

General form:

```
DXXXXXXX N+ N- MNAME <AREA> <OFF> <IC=VD> <TEMP=T>
```

Examples:

```
DBRIDGE 2 10 DIODE1
DCLMP 3 7 DMOD 3.0 IC=0.2
```

N+ and N- are the positive and negative nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) starting condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification using IC=VD is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

3.4.2. Diode Model (D)

The dc characteristics of the diode are determined by the parameters IS and N. An ohmic resistance, RS, is included. Charge storage effects are modeled by a transit time, TT, and a nonlinear depletion layer capacitance which is determined by the parameters CJO, VJ, and M. The temperature dependence of the saturation current is defined by the parameters EG, the energy and XTI, the saturation current temperature exponent. The nominal temperature at which these parameters were measured is TNOM, which defaults to the circuit-wide value specified on the .OPTIONS control line. Reverse breakdown is modeled by an exponential increase in the reverse diode current and is determined by the parameters BV and IBV (both of which are positive numbers).

	name	parameter	units	default	example	area
1	IS	saturation current	A	1.0e-14	1.0e-14	*
2	RS	ohmic resistance	Ω	0	10	*
3	N	emission coefficient	-	1	1.0	
4	TT	transit-time	sec	0	0.1ns	
5	CJO	zero-bias junction capacitance	F	0	2pF	*
6	VJ	junction potential	V	1	0.6	
7	M	grading coefficient	-	0.5	0.5	
8	EG	activation energy	eV	1.11	1.11 Si 0.69 Sbd 0.67 Ge	
9	XTI	saturation-current temp. exp	-	3.0	3.0 jn 2.0 Sbd	
10	KF	flicker noise coefficient	-	0		
11	AF	flicker noise exponent	-	1		
12	FC	coefficient for forward-bias depletion capacitance formula	-	0.5		
13	BV	reverse breakdown voltage	V	infinite	40.0	
14	IBV	current at breakdown voltage	A	1.0e-3		
15	TNOM	parameter measurement temperature	°C	27	50	

3.4.3. Bipolar Junction Transistors (BJTs)

General form:

```
QXXXXXXX NC NB NE <NS> MNAME <AREA> <OFF> <IC=VBE, VCE> <TEMP=T>
```

Examples:

```
Q23 10 24 13 QMOD IC=0.6, 5.0
Q50A 11 26 4 20 MOD1
```

NC, NB, and NE are the collector, base, and emitter nodes, respectively. NS is the (optional) substrate node. If unspecified, ground is used. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for the dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification using IC=VBE, VCE is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line description for a better way to set transient initial conditions. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

3.4.4. BJT Models (NPN/PNP)

The bipolar junction transistor model in SPICE is an adaptation of the integral charge control model of Gummel and Poon. This modified Gummel-Poon model extends the original model to include several effects at high bias levels. The model automatically simplifies to the simpler Ebers-Moll model when certain parameters are not specified. The parameter names used in the modified Gummel-Poon model have been chosen to be more easily understood by the program user, and to reflect better both physical and circuit design thinking.

The dc model is defined by the parameters IS, BF, NF, ISE, IKF, and NE which determine the forward current gain characteristics, IS, BR, NR, ISC, IKR, and NC which determine the reverse current gain characteristics, and VAF and VAR which determine the output conductance for forward and reverse regions. Three ohmic resistances RB, RC, and RE are included, where RB can be high current dependent. Base charge storage is modeled by forward and reverse transit times, TF and TR, the forward transit time TF being bias dependent if desired, and nonlinear depletion layer capacitances which are determined by CJE, VJE, and MJE for the B-E junction , CJC, VJC, and MJC for the B-C junction and CJS, VJS, and MJS for the C-S (Collector-Substrate) junction. The temperature dependence of the saturation current, IS, is determined by the energy-gap, EG, and the saturation current temperature exponent, XTI. Additionally base current temperature dependence is modeled by the beta temperature exponent XTB in the new model. The values specified are assumed to have been measured at the temperature TNOM, which can be specified on the .OPTIONS control line or overridden by a specification on the .MODEL line.

The BJT parameters used in the modified Gummel-Poon model are listed below. The parameter names used in earlier versions of SPICE2 are still accepted.

Modified Gummel-Poon BJT Parameters.

	name	parameter	units	default	example	area
1	IS	transport saturation current	A	1.0e-16	1.0e-15	*
2	BF	ideal maximum forward beta	-	100	100	
3	NF	forward current emission coefficient	-	1.0	1	
4	VAF	forward Early voltage	V	infinite	200	
5	IKF	corner for forward beta high current roll-off	A	infinite	0.01	*
6	ISE	B-E leakage saturation current	A	0	1.0e-13	*
7	NE	B-E leakage emission coefficient	-	1.5	2	
8	BR	ideal maximum reverse beta	-	1	0.1	
9	NR	reverse current emission coefficient	-	1	1	
10	VAR	reverse Early voltage	V	infinite	200	
11	IKR	corner for reverse beta high current roll-off	A	infinite	0.01	*
12	ISC	B-C leakage saturation current	A	0	1.0e-13	*
13	NC	B-C leakage emission coefficient	-	2	1.5	
14	RB	zero bias base resistance	Ω	0	100	*
15	IRB	current where base resistance falls halfway to its min value	A	infinite	0.1	*
16	RBM	minimum base resistance at high currents	Ω	RB	10	*
17	RE	emitter resistance	Ω	0	1	*
18	RC	collector resistance	Ω	0	10	*
19	CJE	B-E zero-bias depletion capacitance	F	0	2pF	*
20	VJE	B-E built-in potential	V	0.75	0.6	
21	MJE	B-E junction exponential factor	-	0.33	0.33	
22	TF	ideal forward transit time	sec	0	0.1ns	
23	XTF	coefficient for bias dependence of TF	-	0		
24	VTF	voltage describing VBC dependence of TF	V	infinite		
25	ITF	high-current parameter for effect on TF	A	0		*
26	PTF	excess phase at freq=1.0/(TF*2PI) Hz	deg	0		
27	CJC	B-C zero-bias depletion capacitance	F	0	2pF	*
28	VJC	B-C built-in potential	V	0.75	0.5	

	name	parameter	units	default	example	area
29	MJC	B-C junction exponential factor	-	0.33	0.5	
30	XCJC	fraction of B-C depletion capacitance connected to internal base node	-	1		
31	TR	ideal reverse transit time	sec	0	10ns	
32	CJS	zero-bias collector-substrate capacitance	F	0	2pF	*
33	VJS	substrate junction built-in potential	V	0.75		
34	MJS	substrate junction exponential factor	-	0	0.5	
35	XTB	forward and reverse beta temperature exponent	-	0		
36	EG	energy gap for temperature effect on IS	eV	1.11		
37	XTI	temperature exponent for effect on IS	-	3		
38	KF	flicker-noise coefficient	-	0		
39	AF	flicker-noise exponent	-	1		
40	FC	coefficient for forward-bias depletion capacitance formula	-	0.5		
41	TNOM	Parameter measurement temperature	°C	27	50	

3.4.5. Junction Field-Effect Transistors (JFETs)

General form:

```
JXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS, VGS> <TEMP=T>
```

Examples:

```
J1 7 2 3 JM1 OFF
```

ND, NG, and NS are the drain, gate, and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification, using IC=VDS, VGS is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line for a better way to set initial conditions. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line.

3.4.6. JFET Models (NJF/PJF)

The JFET model is derived from the FET model of Shichman and Hodges. The dc characteristics are defined by the parameters VTO and BETA, which determine the variation of drain current with gate voltage, LAMBDA, which determines the output conductance, and IS, the saturation current of the two gate junctions. Two ohmic resistances, RD and RS, are included. Charge storage is modeled by nonlinear depletion layer capacitances for both gate junctions which vary as the -1/2 power of junction voltage and are defined by the parameters CGS, CGD, and PB.

Note that in Spice3f and later, a fitting parameter B has been added. For details, see [9].

	name	parameter	units	default	example	area
1	VTO	threshold voltage (V_{TO}	V	-2.0	-2.0	
2	BETA	transconductance parameter (β)	A/V^2	1.0e-4	1.0e-3	*
3	LAMBDA	channel-length modulation parameter (λ)	1/V	0	1.0e-4	
4	RD	drain ohmic resistance	Ω	0	100	*
5	RS	source ohmic resistance	Ω	0	100	*
6	CGS	zero-bias G-S junction capacitance (C_{gs})	F	0	5pF	*
7	CGD	zero-bias G-D junction capacitance (C_{gs})	F	0	1pF	*
8	PB	gate junction potential	V	1	0.6	
9	IS	gate junction saturation current (I_S)	A	1.0e-14	1.0e-14	*
10	B	doping tail parameter	-	1	1.1	
11	KF	flicker noise coefficient	-	0		
12	AF	flicker noise exponent	-	1		
13	FC	coefficient for forward-bias depletion capacitance formula	-	0.5		
14	TNOM	parameter measurement temperature	°C	27	50	

3.4.7. MOSFETs

General form:

```
MXXXXXXX ND NG NS NB MNAME <L=VAL> <W=VAL> <AD=VAL> <AS=VAL>
+ <PD=VAL> <PS=VAL> <NRD=VAL> <NRS=VAL> <OFF>
+ <IC=VDS, VGS, VBS> <TEMP=T>
```

Examples:

```
M1 24 2 0 20 TYPE1
M31 2 17 6 10 MODM L=5U W=2U
M1 2 9 3 0 MOD1 L=10U W=5U AD=100P AS=100P PD=40U PS=40U
```

ND, NG, NS, and NB are the drain, gate, source, and bulk (substrate) nodes, respectively. MNAME is the model name. L and W are the channel length and width, in meters. AD and AS are the areas of the drain and source diffusions, in meters^2. Note that the suffix U specifies microns (1e-6 m) and P sq-microns (1e-12 m^2). If any of L, W, AD, or AS are not specified, default values are used. The use of defaults simplifies input file preparation, as well as the editing required if device geometries are to be changed. PD and PS are the perimeters of the drain and source junctions, in meters. NRD and NRS designate the equivalent number of squares of the drain and source diffusions; these values multiply the sheet resistance RSH specified on the .MODEL control line for an accurate representation of the parasitic series drain and source resistance of each transistor. PD and PS default to 0.0 while NRD and NRS to 1.0. OFF indicates an (optional) initial condition on the device for dc analysis. The (optional) initial condition specification using IC=VDS, VGS, VBS is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line for a better and more convenient way to specify transient initial conditions. The (optional) TEMP value is the temperature at which this device is to operate, and overrides the temperature specification on the .OPTION control line. The temperature specification is ONLY valid for level 1, 2, 3, and 6 MOSFETs, not for level 4 or 5 (BSIM) devices.

3.4.8. MOSFET Models (NMOS/PMOS)

SPICE provides four MOSFET device models, which differ in the formulation of the I-V characteristic. The variable LEVEL specifies the model to be used:

LEVEL=1 ->	Shichman-Hodges
LEVEL=2 ->	MOS2 (as described in [1])
LEVEL=3 ->	MOS3, a semi-empirical model(see [1])
LEVEL=4 ->	BSIM (as described in [3])
LEVEL=5 ->	new BSIM (BSIM2; as described in [5])
LEVEL=6 ->	MOS6 (as described in [2])

The dc characteristics of the level 1 through level 3 MOSFETs are defined by the device parameters VTO, KP, LAMBDA, PHI and GAMMA. These parameters are computed by SPICE if process parameters (NSUB, TOX, ...) are given, but user-specified values always override. VTO is positive (negative) for enhancement mode and negative (positive) for depletion mode N-channel (P-channel) devices. Charge storage is modeled by three constant capacitors, CGSO, CGDO, and CGBO which represent overlap capacitances, by the nonlinear thin-oxide capacitance which is distributed among the gate, source, drain, and bulk regions, and by the nonlinear depletion-layer capacitances for both substrate junctions divided into bottom and periphery, which vary as the MJ and MJSW power of junction voltage respectively, and are determined by the parameters CBD, CBS, CJ, CJSW, MJ, MJSW and PB. Charge storage effects are modeled by the piecewise linear voltages-dependent capacitance model proposed by Meyer. The thin-oxide charge-storage effects are treated slightly different for the LEVEL=1 model. These voltage-dependent capacitances are included only if TOX is specified in the input description and they are represented using Meyer's formulation.

There is some overlap among the parameters describing the junctions, e.g. the reverse current can be input either as IS (in A) or as JS (in A/m^2). Whereas the first is an absolute value the second is multiplied by AD and AS to give the reverse current of the drain and source junctions respectively. This methodology has been chosen since there is no sense in relating always junction characteristics with AD and AS entered on the device line; the areas can be defaulted. The same idea applies also to the zero-bias junction capacitances CBD and CBS (in F) on one hand, and CJ (in F/m^2) on the other. The parasitic drain and source series resistance can be expressed as either RD and RS (in ohms) or RSH (in ohms/sq.), the latter being multiplied by the number of squares NRD and NRS input on the device line.

A discontinuity in the MOS level 3 model with respect to the KAPPA parameter has been detected (see [10]). The supplied fix has been implemented in Spice3f2 and later. Since this fix may affect parameter fitting, the option "BADMOS3" may be set to use the old implementation (see the section on simulation variables and the ".OPTIONS" line).

SPICE level 1, 2, 3 and 6 parameters:

	name	parameter	units	default	example
1	LEVEL	model index	-	1	
2	VTO	zero-bias threshold voltage (V_{TO})	V	0.0	1.0
3	KP	transconductance parameter	A/V^2	2.0e-5	3.1e-5
4	GAMMA	bulk threshold parameter (γ)	$V^{1/2}$	0.0	0.37
5	PHI	surface potential (ϕ)	V	0.6	0.65
6	LAMBDA	channel-length modulation (MOS1 and MOS2 only) (λ)	1/V	0.0	0.02
7	RD	drain ohmic resistance	Ω	0.0	1.0
8	RS	source ohmic resistance	Ω	0.0	1.0
9	CBD	zero-bias B-D junction capacitance	F	0.0	20fF
10	CBS	zero-bias B-S junction capacitance	F	0.0	20fF
11	IS	bulk junction saturation current (I_S)	A	1.0e-14	1.0e-15
12	PB	bulk junction potential	V	0.8	0.87
13	CGSO	gate-source overlap capacitance per meter channel width	F/m	0.0	4.0e-11
14	CGDO	gate-drain overlap capacitance per meter channel width	F/m	0.0	4.0e-11
15	CGBO	gate-bulk overlap capacitance per meter channel length	F/m	0.0	2.0e-10
16	RSH	drain and source diffusion sheet resistance	Ω/□	0.0	10.0
17	CJ	zero-bias bulk junction bottom cap. per sq-meter of junction area	F/m^2	0.0	2.0e-4
18	MJ	bulk junction bottom grading coeff.	-	0.5	0.5
19	CJSW	zero-bias bulk junction sidewall cap. per meter of junction perimeter	F/m	0.0	1.0e-9
20	MJSW	bulk junction sidewall grading coeff.	-	0.50(level1) 0.33(level2, 3)	
21	JS	bulk junction saturation current per sq-meter of junction area	A/m^2		1.0e-8
22	TOX	oxide thickness	meter	1.0e-7	1.0e-7
23	NSUB	substrate doping	$1/cm^3$	0.0	4.0e15
24	NSS	surface state density	$1/cm^2$	0.0	1.0e10
25	NFS	fast surface state density	$1/cm^2$	0.0	1.0e10

continued

	name	parameter	units	default	example
26	TPG	type of gate material: +1 opp. to substrate -1 same as substrate 0 Al gate	-	1.0	
27	XJ	metallurgical junction depth	meter	0.0	1µ
28	LD	lateral diffusion	meter	0.0	0.8µ
29	UO	surface mobility	cm^2/Vs	600	700
30	UCRIT	critical field for mobility degradation (MOS2 only)	V/cm	1.0e4	1.0e4
31	UEXP	critical field exponent in mobility degradation (MOS2 only)	-	0.0	0.1
32	UTRA	transverse field coeff. (mobility) (deleted for MOS2)	-	0.0	0.3
33	VMAX	maximum drift velocity of carriers	m/s	0.0	5.0e4
34	NEFF	total channel-charge (fixed and mobile) coefficient (MOS2 only)	-	1.0	5.0
35	KF	flicker noise coefficient	-	0.0	1.0e-26
36	AF	flicker noise exponent	-	1.0	1.2
37	FC	coefficient for forward-bias depletion capacitance formula	-	0.5	
38	DELTA	width effect on threshold voltage (MOS2 and MOS3)	-	0.0	1.0
39	THETA	mobility modulation (MOS3 only)	1/V	0.0	0.1
40	ETA	static feedback (MOS3 only)	-	0.0	1.0
41	KAPPA	saturation field factor (MOS3 only)	-	0.2	0.5
42	TNOM	parameter measurement temperature	°C	27	50

The level 4 and level 5 (BSIM1 and BSIM2) parameters are all values obtained from process characterization, and can be generated automatically. J. Pierret [4] describes a means of generating a 'process' file, and the program Proc2Mod provided with SPICE3 converts this file into a sequence of BSIM1 ".MODEL" lines suitable for inclusion in a SPICE input file. Parameters marked below with an * in the l/w column also have corresponding parameters with a length and width dependency. For example, VFB is the basic parameter with units of Volts, and LVFB and WVFB also exist and have units of Volt-µmeter The formula

$$P = P_0 + \frac{P_L}{L_{effective}} + \frac{P_W}{W_{effective}}$$

is used to evaluate the parameter for the actual device specified with

$$L_{effective} = L_{input} - DL$$

and

$$W_{effective} = W_{input} - DW$$

Note that unlike the other models in SPICE, the BSIM model is designed for use with a process characterization system that provides all the parameters, thus there are no defaults for the parameters, and leaving one out is considered an error. For an example set of parameters and the format of a process file, see the SPICE2 implementation notes[3].

For more information on BSIM2, see reference [5].

SPICE BSIM (level 4) parameters.

name	parameter	units	l/w
VFB	flat-band voltage	V	*
PHI	surface inversion potential	V	*
K1	body effect coefficient	$V^{1/2}$	*
K2	drain/source depletion charge-sharing coefficient	-	*
ETA	zero-bias drain-induced barrier-lowering coefficient	-	*
MUZ	zero-bias mobility	$cm^2/V–s$	
DL	shortening of channel	µm	
DW	narrowing of channel	µm	
U0	zero-bias transverse-field mobility degradation coefficient	V^{-1}	*
U1	zero-bias velocity saturation coefficient	µm/V	*
X2MZ	sens. of mobility to substrate bias at v_{ds}=0	$cm^2/V^2–s$	*
X2E	sens. of drain-induced barrier lowering effect to substrate bias	V^{-1}	*
X3E	sens. of drain-induced barrier lowering effect to drain bias at $V_{ds}=V_{dd}$	V^{-1}	*
X2U0	sens. of transverse field mobility degradation effect to substrate bias	V^{-2}	*
X2U1	sens. of velocity saturation effect to substrate bias	$µmV^{-2}$	*

name	parameter	units	l/w
MUS	mobility at zero substrate bias and at $V_{ds}=V_{dd}$	$cm^2/V^2–s$	
X2MS	sens. of mobility to substrate bias at $V_{ds}=V_{dd}$	$cm^2/V^2–s$	*
X3MS	sens. of mobility to drain bias at $V_{ds}=V_{dd}$	$cm^2/V^2–s$	*
X3U1	sens. of velocity saturation effect on drain bias at $V_{ds}=V_{dd}$	μmV^{-2}	*
TOX	gate oxide thickness	µm	
TEMP	temperature at which parameters were measured	°C	
VDD	measurement bias range	V	
CGDO	gate-drain overlap capacitance per meter channel width	F/m	
CGSO	gate-source overlap capacitance per meter channel width	F/m	
CGBO	gate-bulk overlap capacitance per meter channel length	F/m	
XPART	gate-oxide capacitance-charge model flag	-	
N0	zero-bias subthreshold slope coefficient	-	*
NB	sens. of subthreshold slope to substrate bias	-	*
ND	sens. of subthreshold slope to drain bias	-	*
RSH	drain and source diffusion sheet resistance	Ω/□	
JS	source drain junction current density	A/m^2	
PB	built in potential of source drain junction	V	
MJ	Grading coefficient of source drain junction	-	
PBSW	built in potential of source, drain junction sidewall	V	
MJSW	grading coefficient of source drain junction sidewall	-	
CJ	Source drain junction capacitance per unit area	F/m^2	
CJSW	source drain junction sidewall capacitance per unit length	F/m	
WDF	source drain junction default width	m	
DELL	Source drain junction length reduction	m	

XPART = 0 selects a 40/60 drain/source charge partition in saturation, while XPART=1 selects a 0/100 drain/source charge partition.

ND, NG, and NS are the drain, gate, and source nodes, respectively. MNAME is the model name, AREA is the area factor, and OFF indicates an (optional) initial condition on the device for dc analysis. If the area factor is omitted, a value of 1.0 is assumed. The (optional) initial condition specification, using IC=VDS, VGS is intended for use with the UIC option on the .TRAN control line, when a transient analysis is desired starting from other than the quiescent operating point. See the .IC control line for a better way to set initial conditions.

3.4.9. MESFETs

General form:

```
ZXXXXXXX ND NG NS MNAME <AREA> <OFF> <IC=VDS, VGS>
```

Examples:

```
Z1 7 2 3 ZM1 OFF
```

3.4.10. MESFET Models (NMF/PMF)

The MESFET model is derived from the GaAs FET model of Statz et al. as described in [11]. The dc characteristics are defined by the parameters VTO, B, and BETA, which determine the variation of drain current with gate voltage, ALPHA, which determines saturation voltage, and LAMBDA, which determines the output conductance. The formula are given by:

$$I_d = \frac{\beta (V_{gs}-V_T)^2}{1+b(V_{gs}-V_T)} \left[1-\left[1-\alpha\frac{V_{ds}}{3}\right]^3\right](1+\lambda V_{ds}) \qquad \text{for } 0 < V_{ds} < \frac{3}{\alpha}$$

$$I_d = \frac{\beta (V_{gs}-V_T)^2}{1+b(V_{gs}-V_T)}(1+\lambda V_{ds}) \qquad \text{for } V_{ds} > \frac{3}{\alpha}$$

Two ohmic resistances, RD and RS, are included. Charge storage is modeled by total gate charge as a function of gate-drain and gate-source voltages and is defined by the parameters CGS, CGD, and PB.

	name	parameter	units	default	example	area
1	VTO	pinch-off voltage	V	-2.0	-2.0	
2	BETA	transconductance parameter	A/V^2	1.0e-4	1.0e-3	*
3	B	doping tail extending parameter	1/V	0.3	0.3	*
4	ALPHA	saturation voltage parameter	1/V	2	2	*
5	LAMBDA	channel-length modulation parameter	1/V	0	1.0e-4	
6	RD	drain ohmic resistance	Ω	0	100	*
7	RS	source ohmic resistance	Ω	0	100	*
8	CGS	zero-bias G-S junction capacitance	F	0	5pF	*
9	CGD	zero-bias G-D junction capacitance	F	0	1pF	*
10	PB	gate junction potential	V	1	0.6	
11	KF	flicker noise coefficient	-	0		
12	AF	flicker noise exponent	-	1		
13	FC	coefficient for forward-bias depletion capacitance formula	-	0.5		

4. ANALYSES AND OUTPUT CONTROL

The following command lines are for specifying analyses or plots within the circuit description file. Parallel commands exist in the interactive command interpreter (detailed in the following section). Specifying analyses and plots (or tables) in the input file is useful for batch runs. Batch mode is entered when either the **-b** option is given or when the default input source is redirected from a file. In batch mode, the analyses specified by the control lines in the input file (e.g. ".ac", ".tran", etc.) are immediately executed (unless ".control" lines exists; see the section on the interactive command interpretor). If the **-r** *rawfile* option is given then all data generated is written to a Spice3 **rawfile**. The rawfile may be read by either the interactive mode of Spice3 or by **nutmeg**; see the previous section for details. In this case, the .SAVE line (see below) may be used to record the value of internal device variables (see Appendix B).

If a rawfile is not specified, then output plots (in "line-printer" form) and tables can be printed according to the **.PRINT**, **.PLOT**, and **.FOUR** control lines, described next. **.PLOT**, **.PRINT**, and **.FOUR** lines are meant for compatibility with Spice2.

4.1. SIMULATOR VARIABLES (.OPTIONS)

Various parameters of the simulations available in Spice3 can be altered to control the accuracy, speed, or default values for some devices. These parameters may be changed via the "set" command (described later in the section on the interactive front-end) or via the ".OPTIONS" line:

General form:

```
.OPTIONS OPT1 OPT2 ... (or OPT=OPTVAL ...)
```

Examples:

```
.OPTIONS RELTOL=.005 TRTOL=8
```

§4.1 ANALYSES AND OUTPUT CONTROL: SIMULATOR VARIABLES (.OPTIONS)

The options line allows the user to reset program control and user options for specific simulation purposes. Additional options for Nutmeg may be specified as well and take effect when Nutmeg reads the input file. Options specified to Nutmeg via the 'set' command are also passed on to SPICE3 as if specified on a .OPTIONS line. See the following section on the interactive command interpreter for the parameters which may be set with a .OPTIONS line and the format of the 'set' command. Any combination of the following options may be included, in any order. 'x' (below) represents some positive number.

option	effect
ABSTOL=x	resets the absolute current error tolerance of the program. The default value is 1 picoamp.
BADMOS3	Use the older version of the MOS3 model with the "kappa" discontinuity.
CHGTOL=x	resets the charge tolerance of the program. The default value is 1.0e-14.
DEFAD=x	resets the value for MOS drain diffusion area; the default is 0.0.
DEFAS=x	resets the value for MOS source diffusion area; the default is 0.0.
DEFL=x	resets the value for MOS channel length; the default is 100.0 micrometer.
DEFW=x	resets the value for MOS channel width; the default is 100.0 micrometer.
GMIN=x	resets the value of GMIN, the minimum conductance allowed by the program. The default value is 1.0e-12.
ITL1=x	resets the dc iteration limit. The default is 100.
ITL2=x	resets the dc transfer curve iteration limit. The default is 50.
ITL3=x	resets the lower transient analysis iteration limit. the default value is 4. (Note: not implemented in Spice3).
ITL4=x	resets the transient analysis timepoint iteration limit. the default is 10.
ITL5=x	resets the transient analysis total iteration limit. the default is 5000. Set ITL5=0 to omit this test. (Note: not implemented in Spice3).
KEEPOPINFO	Retain the operating point information when either an AC, Distortion, or Pole-Zero analysis is run. This is particularly useful if the circuit is large and you do not want to run a (redundant) ".OP" analysis.

option	effect
METHOD=name	sets the numerical integration method used by SPICE. Possible names are "Gear" or "trapezoidal" (or just "trap"). The default is trapezoidal.
PIVREL=x	resets the relative ratio between the largest column entry and an acceptable pivot value. The default value is 1.0e-3. In the numerical pivoting algorithm the allowed minimum pivot value is determined by EPSREL=AMAX1(PIVREL*MAXVAL, PIVTOL) where MAXVAL is the maximum element in the column where a pivot is sought (partial pivoting).
PIVTOL=x	resets the absolute minimum value for a matrix entry to be accepted as a pivot. The default value is 1.0e-13.
RELTOL=x	resets the relative error tolerance of the program. The default value is 0.001 (0.1%).
TEMP=x	Resets the operating temperature of the circuit. The default value is 27 deg C (300 deg K). TEMP can be overridden by a temperature specification on any temperature dependent instance.
TNOM=x	resets the nominal temperature at which device parameters are measured. The default value is 27 deg C (300 deg K). TNOM can be overridden by a specification on any temperature dependent device model.
TRTOL=x	resets the transient error tolerance. The default value is 7.0. This parameter is an estimate of the factor by which SPICE overestimates the actual truncation error.
TRYTOCOMPACT	Applicable only to the LTRA model. When specified, the simulator tries to condense LTRA transmission lines' past history of input voltages and currents.
VNTOL=x	resets the absolute voltage error tolerance of the program. The default value is 1 microvolt.

In addition, the following options have the listed effect when operating in spice2 emulation mode:

option	effect

option	effect
ACCT	causes accounting and run time statistics to be printed
LIST	causes the summary listing of the input data to be printed
NOMOD	suppresses the printout of the model parameters
NOPAGE	suppresses page ejects
NODE	causes the printing of the node table.

option	effect
OPTS	causes the option values to be printed.

4.2. INITIAL CONDITIONS

4.2.1. .NODESET: Specify Initial Node Voltage Guesses

General form:

```
.NODESET V(NODNUM)=VAL V(NODNUM)=VAL ...
```

Examples:

```
.NODESET V(12)=4.5 V(4)=2.23
```

The Nodeset line helps the program find the dc or initial transient solution by making a preliminary pass with the specified nodes held to the given voltages. The restriction is then released and the iteration continues to the true solution. The .NODESET line may be necessary for convergence on bistable or a-stable circuits. In general, this line should not be necessary.

4.2.2. .IC: Set Initial Conditions

General form:

```
.IC V(NODNUM)=VAL V(NODNUM)=VAL ...
```

Examples:

```
.IC V(11)=5 V(4)=-5 V(2)=2.2
```

The IC line is for setting transient initial conditions. It has two different interpretations, depending on whether the UIC parameter is specified on the .TRAN control line. Also, one should not confuse this line with the .NODESET line. The .NODESET line is only to help dc convergence, and does not affect final bias solution (except for multi-stable circuits). The two interpretations of this line are as follows:

1. When the UIC parameter is specified on the .TRAN line, then the node voltages specified on the .IC control line are used to compute the capacitor, diode, BJT, JFET, and MOSFET initial conditions. This is equivalent to specifying the IC=... parameter on each device line, but is much more convenient. The IC=... parameter can still be specified and takes precedence over the .IC values. Since no dc bias (initial transient) solution is computed before the transient analysis, one should take care to specify all dc source voltages on the .IC control line if they are to be used to compute device initial conditions.

2. When the UIC parameter is not specified on the .TRAN control line, the dc bias (initial transient) solution is computed before the transient analysis. In this case, the node voltages specified on the .IC control line is forced to the desired initial values during the bias solution. During transient analysis, the constraint on these node voltages is removed. This is the preferred method since it allows SPICE to compute a consistent dc solution.

4.3. ANALYSES

4.3.1. .AC: Small-Signal AC Analysis

General form:

```
.AC DEC ND FSTART FSTOP
.AC OCT NO FSTART FSTOP
.AC LIN NP FSTART FSTOP
```

Examples:

```
.AC DEC 10 1 10K
.AC DEC 10 1K 100MEG
.AC LIN 100 1 100HZ
```

DEC stands for decade variation, and ND is the number of points per decade. OCT stands for octave variation, and NO is the number of points per octave. LIN stands for linear variation, and NP is the number of points. FSTART is the starting frequency, and FSTOP is the final frequency. If this line is included in the input file, SPICE performs an AC analysis of the circuit over the specified frequency range. Note that in order for this analysis to be meaningful, at least one independent source must have been specified with an ac value.

4.3.2. .DC: DC Transfer Function

General form:

```
.DC SRCNAM VSTART VSTOP VINCR [SRC2 START2 STOP2 INCR2]
```

Examples:

```
.DC VIN 0.25 5.0 0.25
.DC VDS 0 10 .5 VGS 0 5 1
.DC VCE 0 10 .25 IB 0 10U 1U
```

The DC line defines the dc transfer curve source and sweep limits (again with capacitors open and inductors shorted). SRCNAM is the name of an independent voltage or current source. VSTART, VSTOP, and VINCR are the starting, final, and incrementing values respectively. The first example causes the value of the voltage source VIN to be swept from 0.25 Volts to 5.0 Volts in increments of 0.25 Volts. A second source (SRC2) may optionally be specified with associated sweep parameters. In this case, the first source is swept over its range for each value of the second source. This option can be useful for obtaining semiconductor device output characteristics. See the second example circuit description in Appendix A.

4.3.3. .DISTO: Distortion Analysis

General form:

```
.DISTO DEC ND FSTART FSTOP <F2OVERF1>
.DISTO OCT NO FSTART FSTOP <F2OVERF1>
.DISTO LIN NP FSTART FSTOP <F2OVERF1>
```

Examples:

```
.DISTO DEC 10 1kHz 100Mhz
.DISTO DEC 10 1kHz 100Mhz 0.9
```

The Disto line does a small-signal distortion analysis of the circuit. A multi-dimensional Volterra series analysis is done using multi-dimensional Taylor series to represent the nonlinearities at the operating point. Terms of up to third order are used in the series expansions.

If the optional parameter F2OVERF1 is not specified, .DISTO does a harmonic analysis - i.e., it analyses distortion in the circuit using only a single input frequency F1, which is swept as specified by arguments of the .DISTO command exactly as in the .AC command. Inputs at this frequency may be present at more than one input source, and their magnitudes and phases are specified by the arguments of the DISTOF1 keyword in the input file lines for the input sources (see the description for independent sources). (The arguments of the DISTOF2 keyword are not relevant in this case). The analysis produces information about the A.C. values of all node voltages and branch currents at the harmonic frequencies 2F1 and 3F1, vs. the input frequency F1 as it is swept. (A value of 1 (as a complex distortion output) signifies $\cos(2\pi(2F1)t)$ at 2F1 and $\cos(2\pi(3F1)t)$ at 3F1, using the convention that 1 at the input fundamental frequency is equivalent to $\cos(2\pi F1t)$.) The distortion component desired (2F1 or 3F1) can be selected using commands in nutmeg, and then printed or plotted. (Normally, one is interested primarily in the magnitude of the harmonic components, so the magnitude of the AC distortion value is looked at). It should be noted that these are the A.C. values of the actual harmonic components, and are not equal to HD2 and HD3. To obtain HD2 and HD3, one must divide by the corresponding A.C. values at F1, obtained from an .AC line. This division can be done using nutmeg commands.

If the optional F2OVERF1 parameter is specified, it should be a real number between (and not equal to) 0.0 and 1.0; in this case, .DISTO does a spectral analysis. It considers the circuit with sinusoidal inputs at two different frequencies F1 and F2. F1 is swept according to the .DISTO control line options exactly as in the .AC control line. F2 is kept fixed at a single frequency as F1 sweeps - the value at which it is kept fixed is equal to F2OVERF1 times FSTART. Each independent source in the circuit may potentially have two (superimposed) sinusoidal inputs for distortion, at the frequencies F1 and F2. The magnitude and phase of the F1 component are specified by the arguments of the DISTOF1 keyword in the source's input line (see the description of independent sources); the magnitude and phase of the F2 component are specified by the arguments of the DISTOF2 keyword. The analysis produces plots of all node voltages/branch currents at the intermodulation product frequencies F1 + F2, F1 - F2, and (2 F1) - F2, vs the swept frequency F1. The IM product of interest may be selected using the setplot command, and displayed with the print and plot commands. It is to be noted as in the harmonic analysis case, the results are the actual AC

voltages and currents at the intermodulation frequencies, and need to be normalized with respect to .AC values to obtain the IM parameters.

If the DISTOF1 or DISTOF2 keywords are missing from the description of an independent source, then that source is assumed to have no input at the corresponding frequency. The default values of the magnitude and phase are 1.0 and 0.0 respectively. The phase should be specified in degrees.

It should be carefully noted that the number F2OVERF1 should ideally be an irrational number, and that since this is not possible in practice, efforts should be made to keep the denominator in its fractional representation as large as possible, certainly above 3, for accurate results (i.e., if F2OVERF1 is represented as a fraction A/B, where A and B are integers with no common factors, B should be as large as possible; note that A < B because F2OVERF1 is constrained to be < 1). To illustrate why, consider the cases where F2OVERF1 is 49/100 and 1/2. In a spectral analysis, the outputs produced are at F1 + F2, F1 - F2 and 2 F1 - F2. In the latter case, F1 - F2 = F2, so the result at the F1-F2 component is erroneous because there is the strong fundamental F2 component at the same frequency. Also, F1 + F2 = 2 F1 - F2 in the latter case, and each result is erroneous individually. This problem is not there in the case where F2OVERF1 = 49/100, because F1-F2 = 51/100 F1 < > 49/100 F1 = F2. In this case, there are two very closely spaced frequency components at F2 and F1 - F2. One of the advantages of the Volterra series technique is that it computes distortions at mix frequencies expressed symbolically (i.e. n F1 ± m F2), therefore one is able to obtain the strengths of distortion components accurately even if the separation between them is very small, as opposed to transient analysis for example. The disadvantage is of course that if two of the mix frequencies coincide, the results are not merged together and presented (though this could presumably be done as a postprocessing step). Currently, the interested user should keep track of the mix frequencies himself or herself and add the distortions at coinciding mix frequencies together should it be necessary.

4.3.4. .NOISE: Noise Analysis

General form:

```
.NOISE V(OUTPUT <,REF>) SRC ( DEC | LIN | OCT ) PTS FSTART FSTOP
+ <PTS_PER_SUMMARY>
```

Examples:

```
.NOISE V(5) VIN DEC 10 1kHZ 100Mhz
.NOISE V(5,3) V1 OCT 8 1.0 1.0e6 1
```

The Noise line does a noise analysis of the circuit. OUTPUT is the node at which the total output noise is desired; if REF is specified, then the noise voltage V(OUTPUT) - V(REF) is calculated. By default, REF is assumed to be ground. SRC is the name of an independent source to which input noise is referred. PTS, FSTART and FSTOP are .AC type parameters that specify the frequency range over which plots are desired. PTS_PER_SUMMARY is an optional integer; if specified, the noise contributions of each noise generator is produced every PTS_PER_SUMMARY frequency points.

The .NOISE control line produces two plots - one for the Noise Spectral Density curves and one for the total Integrated Noise over the specified frequency range. All noise voltages/currents are in squared units (V^2/Hz and A^2/Hz for spectral density, V^2 and A^2 for integrated noise).

4.3.5. .OP: Operating Point Analysis

General form:

```
.OP
```

The inclusion of this line in an input file directs SPICE to determine the dc operating point of the circuit with inductors shorted and capacitors opened. Note: a DC analysis is automatically performed prior to a transient analysis to determine the transient initial conditions, and prior to an AC small-signal, Noise, and Pole-Zero analysis to determine the linearized, small-signal models for nonlinear devices (see the KEEPOPINFO variable above).

4.3.6. .PZ: Pole-Zero Analysis

General form:

```
.PZ NODE1 NODE2 NODE3 NODE4 CUR POL
.PZ NODE1 NODE2 NODE3 NODE4 CUR ZER
.PZ NODE1 NODE2 NODE3 NODE4 CUR PZ
.PZ NODE1 NODE2 NODE3 NODE4 VOL POL
.PZ NODE1 NODE2 NODE3 NODE4 VOL ZER
.PZ NODE1 NODE2 NODE3 NODE4 VOL PZ
```

Examples:

```
.PZ 1 0 3 0 CUR POL
.PZ 2 3 5 0 VOL ZER
.PZ 4 1 4 1 CUR PZ
```

CUR stands for a transfer function of the type (output voltage)/(input current) while VOL stands for a transfer function of the type (output voltage)/(input voltage). POL stands for pole analysis only, ZER for zero analysis only and PZ for both. This feature is provided mainly because if there is a nonconvergence in finding poles or zeros, then, at least the other can be found. Finally, NODE1 and NODE2 are the two input nodes and NODE3 and NODE4 are the two output nodes. Thus, there is complete freedom regarding the output and input ports and the type of transfer function.

In interactive mode, the command syntax is the same except that the first field is PZ instead of .PZ. To print the results, one should use the command 'print all'.

4.3.7. .SENS: DC or Small-Signal AC Sensitivity Analysis

General form:

```
.SENS OUTVAR
.SENS OUTVAR AC DEC ND FSTART FSTOP
.SENS OUTVAR AC OCT NO FSTART FSTOP
.SENS OUTVAR AC LIN NP FSTART FSTOP
```

Examples:

```
.SENS V(1,OUT)
.SENS V(OUT) AC DEC 10 100 100k
.SENS I(VTEST)
```

The sensitivity of OUTVAR to all non-zero device parameters is calculated when the SENS analysis is specified. OUTVAR is a circuit variable (node voltage or voltage-source branch current). The first form calculates sensitivity of the DC operating-point value of OUTVAR. The second form calculates sensitivity of the AC values of OUTVAR. The parameters listed for AC sensitivity are the same as in an AC analysis (see ".AC" above). The output values are in dimensions of change in output per unit change of input (as opposed to percent change in output or per percent change of input).

4.3.8. .TF: Transfer Function Analysis

General form:

```
.TF OUTVAR INSRC
```

Examples:

```
.TF V(5, 3) VIN
.TF I(VLOAD) VIN
```

The TF line defines the small-signal output and input for the dc small-signal analysis. OUTVAR is the small-signal output variable and INSRC is the small-signal input source. If this line is included, SPICE computes the dc small-signal value of the transfer function (output/input), input resistance, and output resistance. For the first example, SPICE would compute the ratio of V(5, 3) to VIN, the small-signal input resistance at VIN, and the small-signal output resistance measured across nodes 5 and 3.

4.3.9. .TRAN: Transient Analysis

General form:

```
.TRAN TSTEP TSTOP <TSTART <TMAX>>
```

Examples:

```
.TRAN 1NS 100NS
.TRAN 1NS 1000NS 500NS
.TRAN 10NS 1US
```

TSTEP is the printing or plotting increment for line-printer output. For use with the post-processor, TSTEP is the suggested computing increment. TSTOP is the final time, and TSTART is the initial time. If TSTART is omitted, it is assumed to be zero. The transient analysis always begins at time zero. In the interval <zero, TSTART>, the circuit is analyzed (to reach a steady state), but no outputs are stored. In the interval <TSTART, TSTOP>, the circuit is analyzed and outputs are stored. TMAX is the maximum step-size that SPICE uses; for default, the program chooses either TSTEP or (TSTOP-TSTART)/50.0, whichever is smaller. TMAX is useful when one wishes to guarantee a computing interval which is smaller than the printer increment, TSTEP.

UIC (use initial conditions) is an optional keyword which indicates that the user does not want SPICE to solve for the quiescent operating point before beginning the transient analysis. If this keyword is specified, SPICE uses the values specified using IC=... on the various elements as the initial transient condition and proceeds with the analysis. If the .IC control line has been specified, then the node voltages on the .IC line are used to compute the initial conditions for the devices. Look at the description on the .IC control line for its interpretation when UIC is not specified.

4.4. BATCH OUTPUT

4.4.1. .SAVE Lines

General form:

`.SAVE` *vector vector vector ...*

Examples:

```
.SAVE i(vin) input output
.SAVE @m1[id]
```

The vectors listed on the .SAVE line are recorded in the rawfile for use later with spice3 or nutmeg (nutmeg is just the data-analysis half of spice3, without the ability to simulate). The standard vector names are accepted. If no .SAVE line is given, then the default set of vectors are saved (node voltages and voltage source branch currents). If .SAVE lines are given, only those vectors specified are saved. For more discussion on internal device data, see Appendix B. See also the section on the interactive command interpretor for information on how to use the rawfile.

4.4.2. .PRINT Lines

General form:

```
.PRINT PRTYPE OV1 <OV2 ... OV8>
```

Examples:

```
.PRINT TRAN V(4) I(VIN)
.PRINT DC V(2) I(VSRC) V(23, 17)
.PRINT AC VM(4, 2) VR(7) VP(8, 3)
```

The Print line defines the contents of a tabular listing of one to eight output variables. PRTYPE is the type of the analysis (DC, AC, TRAN, NOISE, or DISTO) for which the specified outputs are desired. The form for voltage or current output variables is the same as given in the previous section for the **print** command; Spice2 restricts the output variable to the following forms (though this restriction is not enforced by Spice3):

`V(N1<,N2>)`

specifies the voltage difference between nodes N1 and N2. If N2 (and the preceding comma) is omitted, ground (0) is assumed. See the **print** command in the previous section for more details. For compatibility with spice2, the following five additional values can be accessed for the ac analysis by replacing the "V" in `V(N1,N2)` with:

VR	-	real part
VI	-	imaginary part
VM	-	magnitude
VP	-	phase
VDB	-	20 log10(magnitude)

`I(VXXXXXXX)`

specifies the current flowing in the independent voltage source named VXXXXXXX. Positive current flows from the positive node, through the source, to the negative node. For the ac analysis, the corresponding replacements for the letter I may be made in the same way as described for voltage outputs.

Output variables for the noise and distortion analyses have a different general form from that of the other analyses.

There is no limit on the number of .PRINT lines for each type of analysis.

4.4.3. .PLOT Lines

General form:

```
.PLOT PLTYPE OV1 <(PLO1, PHI1)> <OV2 <(PLO2, PHI2)> ... OV8>
```

Examples:

```
.PLOT DC V(4) V(5) V(1)
.PLOT TRAN V(17, 5) (2, 5) I(VIN) V(17) (1, 9)
.PLOT AC VM(5) VM(31, 24) VDB(5) VP(5)
.PLOT DISTO HD2 HD3(R) SIM2
.PLOT TRAN V(5, 3) V(4) (0, 5) V(7) (0, 10)
```

The Plot line defines the contents of one plot of from one to eight output variables. PLTYPE is the type of analysis (DC, AC, TRAN, NOISE, or DISTO) for which the specified outputs are desired. The syntax for the OVI is identical to that for the .PRINT line and for the **plot** command in the interactive mode.

The overlap of two or more traces on any plot is indicated by the letter X.

When more than one output variable appears on the same plot, the first variable specified is printed as well as plotted. If a printout of all variables is desired, then a companion .PRINT line should be included.

There is no limit on the number of .PLOT lines specified for each type of analysis.

4.4.4. .FOUR: Fourier Analysis of Transient Analysis Output

General form:

```
.FOUR FREQ OV1 <OV2 OV3 ...>
```

Examples:

```
.FOUR 100K  V(5)
```

The Four (or Fourier) line controls whether SPICE performs a Fourier analysis as a part of the transient analysis. FREQ is the fundamental frequency, and OV1, ..., are the output variables for which the analysis is desired. The Fourier analysis is performed over the interval <TSTOP-period, TSTOP>, where TSTOP is the final time specified for the transient analysis, and period is one period of the fundamental frequency. The dc component and the first nine harmonics are determined. For maximum accuracy, TMAX (see the .TRAN line) should be set to period/100.0 (or less for very high-Q circuits).

5. INTERACTIVE INTERPRETER

Spice3 consists of a simulator and a front-end for data analysis and plotting. The front-end may be run as a separate "stand-alone" program under the name **Nutmeg**.

Nutmeg will read in the "raw" data output file created by **spice -r** or with the **write** command in an interactive Spice3 session. Nutmeg or interactive Spice3 can plot data from a simulation on a graphics terminal or a workstation display. Most of the commands available in the interactive Spice3 front end are available in **nutmeg**; where this is not the case, Spice-only commands have been marked with an asterisk ("*"). Note that the raw output file is different from the data that Spice2 writes to the standard output, which may also be produced by spice3 with the "-b" command line option.

Spice and **Nutmeg** use the **X Window System** for plotting if they find the environment variable **DISPLAY**. Otherwise, a graphics-terminal independent interface (**MFB**) is used. If you are using **X** on a workstation, the **DISPLAY** variable should already be set; if you want to display graphics on a system different from the one you are running **Spice3** or **Nutmeg** on, **DISPLAY** should be of the form "*machine*:0.0". See the appropriate documentation on the X Window Sytem for more details.

Command Synopsis

```
spice [ -n ] [ -t term ] [ -r rawfile] [ -b ] [ -i ] [ input file ... ]

nutmeg [ - ] [ -n ] [ -t term ] [ datafile ... ]
```

Options are:

- Don't try to load the default data file ("rawspice.raw") if no other files are given. **Nutmeg** only.

-n (or **-N**)
Don't try to source the file ".spiceinit" upon startup. Normally **spice** and **nutmeg** try to find the file in the current directory, and if it is not found then in the user's home directory.

-t term (or **-T term**)
The program is being run on a terminal with *mfb* name **term**.

-b (or **-B**)
Run in batch mode. Spice3 reads the default input source (e.g. keyboard) or reads the given input file and performs the analyses specified; output is either Spice2-like line-printer plots ("ascii plots") or a spice rawfile. See the following section for details. Note that if the input source is not a terminal (e.g. using the IO redirection notation of "<") Spice3 defaults to batch mode (**-i** overrides). This option is valid for **Spice3** only.

-s (or **-S**)
Run in server mode. This is like batch mode, except that a temporary rawfile is used and then written to the standard output, preceded by a line with a single "@", after the simulation is done. This mode is used by the spice daemon. This option is valid for **Spice3** only.

-i (or **-I**)

Run in interactive mode. This is useful if the standard input is not a terminal but interactive mode is desired. Command completion is not available unless the standard input is a terminal, however. This option is valid for **Spice3** only.

-r *rawfile* (or **-P** *rawfile*)

Use *rawfile* as the default file into which the results of the simulation are saved. This option is valid for **Spice3** only.

Further arguments to **spice** are taken to be Spice3 input files, which are read and saved (if running in batch mode then they are run immediately). Spice3 accepts most Spice2 input file, and output ascii plots, fourier analyses, and node printouts as specified in .plot, .four, and .print cards. If an **out** parameter is given on a .width card, the effect is the same as **set width = ...** Since Spice3 ascii plots do not use multiple ranges, however, if vectors together on a .plot card have different ranges they are not provide as much information as they would in Spice2. The output of Spice3 is also much less verbose than Spice2, in that the only data printed is that requested by the above cards.

For **nutmeg**, further arguments are taken to be data files in binary or ascii format (see **sconvert**(1)) which are loaded into nutmeg. If the file is in binary format, it may be only partially completed (useful for examining Spice2 output before the simulation is finished). One file may contain any number of data sets from different analyses.

5.1. EXPRESSIONS, FUNCTIONS, AND CONSTANTS

Spice and **Nutmeg** data is in the form of vectors: time, voltage, etc. Each vector has a type, and vectors can be operated on and combined algebraicly in ways consistent with their types. Vectors are normally created when a data file is read in (see the **load** command below), and when the initial datafile is loaded. They can also be created with the **let** command.

An expression is an algebraic formula involving vectors and scalars (a scalar is a vector of length 1) and the following operations:

\+ - * / ^ %

% is the modulo operator, and the comma operator has two meanings: if it is present in the argument list of a user-definable function, it serves to separate the arguments. Otherwise, the term **x , y** is synonymous with **x + j(y)**.

Also available are the logical operations & (and), | (or), ! (not), and the relational operations <, >, >=, <=, =, and <> (not equal). If used in an algebraic expression they work like they would in C, producing values of 0 or 1. The relational operators have the following synonyms:

gt	>
lt	<
ge	>=
le	<=
ne	<>
eq	=
and	&
or	|
not	!

These are useful when < and > might be confused with IO redirection (which is almost always).

The following functions are available:

mag(vector)	The magnitude of vector
ph(vector)	The phase of vector
j(vector)	*i* (sqrt(-1)) times vector
real(vector)	The real component of vector
imag(vector)	The imaginary part of vector
db(vector)	20 log10(mag(vector))
log(vector)	The logarithm (base 10) of vector
ln(vector)	The natural logarithm (base e) of vector
exp(vector)	e to the vector power
abs(vector)	The absolute value of vector.
sqrt(vector)	The square root of vector.
sin(vector)	The sine of vector.
cos(vector)	The cosine of vector.
tan(vector)	The tangent of vector.
atan(vector)	The inverse tangent of vector.
norm(vector)	The **vector** normalized to 1 (i.e, the largest magnitude of any component is 1).
rnd(vector)	A vector with each component a random integer between 0 and the absolute value of the vectors's corresponding component.
mean(vector)	The result is a scalar (a length 1 vector) that is the mean of the elements of **vector**.

vector(number)	The result is a vector of length **number**, with elements 0, 1, ... **number - 1**. If **number** is a vector then just the first element is taken, and if it isn't an integer then the floor of the magnitude is used.
length(vector)	The length of **vector**.
interpolate(plot.vector)	The result of interpolating the named vector onto the scale of the current plot. This function uses the variable **polydegree** to determine the degree of interpolation.
deriv(vector)	Calculates the derivative of the given vector. This uses numeric differentiation by interpolating a polynomial and may not produce satisfactory results (particularly with iterated differentiation). The implementation only caculates the dirivative with respect to the real componant of that vector's scale.

A vector may be either the name of a vector already defined or a floating-point number (a scalar). A number may be written in any format acceptable to SPICE, such as **14.6Meg** or **-1.231e-4**. Note that you can either use scientific notation or one of the abbreviations like *MEG* or *G*, but not both. As with SPICE, a number may have trailing alphabetic characters after it.

The notation **expr [num]** denotes the **num**'th element of **expr**. For multi-dimensional vectors, a vector of one less dimension is returned. Also for multi-dimensional vectors, the notation **expr[m][n]** will return the *n*th element of the **m**th subvector. To get a subrange of a vector, use the form **expr[lower, upper]**.

To reference vectors in a plot that is not the *current plot* (see the **setplot** command, below), the notation **plotname.vecname** can be used.

Either a plotname or a vector name may be the wildcard **all**. If the plotname is **all**, matching vectors from all plots are specified, and if the vector name is **all**, all vectors in the specified plots are referenced. Note that you may not use binary operations on expressions involving wildcards – it is not obvious what **all + all** should denote, for instance. Thus some (contrived) examples of expressions are:

```
cos(TIME) + db(v(3))
sin(cos(log([1 2 3 4 5 6 7 8 9 10])))
TIME * rnd(v(9)) - 15 * cos(vin#branch) ^ [7.9e5 8]
not ((ac3.FREQ[32] & tran1.TIME[10]) gt 3)
```

Vector names in **spice** may have a name such as **@name[param]**, where **name** is either the name of a device instance or model. This denotes the value of the **param** parameter of the device or model. See Appendix B for details of what parameters are available. The value is a vector of length 1. This function is also available with the **show** command, and is available with variables for convenience for command scripts.

There are a number of pre-defined constants in **nutmeg**. They are:

pi	π (3.14159...)
e	The base of natural logarithms (2.71828...)
c	The speed of light (299,792,500 m/sec)
i	The square root of -1
kelvin	Absolute 0 in Centigrade (-273.15 °C)
echarge	The charge on an electron (1.6021918e-19 C)
boltz	Boltzman's constant (1.3806226e-23)
planck	Planck's constant (h = 6.626200e-34)

These are all in MKS units. If you have another variable with a name that conflicts with one of these then it takes precedence.

5.2. COMMAND INTERPRETATION

If a word is typed as a command, and there is no built-in command with that name, the directories in the *sourcepath* list are searched in order for the file. If it is found, it is read in as a command file (as if it were **source**d). Before it is read, however, the variables *argc* and *argv* are set to the number of words following the filename on the command line, and a list of those words respectively. After the file is finished, these variables are **unset**. Note that if a command file calls another, it must save its *argv* and *argc* since they are altered. Also, command files may not be re-entrant since there are no local variables. (Of course, the procedures may explicitly manipulate a stack...) This way one can write scripts analogous to shell scripts for **nutmeg** and Spice3.

Note that for the script to work with Spice3, it must begin with a blank line (or whatever else, since it is thrown away) and then a line with **.control** on it. This is an unfortunate result of the **source** command being used for both circuit input and command file execution. Note also that this allows the user to merely type the name of a circuit file as a command and it is automatically run. The commands are executed immediately, without running any analyses that may be spicified in the circuit (to execute the analyses before the script executes, include a "run" command in the script).

There are various command scripts installed in */usr/local/lib/spice/scripts* (or whatever the path is on your machine), and the default *sourcepath* includes this directory, so you can use these command files (almost) like builtin commands.

5.3. COMMANDS

5.3.1. Ac*: Perform an AC, small-signal frequency response analysis

General Form

`ac ( DEC | OCT | LIN )` *N Fstart Fstop*

Do an ac analysis. See the previous sections of this manual for more details.

5.3.2. Alias: Create an alias for a command

General Form

`alias [word] [text ...]`

Causes **word** to be aliased to **text**. History substitutions may be used, as in C-shell aliases.

5.3.3. Alter*: Change a device or model parameter

General Form

`alter` *device value*
`alter` *device parameter value [parameter value]*

Alter changes the value for a device or a specified parameter of a device or model. The first form is used by simple devices which have one principal value (resistors, capacitors, etc.) where the second form is for more complex devices (bjt's, etc.). Model parameters can be changed with the second form if the name contains a "#".

For specifying vectors as values, start the vector with "[", followed by the values in the vector, and end with "]". Be sure to place a space between each of the values and before and after the "[" and "]".

5.3.4. Asciiplot: Plot values using old-style character plots

General Form

```
asciiplot plotargs
```

Produce a line printer plot of the vectors. The plot is sent to the standard output, so you can put it into a file with *asciiplot args ... > file*. The **set** options **width, height,** and **nobreak** determine the width and height of the plot, and whether there are page breaks, respectively. Note that you will have problems if you try to **asciiplot** something with an X-scale that isn't monotonic (i.e, something like *sin(TIME)*), because **asciiplot** uses a simple-minded linear interpolation.

5.3.5. Aspice: Asynchronous spice run

General Form

```
aspice input-file [output-file]
```

Start a SPICE-3 run, and when it is finished load the resulting data. The raw data is kept in a temporary file. If *output-file* is specified then the diagnostic output is directed into that file, otherwise it is thrown away.

5.3.6. Bug: Mail a bug report

General Form

```
bug
```

Send a bug report. Please include a short summary of the problem, the version number and name of the operating system that you are running, the version of Spice that you are running, and the relevant spice input file. (If you have defined BUGADDR, the mail is delivered to there.)

5.3.7. Cd: Change directory

General Form

```
cd [directory]
```

Change the current working directory to **directory**, or to the user's home directory if none is given.

5.3.8. Destroy: Delete a data set

General Form

`destroy [`*plotnames* `|  all]`

Release the memory holding the data for the specified runs.

5.3.9. Dc*: Perform a DC-sweep analysis

General Form

`dc` *Source-Name Vstart Vstop Vincr [Source2 Vstart2 Vstop2 Vincr2]*

Do a dc transfer curve analysis. See the previous sections of this manual for more details.

5.3.10. Define: Define a function

General Form

```
define function(arg1, arg2, ...) expression
```

Define the *user-definable function* with the name *function* and arguments *arg1, arg2,* ... to be *expression*, which may involve the arguments. When the function is later used, the arguments it is given are substituted for the formal arguments when it is parsed. If *expression* is not present, any definition for *function* is printed, and if there are no arguments to *define* then all currently active definitions are printed. Note that you may have different functions defined with the same name but different arities.

Some useful definitions are:

```
define max(x,y) (x > y) * x + (x <= y) * y
define min(x,y) (x < y) * x + (x >= y) * y
```

5.3.11. Delete*: Remove a trace or breakpoint

General Form

`delete` *[debug-number ...]*

Delete the specified breakpoints and traces. The **debug numbers** are those shown by the **status** command (unless you do **status > file**, in which case the debug numbers are not printed).

5.3.12. Diff: Compare vectors

General Form

```
diff plot1 plot2 [vec ...]
```

Compare all the vectors in the specified *plots*, or only the named vectors if any are given. There are different vectors in the two plots, or any values in the vectors differ significantly the difference is reported. The variable **diff_abstol**, **diff_reltol,** and **diff_vntol** are used to determine a significant difference.

5.3.13. Display: List known vectors and types

General Form

```
display [varname ...]
```

Prints a summary of currently defined vectors, or of the names specified. The vectors are sorted by name unless the variable **nosort** is set. The information given is the name of the vector, the length, the type of the vector, and whether it is real or complex data. Additionally, one vector is labeled **[scale]**. When a command such as *plot* is given without a *vs* argument, this scale is used for the X-axis. It is always the first vector in a rawfile, or the first vector defined in a new plot. If you undefine the scale (i.e, *let TIME = []*), one of the remaining vectors becomes the new scale (which is undetermined).

5.3.14. Echo: Print text

General Form

```
echo [text...]
```

Echos the given text to the screen.

5.3.15. Edit*: Edit the current circuit

General Form

```
edit [ file ]
```

Print the current Spice3 input file into a file, call up the editor on that file and allow the user to modify it, and then read it back in, replacing the original file. If a *filename* is given, then edit that file and load it, making the circuit the current one.

5.3.16. Fourier: Perform a fourier transform

General Form

```
fourier fundamental_frequency [value ...]
```

Does a fourier analysis of each of the given values, using the first 10 multiples of the fundamental frequency (or the first *nfreqs*, if that variable is set – see below). The output is like that of the **.four** Spice3 line. The values may be any valid expression. The values are interpolated onto a fixed-space grid with the number of points given by the **fourgridsize** variable, or 200 if it is not set. The interpolation is of degree **polydegree** if that variable is set, or 1. If **polydegree** is 0, then no interpolation is done. This is likely to give erroneous results if the time scale is not monotonic, though.

5.3.17. Hardcopy: Save a plot to a file for printing

General Form

```
hardcopy file plotargs
```

Just like **plot**, except creates a file called **file** containing the plot. The file is an image in *plot(5)* format, and can be printed by either the **plot(1)** program or **lpr** with the **-g** flag.

5.3.18. Help: Print summaries of Spice3 commands

General Form

```
help [all] [command ...]
```

Prints help. If the argument **all** is given, a short description of everything you could possibly type is printed. If **command**s are given, descriptions of those commands are printed. Otherwise help for only a few major commands is printed.

5.3.19. History: Review previous commands

General Form

```
history [number]
```

Print out the history, or the last **number** commands typed at the keyboard. *Note:* in Spice3 version 3a7 and earlier, all commands (including ones read from files) were saved.

5.3.20. Iplot*: Incremental plot

General Form

```
iplot [ node ...]
```

Incrementally plot the values of the nodes while Spice3 runs. The **iplot** command can be used with the **where** command to find trouble spots in a transient simulation.

5.3.21. Jobs: List active asynchronous spice runs

General Form

```
jobs
```

Report on the asynchronous SPICE-3 jobs currently running. **Nutmeg** checks to see if the jobs are finished every time you execute a command. If it is done then the data is loaded and becomes available.

5.3.22. Let: Assign a value to a vector

General Form

```
let name = expr
```

Creates a new vector called **name** with the value specified by **expr,** an expression as described above. If **expr** is [] (a zero-length vector) then the vector becomes undefined. Individual elements of a vector may be modified by appending a subscript to **name** (ex. **name[0]**). If there are no arguments, **let** is the same as **display**.

5.3.23. Linearize*: Interpolate to a linear scale

General Form

```
linearize vec ...
```

Create a new plot with all of the vectors in the current plot, or only those mentioned if arguments are given. The new vectors are interpolated onto a linear time scale, which is determined by the values of **tstep, tstart,** and **tstop** in the currently active transient analysis. The currently loaded input file must include a transient analysis (a **tran** command may be run interactively before the last **reset**, alternately), and the current plot must be from this transient analysis. This command is needed because Spice3 doesn't output the results from a transient analysis in the same manner that Spice2 did.

5.3.24. Listing*: Print a listing of the current circuit

General Form

```
listing [logical] [physical] [deck] [expand]
```

If the **logical** argument is given, the listing is with all continuation lines collapsed into one line, and if the **physical** argument is given the lines are printed out as they were found in the file. The default is **logical**. A **deck** listing is just like the **physical** listing, except without the line numbers it recreates the input file verbatim (except that it does not preserve case). If the word **expand** is present, the circuit is printed with all subcircuits expanded.

5.3.25. Load: Load rawfile data

General Form

```
load [filename] ...
```

Loads either binary or ascii format rawfile data from the files named. The default filename is **rawspice.raw**, or the argument to the **-r** flag if there was one.

5.3.26. Op*: Perform an operating point analysis

General Form

```
op
```

Do an operating point analysis. See the previous sections of this manual for more details.

5.3.27. Plot: Plot values on the display

General Form

```
plot exprs [ylimit ylo yhi] [xlimit xlo xhi] [xindices xilo xihi]
     [xcompress comp] [xdelta xdel] [ydelta ydel] [xlog] [ylog] [loglog]
     [vs xname] [xlabel word] [ylabel word] [title word] [samep]
    [linear]
```

Plot the given **exprs** on the screen (if you are on a graphics terminal). The **xlimit** and **ylimit** arguments determine the high and low x- and y-limits of the axes, respectively. The **xindices** arguments determine what range of points are to be plotted – everything between the **xilo**'th point and the **xihi**'th point is plotted. The **xcompress** argument specifies that only one out of every **comp** points should be plotted. If an **xdelta** or a **ydelta** parameter is present, it specifies the spacing between grid lines on the X- and Y-axis. These parameter names may be abbreviated to **xl, yl, xind, xcomp, xdel,** and **ydel** respectively.

The **xname** argument is an expression to use as the scale on the x-axis. If **xlog** or **ylog** are present then the X or Y scale, respectively, is logarithmic (**loglog** is the same as specifying both). The **xlabel** and **ylabel** arguments cause the specified labels to be used for the X and Y axes, respectively.

If **samep** is given, the values of the other parameters (other than **xname**) from the previous **plot, hardcopy,** or **asciiplot** command is used unless re-defined on the command line.

The **title** argument is used in the place of the plot name at the bottom of the graph.

The **linear** keyword is used to override a default log-scale plot (as in the output for an AC analysis).

Finally, the keyword **polar** to generate a polar plot. To produce a smith plot, use the keyword **smith.** Note that the data is transformed, so for smith plots you will see the data transformed by the function (x–1)/(x+1). To produce a polar plot with a smith grid but without performing the smith transform, use the keyword **smithgrid.**

5.3.28. Print: Print values

General Form

```
print [col] [line] expr ...
```

Prints the vector described by the expression **expr.** If the **col** argument is present, print the vectors named side by side. If **line** is given, the vectors are printed horizontally. **col** is the default, unless all the vectors named have a length of one, in which case **line** is the default. The options **width, length,** and **nobreak** are effective for this command (see **asciiplot**). If the expression is **all**, all of the vectors available are printed. Thus **print col all > file** prints everything in the file in SPICE2 format. The scale vector (time, frequency) is always in the first column unless the variable **noprintscale** is true.

5.3.29. Quit: Leave Spice3 or Nutmeg

General Form

```
quit
```

Quit nutmeg or spice.

5.3.30. Rehash: Reset internal hash tables

General Form

```
rehash
```

Recalculate the internal hash tables used when looking up UNIX commands, and make all UNIX commands in the user's PATH available for command completion. This is useless unless you have **set unixcom** first (see above).

5.3.31. Reset*: Reset an analysis

General Form

```
reset
```

Throw out any intermediate data in the circuit (e.g, after a breakpoint or after one or more analyses have been done already), and re-parse the input file. The circuit can then be re-run from it's initial state, overriding the affect of any **set** or **alter** commands. In Spice-3e and earlier versions this was done automatically by the **run** command.

5.3.32. Reshape: Alter the dimensionality or dimensions of a vector

General Form

`reshape` *vector vector* ...
or
`reshape` *vector vector* ... `[` *dimension, dimension, ...* `]`
or
`reshape` *vector vector* ... `[` *dimension* `][` *dimension* `]` ...

This command changes the dimensions of a vector or a set of vectors. The final dimension may be left off and it will be filled in automatically. If no dimensions are specified, then the dimensions of the first vector are copied to the other vectors. An error message of the form 'dimensions of *x* were inconsistent' can be ignored.

5.3.33. Resume*: Continue a simulation after a stop

General Form

`resume`

Resume a simulation after a stop or interruption (control-C).

5.3.34. Rspice: Remote spice submission

General Form

`rspice` *input file*

Runs a SPICE-3 remotely taking the **input file** as a SPICE-3 input file, or the current circuit if no argument is given. **Nutmeg** or **Spice3** waits for the job to complete, and passes output from the remote job to the user's standard output. When the job is finished the data is loaded in as with aspice. If the variable *rhost* is set, **nutmeg** connects to this host instead of the default remote Spice3 server machine. This command uses the "rsh" command and thereby requires authentication via a ".rhosts" file or other equivalent method. Note that "rsh" refers to the "remote shell" program, which may be "remsh" on your system; to override the default name of "rsh", set the variable *remote_shell*. If the variable *rprogram* is set, then **rspice** uses this as the pathname to the program to run on the remote system.

Note: rspice will not acknowledge elements that have been changed via the "alter" or "altermod" commands.

5.3.35. Run*: Run analysis from the input file

General Form

```
run [rawfile]
```

Run the simulation as specified in the input file. If there were any of the control lines .ac, .op, able interactively. In Spice-3e and earlier versions, the input file would be re-read and any affects of the **set** or **alter** commands would be reversed. This is no longer the affect.

5.3.36. Rusage: Resource usage

General Form

```
rusage [resource ...]
```

Print resource usage statistics. If any **resource**s are given, just print the usage of that resource. Most resources require that a circuit be loaded. Currently valid **resource**s are:

elapsed	The amount of time elapsed since the last **rusage elaped** call.
faults	Number of page faults and context switches (BSD only).
space	Data space used.
time	CPU time used so far.
temp	Operating temperature.
tnom	Temperature at which device parameters were measured.
equations	Circuit Equations
time	Total Analysis Time
totiter	Total iterations
accept	Accepted timepoints
rejected	Rejected timepoints
loadtime	Time spent loading the circuit matrix and RHS.
reordertime	Matrix reordering time
lutime	L-U decomposition time
solvetime	Matrix solve time
trantime	Transient analysis time
tranpoints	Transient timepoints
traniter	Transient iterations
trancuriters	Transient iterations for the last time point*
tranlutime	Transient L-U decomposition time
transolvetime	Transient matrix solve time
everything	All of the above.

* listed incorrectly as "Transient iterations per point".

5.3.37. Save*: Save a set of outputs

General Form

```
save [all | output ...]
.save [all | output ...]
```

Save a set of outputs, discarding the rest. If a node has been mentioned in a **save** command, it appears in the working plot after a run has completed, or in the rawfile if spice is run in batch mode. If a node is traced or plotted (see below) it is also saved. For backward compatibility, if there are **no** save commands given, all outputs are saved.

When the keyword "all" appears in the save command, all default values (node voltages and voltage source currents) are saved in addition to any other values listed.

5.3.38. Sens*: Run a sensitivity analysis

General Form

```
sens output_variable
sens output_variable ac ( DEC | OCT | LIN ) N Fstart Fstop
```

Perform a Sensitivity analysis. *output_variable* is either a node voltage (ex. "v(1)" or "v(A,out)") or a current through a voltage source (ex. "i(vtest)"). The first form calculates DC sensitivities, the second form calculates AC sensitivies. The output values are in dimensions of change in output per unit change of input (as opposed to percent change in output or per percent change of input).

5.3.39. Set: Set the value of a variable

General Form

```
set [word]
set [word = value] ...
```

Set the value of **word** to be **value**, if it is present. You can set any word to be any value, numeric or string. If no value is given then the value is the boolean 'true'.

The value of *word* may be inserted into a command by writing *$word*. If a variable is set to a list of values that are enclosed in parentheses (which **must** be separated from their values by white space), the value of the variable is the list.

The variables used by nutmeg are listed in the following section.

5.3.40. Setcirc*: Change the current circuit

General Form

```
setcirc [circuit name]
```

The current circuit is the one that is used for the simulation commands below. When a circuit is loaded with the **source** command (see below) it becomes the current circuit.

5.3.41. Setplot: Switch the current set of vectors

General Form

```
setplot [plotname]
```

Set the **current plot** to the plot with the given name, or if no name is given, prompt the user with a menu. (Note that the plots are named as they are loaded, with names like **tran1** or **op2.** These names are shown by the **setplot** and **display** commands and are used by **diff**, below.) If the "New plot" item is selected, the current plot becomes one with no vectors defined.

Note that here the word "plot" refers to a group of vectors that are the result of one SPICE run. When more than one file is loaded in, or more than one plot is present in one file, **nutmeg** keeps them separate and only shows you the vectors in the current plot.

5.3.42. Settype: Set the type of a vector

General Form

```
settype type vector ...
```

Change the type of the named vectors to **type.** Type names can be found in the manual page for **sconvert.**

5.3.43. Shell: Call the command interpreter

General Form

```
shell [ command ]
```

Call the operating system's command interpreter; execute the specified command or call for interactive use.

5.3.44. Shift: Alter a list variable

General Form

```
shift [varname] [number]
```

If *varname* is the name of a list variable, it is shifted to the left by *number* elements (i.e, the *number* leftmost elements are removed). The default *varname* is **argv**, and the default *number* is 1.

5.3.45. Show*: List device state

General Form

`show` *devices [* **:** *parameters] , ...*

Old Form

`show -v @`*device [* `[` *name* `]` *]*

The **show** command prints out tables summarizing the operating condition of selected devices (much like the **spice2** operation point summary). If *device* is missing, a default set of devices are listed, if *device* is a single letter, devices of that type are listed; if *device* is a subcircuit name (beginning and ending in ":") only devices in that subcircuit are shown (end the name in a double-":" to get devices within sub-subcircuits recursively). The second and third forms may be combined ("letter:subcircuit:") or "letter:subcircuit::") to select a specific type of device from a subcircuit. A device's full name may be specified to list only that device. Finally, devices may be selected by model by using the form "#modelname" or ":subcircuit#modelname" or "letter:subcircuit#modelname".

If no *parameters* are specified, the values for a standard set of parameters are listed. If the list of *parameters* contains a "+", the default set of parameters is listed along with any other specified parameters.

For both *devices* and *parameters*, the word "all" has the obvious meaning. Note: there must be spaces separating the ":" that divides the *device* list from the *parameter* list.

The "old form" (with "-v") prints the data in a older, more verbose pre-spice3f format.

5.3.46. Showmod*: List model parameter values

General Form

`showmod` *models [* **:** *parameters* `]` *, ...*

The **showmod** command operates like the **show** command (above) but prints out model parameter values. The applicable forms for *models* are a single letter specifying the device type letter, "letter:subckt:", "modelname", ":subckt:modelname", or "letter:subcircuit:modelname".

5.3.47. Source: Read a Spice3 input file

General Form

`source` *file*

For Spice3: Read the Spice3 input file **file**. **Nutmeg** and Spice3 commands may be included in the file, and must be enclosed between the lines *.control* and *.endc*. These commands are executed immediately after the circuit is loaded, so a control line of *ac* ... works the same as the corresponding the circuit. The exception to this rule is the file *.spiceinit*. Thus, a Spice3 command script must begin with a blank line and then with a *.control* line. Also, any line beginning with the characters *# is considered a control line. This makes it possible to imbed commands in Spice3 input files that are ignored by earlier versions of Spice2

For Nutmeg: Reads commands from the file **filename.** Lines beginning with the character * are considered comments and ignored.

5.3.48. Status*: Display breakpoint information

General Form

`status`

Display all of the traces and breakpoints currently in effect.

5.3.49. Step*: Run a fixed number of timepoints

General Form

```
step [number]
```

Iterate **number** times, or once, and then stop.

5.3.50. Stop*: Set a breakpoint

General Form

`stop [ after n] [ when` *value cond value* `] ...`

Set a breakpoint. The argument **after n** means stop after **n** iteration number **n**, and the argument **when** *value cond value* means stop when the first *value* is in the given relation with the second *value*, the possible relations being

eq	or	=	equal to
ne	or	<>	not equal to
gt	or	>	greater than
lt	or	<	less than
ge	or	>=	greater than or equal to
le	or	<=	less than or equal to

IO redirection is disabled for the **stop** command, since the relational operations conflict with it (it doesn't produce any output anyway). The *value*s above may be node names in the running circuit, or real values. If more than one condition is given, e.g. **stop after 4 when v(1) > 4 when v(2) < 2**, the conjunction of the conditions is implied.

5.3.51. Tf*: Run a Transfer Function analysis

General Form

`tf` *output_node input_source*

The **tf** command performs a transfer function analysis, returning the transfer function (output/input), output resistance, and input resistance between the given output node and the given input source. The analysis assumes a small-signal DC (slowly varying) input.

5.3.52. Trace*: Trace nodes

General Form

```
trace [ node ...]
```

For every step of an analysis, the value of the node is printed. Several traces may be active at once. Tracing is not applicable for all analyses. To remove a trace, use the **delete** command.

5.3.53. Tran*: Perform a transient analysis

General Form

`tran` *Tstep Tstop* `[` *Tstart* `[` *Tmax* `] ] [ UIC ]`

Perform a transient analysis. See the previous sections of this manual for more details.

5.3.54. Transpose: Swap the elements in a multi-dimensional data set

General Form

`transpose` *vector vector ...*

This command transposes a multidimensional vector. No analysis in Spice3 produces multidimensional vectors, although the DC transfer curve may be run with two varying sources. You must use the "reshape" command to reform the one-dimensional vectors into two dimensional vectors. In addition, the default scale is incorrect for plotting. You must plot versus the vector corresponding to the second source, but you must also refer only to the first segment of this second source vector. For example (circuit to produce the tranfer characteristic of a MOS transistor):

```
spice3 > dc vgg 0 5 1 vdd 0 5 1
spice3 > plot i(vdd)
spice3 > reshape all [6,6]
spice3 > transpose i(vdd) v(drain)
spice3 > plot i(vdd) vs v(drain)[0]
```

5.3.55. Unalias: Retract an alias

General Form

```
unalias [word ...]
```

Removes any aliases present for the **word**s.

5.3.56. Undefine: Retract a definition

General Form

```
undefine function
```

Definitions for the named user-defined functions are deleted.

5.3.57. Unset: Clear a variable

General Form

```
unset [word ...]
```

Clear the value of the specified variable(s) (*word*).

5.3.58. Version: Print the version of Spice

General Form

```
version [version id]
```

Print out the version of **nutmeg** that is running. If there are arguments, it checks to make sure that the arguments match the current version of SPICE. (This is mainly used as a **Command:** line in rawfiles.)

5.3.59. Where: Identify troublesome node or device

General Form

```
where
```

When performing a transient or operating point analysis, the name of the last node or device to cause non-convergence is saved. The **where** command prints out this information so that you can examine the circuit and either correct the problem or make a bug report. You may do this either in the middle of a run or after the simulator has given up on the analysis. For transient simulation, the **iplot** command can be used to monitor the progress of the analysis. When the analysis slows down severly or hangs, interrupt the simulator (with control-C) and issue the **where** command. Note that only one node or device is printed; there may be problems with more than one node.

5.3.60. Write: Write data to a file

General Form

```
write [file] [exprs]
```

Writes out the expressions to **file.**

First vectors are grouped together by plots, and written out as such (i.e, if the expression list contained three vectors from one plot and two from another, then two plots are written, one with three vectors and one with two). Additionally, if the scale for a vector isn't present, it is automatically written out as well.

The default format is ascii, but this can be changed with the **set filetype** command. The default filename is **rawspice.raw**, or the argument to the **-r** flag on the command line, if there was one, and the default expression list is **all**.

5.3.61. Xgraph: use the xgraph(1) program for plotting.

General Form

```
xgraph file [exprs] [plot options]
```

The spice3/nutmeg xgraph command plots data like the **plot** command but via **xgraph**, a popular X11 plotting program.

If *file* is either "temp" or "tmp" a temporary file is used to hold the data while being plotted. For available plot options, see the **plot** command. All options except for polar or smith plots are supported.

5.4. CONTROL STRUCTURES

5.4.1. While — End

General Form

```
while condition
      statement
      ...
end
```

While *condition*, an arbitrary algebraic expression, is true, execute the statements.

5.4.2. Repeat — End

General Form

```
repeat [number]
      statement
      ...
end
```

Execute the statements *number* times, or forever if no argument is given.

5.4.3. Dowhile — End

General Form

```
dowhile condition
      statement
      ...
end
```

The same as **while**, except that the *condition* is tested after the statements are executed.

5.4.4. Foreach — End

General Form

```
foreach var value ...
      statement
      ...
end
```

The statements are executed once for each of the *value*s, each time with the variable *var* set to the current one. (*var* can be accessed by the $*var* notation – see below).

5.4.5. If — Then — Else

General Form

```
if condition
      statement
      ...
else
      statement
      ...
end
```

If the *condition* is non-zero then the first set of statements are executed, otherwise the second set. The **else** and the second set of statements may be omitted.

5.4.6. Label

General Form

```
label word
```

If a statement of the form `goto` *word* is encountered, control is transferred to this point, otherwise this is a no-op.

5.4.7. Goto

General Form

`goto` *word*

If a statement of the form `label` *word* is present in the block or an enclosing block, control is transferred there. Note that if the label is at the top level, it *must* be before the `goto` statement (i.e, a forward `goto` may occur only within a block).

5.4.8. Continue

General Form

`continue`

If there is a **while, dowhile,** or **foreach** block enclosing this statement, control passes to the test, or in the case of **foreach**, the next value is taken. Otherwise an error results.

5.4.9. Break

General Form

`break`

If there is a **while, dowhile,** or **foreach** block enclosing this statement, control passes out of the block. Otherwise an error results.

Of course, control structures may be nested. When a block is entered and the input is the terminal, the prompt becomes a number of >'s corresponding to the number of blocks the user has entered. The current control structures may be examined with the debugging command **cdump.**

5.5. VARIABLES

The operation of both **Nutmeg** and **Spice3** may be affected by setting variables with the "set" command. In addition to the variables mentioned below, the **set** command in **Spice3** also affect the behaviour of the simulator via the options previously described under the section on ".OPTIONS".

The variables meaningful to **nutmeg** which may be altered by the **set** command are:

diff_abstol	The absolute tolerance used by the **diff** command.
appendwrite	Append to the file when a **write** command is issued, if one already exists.
color*N*	These variables determine the colors used, if **X** is being run on a color display. *N* may be between 0 and 15. Color 0 is the background, color 1 is the grid and text color, and colors 2 through 15 are used in order for vectors plotted. The value of the **color** variables should be names of colors, which may be found in the file **/usr/lib/rgb.txt.**
combplot	Plot vectors by drawing a vertical line from each point to the X-axis, as opposed to joining the points. Note that this option is subsumed in the *plottype* option, below.
cpdebug	Print *cshpar* debugging information (must be complied with the -DCPDEBUG flag). Unsupported in the current release.
debug	If set then a lot of debugging information is printed (must be compiled with the -DFTEDEBUG flag). Unsupported in the current release.
device	The name (/dev/tty??) of the graphics device. If this variable isn't set then the user's terminal is used. To do plotting on another monitor you probably have to set both the **device** and **term** variables. (If **device** is set to the name of a file, **nutmeg** dumps the graphics control codes into this file -- this is useful for saving plots.)
echo	Print out each command before it is executed.
filetype	This can be either **ascii** or **binary,** and determines what format are. The default is **ascii.**
fourgridsize	How many points to use for interpolating into when doing fourier analysis.
gridsize	If this variable is set to an integer, this number is used as the number of equally spaced points to use for the Y-axis when plotting. Otherwise the current scale is used (which may not have equally spaced points). If the current scale isn't strictly monotonic, then this option has no effect.
hcopydev	If this is set, when the **hardcopy** command is run the resulting file is automatically printed on the printer named **hcopydev** with the command *lpr -P***hcopydev** *-g* **file**.
hcopyfont	This variable specifies the font name for hardcopy output plots. The value is device dependent.
hcopyfontsize	This is a scaling factor for the font used in hardcopy plots.

hcopydevtype	This variable specifies the type of the printer output to use in the **hardcopy** command. If hcopydevtype is not set, plot (5) format is assumed. The standard distribution currently recognizes **postscript** as an alternative output format. When used in conjunction with **hcopydev**, **hcopydevtype** should specify a format supported by the printer.
height	The length of the page for **asciiplot** and **print col**.
history	The number of events to save in the history list.
lprplot5	This is a printf(3s) style format string used to specify the command to use for sending plot(5)-style plots to a printer or plotter. The first parameter supplied is the printer name, the second parameter supplied is a file name containing the plot. Both parameters are strings. It is trivial to cause Spice3 to abort by supplying a unreasonable format string.
lprps	This is a printf(3s) style format string used to specify the command to use for sending PostScript plots to a printer or plotter. The first parameter supplied is the printer name, the second parameter supplied is a file name containing the plot. Both parameters are strings. It is trivial to cause Spice3 to abort by supplying a unreasonable format string.
nfreqs	The number of frequencies to compute in the **fourier** command. (Defaults to 10.)
nobreak	Don't have **asciiplot** and **print col** break between pages.
noasciiplotvalue	Don't print the first vector plotted to the left when doing an **asciiplot**.
noclobber	Don't overwrite existing files when doing IO redirection.
noglob	Don't expand the global characters '*', '?', '[', and ']'. This is the default.
nogrid	Don't plot a grid when graphing curves (but do label the axes).
nomoremode	If **nomoremode** is not set, whenever a large amount of data is being printed to the screen (e.g, the **print** or **asciiplot** commands), the output is stopped every screenful and continues when a carriage return is typed. If **nomoremode** is set then data scrolls off the screen without check.
nonomatch	If **noglob** is unset and a global expression cannot be matched, use the global characters literally instead of complaining.
nosort	Don't have **display** sort the variable names.
noprintscale	Don't print the scale in the leftmost column when a **print col** command is given.

numdgt The number of digits to print when printing tables of data (**fourier, print col**). The default precision is 6 digits. On the VAX, approximately 16 decimal digits are available using double precision, so **numdgt** should not be more than 16. If the number is negative, one fewer digit is printed to ensure constant widths in tables.

plottype This should be one of **normal**, **comb**, or **point:***chars*. **normal**, the default, causes points to be plotted as parts of connected lines. **comb** causes a comb plot to be done (see the description of the **combplot** variable above). **point** causes each point to be plotted separately – the **chars** are a list of characters that are used for each vector plotted. If they are omitted then a default set is used.

polydegree The degree of the polynomial that the **plot** command should fit to the data. If *polydegree* is N, then **nutmeg** fits a degree N polynomial to every set of N points and draw 10 intermediate points in between each endpoint. If the points aren't monotonic, then it tries rotating the curve and reducing the degree until a fit is achieved.

polysteps The number of points to interpolate between every pair of points available when doing curve fitting. The default is 10.

program The name of the current program (*argv[0]*).

prompt The prompt, with the character '!' replaced by the current event number.

rawfile The default name for rawfiles created.

diff_reltol The relative tolerance used by the **diff** command.

remote_shell Overrides the name used for generating rspice runs (default is "rsh").

rhost The machine to use for remote SPICE-3 runs, instead of the default one (see the description of the **rspice** command, below).

rprogram The name of the remote program to use in the **rspice** command.

slowplot Stop between each graph plotted and wait for the user to type return before continuing.

sourcepath A list of the directories to search when a **source** command is given. The default is the current directory and the standard spice library (*/usr/local/lib/spice*, or whatever **LIBPATH** is #defined to in the Spice3 source.

spicepath The program to use for the **aspice** command. The default is /cad/bin/spice.

term The *mfb* name of the current terminal.

units If this is **degrees**, then all the trig functions will use degrees instead of radians.

unixcom	If a command isn't defined, try to execute it as a UNIX command. Setting this option has the effect of giving a **rehash** command, below. This is useful for people who want to use **nutmeg** as a login shell.
verbose	Be verbose. This is midway between **echo** and **debug / cpdebug**.
diff_vntol	The absolute voltage tolerance used by the **diff** command.
width	The width of the page for **asciiplot** and **print col**.
x11linearcs	Some X11 implementations have poor arc drawing. If you set this option, Spice3 will plot using an approximation to the curve using straight lines.
xbrushheight	The height of the brush to use if **X** is being run.
xbrushwidth	The width of the brush to use if **X** is being run.
xfont	The name of the X font to use when plotting data and entering labels. The plot may not look good if this is a variable-width font.

There are several **set** variables that Spice3 uses but Nutmeg does not. They are:

editor	The editor to use for the **edit** command.
modelcard	The name of the model card (normally **.model**).
noaskquit	Do not check to make sure that there are no circuits suspended and no plots unsaved. Normally Spice3 warns the user when he tries to quit if this is the case.
nobjthack	Assume that BJTs have 4 nodes.
noparse	Don't attempt to parse input files when they are read in (useful for debugging). Of course, they cannot be run if they are not parsed.
nosubckt	Don't expand subcircuits.
renumber	Renumber input lines when an input file has **.include**'s.
subend	The card to end subcircuits (normally **.ends**).
subinvoke	The prefix to invoke subcircuits (normally **x**).
substart	The card to begin subcircuits (normally **.subckt**).

5.6. MISCELLANEOUS

If there are subcircuits in the input file, Spice3 expands instances of them. A subcircuit is delimited by the cards **.subckt** and **.ends,** or whatever the value of the variables **substart** and **subend** is, respectively. An instance of a subcircuit is created by specifying a device with type 'x' – the device line is written

```
xname node1 node2 ... subcktname
```

where the nodes are the node names that replace the formal parameters on the **.subckt** line. All nodes that are not formal parameters are prepended with the name given to the instance and a ':', as are the names of the devices in the subcircuit. If there are several nested subcircuits, node and device names look like **subckt1:subckt2:...:name**. If the variable **subinvoke** is set, then it is used as the prefix that specifies instances of subcircuits, instead of 'x'.

Nutmeg occasionally checks to see if it is getting close to running out of space, and warns the user if this is the case. (This is more likely to be useful with the SPICE front end.)

C-shell type quoting with "" and '', and backquote substitution may be used. Within single quotes, no further substitution (like history substitution) is done, and within double quotes, the words are kept together but further substitution is done. Any text between backquotes is replaced by the result of executing the text as a command to the shell.

Tenex-style ('set filec' in the 4.3 C-shell) command, filename, and keyword completion is possible: If EOF (control-D) is typed after the first character on the line, a list of the commands or possible arguments is printed (If it is alone on the line it exits **nutmeg**). If escape is typed, then **nutmeg** trys to complete what the user has already typed. To get a list of all commands, the user should type <space> ^D.

The values of variables may be used in commands by writing **$varname** where the value of the variable is to appear. The special variables *$$* and *$<* refer to the process ID of the program and a line of input which is read from the terminal when the variable is evaluated, respectively. If a variable has a name of the form **$&word**, then **word** is considered a vector (see above), and its value is taken to be the value of the variable. If *$foo* is a valid variable, and is of type **list**, then the expression *$foo[low-high]* represents a range of elements. Either the upper index or the lower may be left out, and the reverse of a list may be obtained with *$foo[len-0]*. Also, the notation *$?foo* evaluates to 1 if the variable *foo* is defined, 0 otherwise, and *$#foo* evaluates to the number of elements in *foo* if it is a list, 1 if it is a number or string, and 0 if it is a boolean variable.

History substitutions, similar to C-shell history substitutions, are also available – see the C-shell manual page for all of the details.

The characters ~, {, and } have the same effects as they do in the C-Shell, i.e., home directory and alternative expansion. It is possible to use the wildcard characters *, ?, [, and] also, but only if you **unset noglob** first. This makes them rather useless for typing algebraic expressions, so you should **set noglob** again after you are done with wildcard expansion. Note that the pattern **[^abc]** matchs all characters *except* **a, b,** *and* **c.**

IO redirection is available – the symbols **>, >>, >&, >>&,** and **<** have the same effects as in the C-shell.

You may type multiple commands on one line, separated by semicolons.

If you want to use a different **mfbcap** file than the default (usually **~cad/lib/mfbcap**), you have to set the environment variable **SPICE_MFBCAP** before you start **nutmeg** or **spice**. The **-m** option and the **mfbcap** variable no longer work.

If **X** is being used, the cursor may be positioned at any point on the screen when the window is up and characters typed at the keyboard are added to the window at that point. The window may then be sent to a printer using the **xpr(1)** program.

Nutmeg can be run under VAX/VMS, as well as several other operating systems. Some features like command completion, expansion of *, ?, and [], backquote substitution, the shell command, and so forth do not work.

On some systems you have to respond to the *-more-* prompt during plot with a carriage return instead of any key as you can do on UNIX.

5.7. BUGS

The label entry facilities are primitive. You must be careful to type slowly when entering labels -- **nutmeg** checks for input once every second, and can get confused if characters arrive faster.

If you redefine colors after creating a plot window with X, and then cause the window to be redrawn, it does not redraw in the correct colors.

When defining aliases like

```
alias pdb plot db( '!:1' - '!:2' )
```

you must be careful to quote the argument list substitutions in this manner. If you quote the whole argument it might not work properly.

In a user-defined function, the arguments cannot be part of a name that uses the *plot.vec* syntax. For example:

```
define check(v(1)) cos(tran1.v(1))
```

does **not** work.

If you type **plot all all**, or otherwise use a wildcard reference for one plot twice in a command, the effect is unpredictable.

The **asciiplot** command doesn't deal with log scales or the **delta** keywords.

Often the names of terminals recognized by **MFB** are different from those in /etc/termcap. Thus you may have to reset your terminal type with the command

set term = termname

where **termname** is the name in the **mfbcap** file.

The **hardcopy** command is useless on VMS and other systems without the **plot** command, unless the user has a program that understands *plot(5)* format.

Spice3 recognizes all the notations used in SPICE2 **.plot** cards, and translates **vp(1)** into **ph(v(1))**, and so forth. However, if there are spaces in these names it won't work. Hence **v(1, 2)** and **(-.5, .5)** aren't recognized.

BJTs can have either 3 or 4 nodes, which makes it difficult for the subcircuit expansion routines to decide what to rename. If the fourth parameter has been declared as a model name, then it is assumed that there are 3 nodes, otherwise it is considered a node. To disable this, you can set the variable "nobjthack" which forces BJTs to have 4 nodes (for the purposes of subcircuit expansion, at least).

The **@name[param]** notation might not work with **trace, iplot,** etc. yet.

The first line of a command file (except for the *.spiceinit* file) should be a comment, otherwise SPICE may create an empty circuit.

Files specified on the command line are read before **.spiceinit** is read.

6. BIBLIOGRAPHY

[1] A. Vladimirescu and S. Liu, *The Simulation of MOS Integrated Circuits Using SPICE2*
ERL Memo No. ERL M80/7, Electronics Research Laboratory
University of California, Berkeley, October 1980

[2] T. Sakurai and A. R. Newton, *A Simple MOSFET Model for Circuit Analysis and its application to CMOS gate delay analysis and series-connected MOSFET Structure*
ERL Memo No. ERL M90/19, Electronics Research Laboratory,
University of California, Berkeley, March 1990

[3] B. J. Sheu, D. L. Scharfetter, and P. K. Ko, *SPICE2 Implementation of BSIM*
ERL Memo No. ERL M85/42, Electronics Research Laboratory
University of California, Berkeley, May 1985

[4] J. R. Pierret, *A MOS Parameter Extraction Program for the BSIM Model*
ERL Memo Nos. ERL M84/99 and M84/100, Electronics Research Laboratory
University of California, Berkeley, November 1984

[5] Min-Chie Jeng, *Design and Modeling of Deep-Submicrometer MOSFETSs*
ERL Memo Nos. ERL M90/90, Electronics Research Laboratory
University of California, Berkeley, October 1990

[6] Soyeon Park, *Analysis and SPICE implementation of High Temperature Effects on MOSFET*,
Master's thesis, University of California, Berkeley, December 1986.

[7] Clement Szeto, *Simulator of Temperature Effects in MOSFETs (STEIM)*,
Master's thesis, University of California, Berkeley, May 1988.

[8] J.S. Roychowdhury and D.O. Pederson, *Efficient Transient Simulation of Lossy Interconnect*,
Proc. of the 28th ACM/IEEE Design Automation Conference, June 17-21 1991, San Francisco

[9] A. E. Parker and D. J. Skellern, *An Improved FET Model for Computer Simulators*,
IEEE Trans CAD, vol. 9, no. 5, pp. 551-553, May 1990.

[10] R. Saleh and A. Yang, Editors, *Simulation and Modeling*,
IEEE Circuits and Devices, vol. 8, no. 3, pp. 7-8 and 49, May 1992

[11] H.Statz et al., *GaAs FET Device and Circuit Simulation in SPICE*,
IEEE Transactions on Electron Devices, V34, Number 2, February, 1987 pp160-169.

A. APPENDIX A: EXAMPLE CIRCUITS

A.1. Circuit 1: Differential Pair

The following deck determines the dc operating point of a simple differential pair. In addition, the ac small-signal response is computed over the frequency range 1Hz to 100MEGHz.

```
SIMPLE DIFFERENTIAL PAIR
VCC  7  0    12
VEE  8  0    -12
VIN  1  0    AC 1
RS1  1  2    1K
RS2  6  0    1K
Q1   3  2  4 MOD1
Q2   5  6  4 MOD1
RC1  7  3    10K
RC2  7  5    10K
RE   4  8    10K
.MODEL MOD1 NPN BF=50 VAF=50 IS=1.E-12 RB=100 CJC=.5PF TF=.6NS
.TF V(5) VIN
.AC DEC 10 1 100MEG
.END
```

A.2. Circuit 2: MOSFET Characterization

The following deck computes the output characteristics of a MOSFET device over the range 0-10V for VDS and 0-5V for VGS.

```
MOS OUTPUT CHARACTERISTICS
.OPTIONS NODE NOPAGE
VDS  3  0
VGS  2  0
M1   1  2  0  0 MOD1 L=4U W=6U AD=10P AS=10P
* VIDS MEASURES ID, WE COULD HAVE USED VDS, BUT ID WOULD BE NEGATIVE
VIDS 3  1
.MODEL MOD1 NMOS VTO=-2 NSUB=1.0E15 UO=550
.DC VDS 0 10 .5 VGS 0 5 1
.END
```

A.3. Circuit 3: RTL Inverter

The following deck determines the dc transfer curve and the transient pulse response of a simple RTL inverter. The input is a pulse from 0 to 5 Volts with delay, rise, and fall times of 2ns and a pulse width of 30ns. The transient interval is 0 to 100ns, with printing to be done every nanosecond.

```
SIMPLE RTL INVERTER
VCC  4  0    5
VIN  1  0    PULSE 0 5 2NS 2NS 2NS 30NS
RB   1  2    10K
Q1   3  2  0 Q1
RC   3  4    1K
.MODEL Q1 NPN BF 20 RB 100 TF .1NS CJC 2PF
.DC VIN 0 5 0.1
.TRAN 1NS 100NS
.END
```

A.4. Circuit 4: Four-Bit Binary Adder

The following deck simulates a four-bit binary adder, using several subcircuits to describe various pieces of the overall circuit.

```
ADDER - 4 BIT ALL-NAND-GATE BINARY ADDER

*** SUBCIRCUIT DEFINITIONS
.SUBCKT NAND 1 2 3 4
*   NODES:  INPUT(2), OUTPUT, VCC
Q1        9  5  1 QMOD
D1CLAMP   0  1    DMOD
Q2        9  5  2 QMOD
D2CLAMP   0  2    DMOD
RB        4  5    4K
R1        4  6    1.6K
Q3        6  9  8 QMOD
R2        8  0    1K
RC        4  7    130
Q4        7  6 10 QMOD
DVBEDROP 10  3    DMOD
Q5        3  8  0 QMOD
.ENDS NAND

.SUBCKT ONEBIT 1 2 3 4 5 6
*   NODES:  INPUT(2), CARRY-IN, OUTPUT, CARRY-OUT, VCC
X1   1  2  7  6   NAND
X2   1  7  8  6   NAND
X3   2  7  9  6   NAND
X4   8  9 10  6   NAND
X5   3 10 11  6   NAND
```

```
X6   3 11 12  6   NAND
X7  10 11 13  6   NAND
X8  12 13  4  6   NAND
X9  11  7  5  6   NAND
.ENDS ONEBIT

.SUBCKT TWOBIT 1 2 3 4 5 6 7 8 9
*   NODES:  INPUT - BIT0(2) / BIT1(2), OUTPUT - BIT0 / BIT1,
*           CARRY-IN, CARRY-OUT, VCC
X1   1  2  7  5 10  9   ONEBIT
X2   3  4 10  6  8  9   ONEBIT
.ENDS TWOBIT

.SUBCKT FOURBIT 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
*   NODES:  INPUT - BIT0(2) / BIT1(2) / BIT2(2) / BIT3(2),
*           OUTPUT - BIT0 / BIT1 / BIT2 / BIT3, CARRY-IN, CARRY-OUT, VCC
X1   1  2  3  4  9 10 13 16 15   TWOBIT
X2   5  6  7  8 11 12 16 14 15   TWOBIT
.ENDS FOURBIT

*** DEFINE NOMINAL CIRCUIT
.MODEL DMOD D
.MODEL QMOD NPN(BF=75 RB=100 CJE=1PF CJC=3PF)
VCC   99  0   DC 5V
VIN1A  1  0   PULSE(0 3 0 10NS 10NS   10NS   50NS)
VIN1B  2  0   PULSE(0 3 0 10NS 10NS   20NS  100NS)
VIN2A  3  0   PULSE(0 3 0 10NS 10NS   40NS  200NS)
VIN2B  4  0   PULSE(0 3 0 10NS 10NS   80NS  400NS)
VIN3A  5  0   PULSE(0 3 0 10NS 10NS  160NS  800NS)
VIN3B  6  0   PULSE(0 3 0 10NS 10NS  320NS 1600NS)
VIN4A  7  0   PULSE(0 3 0 10NS 10NS  640NS 3200NS)
VIN4B  8  0   PULSE(0 3 0 10NS 10NS 1280NS 6400NS)
X1     1  2  3  4  5  6  7  8  9 10 11 12  0 13 99 FOURBIT
RBIT0  9  0   1K
RBIT1 10  0   1K
RBIT2 11  0   1K
RBIT3 12  0   1K
RCOUT 13  0   1K

*** (FOR THOSE WITH MONEY (AND MEMORY) TO BURN)
.TRAN 1NS 6400NS
.END
```

A.5. Circuit 5: Transmission-Line Inverter

The following deck simulates a transmission-line inverter. Two transmission-line elements are required since two propagation modes are excited. In the case of a coaxial line, the first line (T1) models the inner conductor with respect to the shield, and the second line (T2) models the shield with respect to the outside world.

```
TRANSMISSION-LINE INVERTER
V1    1  0          PULSE(0 1 0 0.1N)
R1    1  2          50
X1    2  0  0  4    TLINE
R2    4  0          50

.SUBCKT TLINE 1 2 3 4
T1    1  2  3  4    Z0=50 TD=1.5NS
T2    2  0  4  0    Z0=100 TD=1NS
.ENDS TLINE

.TRAN 0.1NS 20NS
.END
```

B. APPENDIX B: MODEL AND DEVICE PARAMETERS

The following tables summarize the parameters available on each of the devices and models in (note that for some systems with limited memory, output parameters are not available). There are several tables for each type of device supported by . Input parameters to instances and models are parameters that can occur on an instance or model definition line in the form ''keyword=value'' where ''keyword'' is the parameter name as given in the tables. Default input parameters (such as the resistance of a resistor or the capacitance of a capacitor) obviously do not need the keyword specified.

Output parameters are those additional parameters which are available for many types of instances for the output of operating point and debugging information. These parameters are specified as ''@device[keyword]'' and are available for the most recent point computed or, if specified in a ''.save'' statement, for an entire simulation as a normal output vector. Thus, to monitor the gate-to-source capacitance of a MOSFET, a command

```
save @m1[cgs]
```

given before a transient simulation causes the specified capacitance value to be saved at each timepoint, and a subsequent command such as

```
plot @m1[cgs]
```

produces the desired plot. (Note that the **show** command does not use this format).

Some variables are listed as both input and output, and their output simply returns the previously input value, or the default value **after** the simulation has been run. Some parameter are input only because the output system can not handle variables of the given type yet, or the need for them as output variables has not been apparent. Many such input variables are available as output variables in a different format, such as the initial condition vectors that can be retrieved as individual initial condition values. Finally, internally derived values are output only and are provided for debugging and operating point output purposes.

Please note that these tables do **not** provide the detailed information available about the parameters provided in the section on each device and model, but are provided as a quick reference guide.

APPENDIX B: MODEL AND DEVICE PARAMETERS

B.1. URC: Uniform R.C. line

URC — instance parameters (input-output)	
l	Length of transmission line
n	Number of lumps

URC — instance parameters (output-only)	
pos_node	Positive node of URC
neg_node	Negative node of URC
gnd	Ground node of URC

URC — model parameters (input-only)	
urc	Uniform R.C. line model

URC — model parameters (input-output)	
k	Propagation constant
fmax	Maximum frequency of interest
rperl	Resistance per unit length
cperl	Capacitance per unit length
isperl	Saturation current per length
rsperl	Diode resistance per length

B.2. ASRC: Arbitrary Source

ASRC — instance parameters (input-only)	
i	Current source
v	Voltage source

ASRC — instance parameters (output-only)	
i	Current through source
v	Voltage across source
pos_node	Positive Node
neg_node	Negative Node

B.3. BJT: Bipolar Junction Transistor

BJT — instance parameters (input-only)	
ic	Initial condition vector

BJT — instance parameters (input-output)	
off	Device initially off
icvbe	Initial B-E voltage
icvce	Initial C-E voltage
area	Area factor
temp	instance temperature

BJT — instance parameters (output-only)	
colnode	Number of collector node
basenode	Number of base node
emitnode	Number of emitter node
substnode	Number of substrate node
colprimenode	Internal collector node
baseprimenode	Internal base node
emitprimenode	Internal emitter node
ic	Current at collector node
ib	Current at base node
ie	Emitter current
is	Substrate current
vbe	B-E voltage
vbc	B-C voltage
gm	Small signal transconductance
gpi	Small signal input conductance - pi
gmu	Small signal conductance - mu
gx	Conductance from base to internal base
go	Small signal output conductance
geqcb	d(Ibe)/d(Vbc)
gccs	Internal C-S cap. equiv. cond.
geqbx	Internal C-B-base cap. equiv. cond.
cpi	Internal base to emitter capactance
cmu	Internal base to collector capactiance
cbx	Base to collector capacitance
ccs	Collector to substrate capacitance
cqbe	Cap. due to charge storage in B-E jct.
cqbc	Cap. due to charge storage in B-C jct.
cqcs	Cap. due to charge storage in C-S jct.
cqbx	Cap. due to charge storage in B-X jct.

continued

BJT — instance output-only parameters — *continued*	
cexbc	Total Capacitance in B-X junction
qbe	Charge storage B-E junction
qbc	Charge storage B-C junction
qcs	Charge storage C-S junction
qbx	Charge storage B-X junction
p	Power dissipation

BJT — model parameters (input-output)	
npn	NPN type device
pnp	PNP type device
is	Saturation Current
bf	Ideal forward beta
nf	Forward emission coefficient
vaf	Forward Early voltage
va	(null)
ikf	Forward beta roll-off corner current
ik	(null)
ise	B-E leakage saturation current
ne	B-E leakage emission coefficient
br	Ideal reverse beta
nr	Reverse emission coefficient
var	Reverse Early voltage
vb	(null)
ikr	reverse beta roll-off corner current
isc	B-C leakage saturation current
nc	B-C leakage emission coefficient
rb	Zero bias base resistance
irb	Current for base resistance=(rb+rbm)/2
rbm	Minimum base resistance
re	Emitter resistance
rc	Collector resistance
cje	Zero bias B-E depletion capacitance
vje	B-E built in potential
pe	(null)
mje	B-E junction grading coefficient
me	(null)
tf	Ideal forward transit time
xtf	Coefficient for bias dependence of TF
vtf	Voltage giving VBC dependence of TF
itf	High current dependence of TF
ptf	Excess phase
cjc	Zero bias B-C depletion capacitance
vjc	B-C built in potential

continued

BJT — model input-output parameters — *continued*	
pc	(null)
mjc	B-C junction grading coefficient
mc	(null)
xcjc	Fraction of B-C cap to internal base
tr	Ideal reverse transit time
cjs	Zero bias C-S capacitance
ccs	Zero bias C-S capacitance
vjs	Substrate junction built in potential
ps	(null)
mjs	Substrate junction grading coefficient
ms	(null)
xtb	Forward and reverse beta temp. exp.
eg	Energy gap for IS temp. dependency
xti	Temp. exponent for IS
fc	Forward bias junction fit parameter
tnom	Parameter measurement temperature
kf	Flicker Noise Coefficient
af	Flicker Noise Exponent

	BJT — model parameters (output-only)
type	NPN or PNP
invearlyvoltf	Inverse early voltage:forward
invearlyvoltr	Inverse early voltage:reverse
invrollofff	Inverse roll off - forward
invrolloffr	Inverse roll off - reverse
collectorconduct	Collector conductance
emitterconduct	Emitter conductance
transtimevbcfact	Transit time VBC factor
excessphasefactor	Excess phase fact.

B.4. BSIM1: Berkeley Short Channel IGFET Model

BSIM1 — instance parameters (input-only)	
ic	Vector of DS,GS,BS initial voltages

BSIM1 — instance parameters (input-output)	
l	Length
w	Width
ad	Drain area
as	Source area
pd	Drain perimeter
ps	Source perimeter
nrd	Number of squares in drain
nrs	Number of squares in source
off	Device is initially off
vds	Initial D-S voltage
vgs	Initial G-S voltage
vbs	Initial B-S voltage

BSIM1 — model parameters (input-only)	
nmos	Flag to indicate NMOS
pmos	Flag to indicate PMOS

BSIM1 — model parameters (input-output)	
vfb	Flat band voltage
lvfb	Length dependence of vfb
wvfb	Width dependence of vfb
phi	Strong inversion surface potential
lphi	Length dependence of phi
wphi	Width dependence of phi
k1	Bulk effect coefficient 1
lk1	Length dependence of k1
wk1	Width dependence of k1
k2	Bulk effect coefficient 2
lk2	Length dependence of k2
wk2	Width dependence of k2
eta	VDS dependence of threshold voltage
leta	Length dependence of eta
weta	Width dependence of eta
x2e	VBS dependence of eta
lx2e	Length dependence of x2e

continued

BSIM1 — model input-output parameters — *continued*	
wx2e	Width dependence of x2e
x3e	VDS dependence of eta
lx3e	Length dependence of x3e
wx3e	Width dependence of x3e
dl	Channel length reduction in um
dw	Channel width reduction in um
muz	Zero field mobility at VDS=0 VGS=VTH
x2mz	VBS dependence of muz
lx2mz	Length dependence of x2mz
wx2mz	Width dependence of x2mz
mus	Mobility at VDS=VDD VGS=VTH, channel length modulation
lmus	Length dependence of mus
wmus	Width dependence of mus
x2ms	VBS dependence of mus
lx2ms	Length dependence of x2ms
wx2ms	Width dependence of x2ms
x3ms	VDS dependence of mus
lx3ms	Length dependence of x3ms
wx3ms	Width dependence of x3ms
u0	VGS dependence of mobility
lu0	Length dependence of u0
wu0	Width dependence of u0
x2u0	VBS dependence of u0
lx2u0	Length dependence of x2u0
wx2u0	Width dependence of x2u0
u1	VDS depence of mobility, velocity saturation
lu1	Length dependence of u1
wu1	Width dependence of u1
x2u1	VBS depence of u1
lx2u1	Length depence of x2u1
wx2u1	Width depence of x2u1
x3u1	VDS depence of u1
lx3u1	Length dependence of x3u1
wx3u1	Width depence of x3u1
n0	Subthreshold slope
ln0	Length dependence of n0
wn0	Width dependence of n0
nb	VBS dependence of subthreshold slope
lnb	Length dependence of nb
wnb	Width dependence of nb
nd	VDS dependence of subthreshold slope
lnd	Length dependence of nd
wnd	Width dependence of nd

continued

BSIM1 — model input-output parameters — *continued*	
tox	Gate oxide thickness in um
temp	Temperature in degree Celcius
vdd	Supply voltage to specify mus
cgso	Gate source overlap capacitance per unit channel width(m)
cgdo	Gate drain overlap capacitance per unit channel width(m)
cgbo	Gate bulk overlap capacitance per unit channel length(m)
xpart	Flag for channel charge partitioning
rsh	Source drain diffusion sheet resistance in ohm per square
js	Source drain junction saturation current per unit area
pb	Source drain junction built in potential
mj	Source drain bottom junction capacitance grading coefficient
pbsw	Source drain side junction capacitance built in potential
mjsw	Source drain side junction capacitance grading coefficient
cj	Source drain bottom junction capacitance per unit area
cjsw	Source drain side junction capacitance per unit area
wdf	Default width of source drain diffusion in um
dell	Length reduction of source drain diffusion

B.5. BSIM2: Berkeley Short Channel IGFET Model

BSIM2 — instance parameters (input-only)	
ic	Vector of DS,GS,BS initial voltages

BSIM2 — instance parameters (input-output)	
l	Length
w	Width
ad	Drain area
as	Source area
pd	Drain perimeter
ps	Source perimeter
nrd	Number of squares in drain
nrs	Number of squares in source
off	Device is initially off
vds	Initial D-S voltage
vgs	Initial G-S voltage
vbs	Initial B-S voltage

APPENDIX B: MODEL AND DEVICE PARAMETERS

BSIM2 — model parameters (input-only)	
nmos	Flag to indicate NMOS
pmos	Flag to indicate PMOS

BSIM2 — model parameters (input-output)	
vfb	Flat band voltage
lvfb	Length dependence of vfb
wvfb	Width dependence of vfb
phi	Strong inversion surface potential
lphi	Length dependence of phi
wphi	Width dependence of phi
k1	Bulk effect coefficient 1
lk1	Length dependence of k1
wk1	Width dependence of k1
k2	Bulk effect coefficient 2
lk2	Length dependence of k2
wk2	Width dependence of k2
eta0	VDS dependence of threshold voltage at VDD=0
leta0	Length dependence of eta0
weta0	Width dependence of eta0
etab	VBS dependence of eta
letab	Length dependence of etab
wetab	Width dependence of etab
dl	Channel length reduction in um
dw	Channel width reduction in um
mu0	Low-field mobility, at VDS=0 VGS=VTH
mu0b	VBS dependence of low-field mobility
lmu0b	Length dependence of mu0b
wmu0b	Width dependence of mu0b
mus0	Mobility at VDS=VDD VGS=VTH
lmus0	Length dependence of mus0
wmus0	Width dependence of mus
musb	VBS dependence of mus
lmusb	Length dependence of musb
wmusb	Width dependence of musb
mu20	VDS dependence of mu in tanh term
lmu20	Length dependence of mu20
wmu20	Width dependence of mu20
mu2b	VBS dependence of mu2
lmu2b	Length dependence of mu2b
wmu2b	Width dependence of mu2b
mu2g	VGS dependence of mu2

continued

BSIM2 — model input-output parameters — *continued*	
lmu2g	Length dependence of mu2g
wmu2g	Width dependence of mu2g
mu30	VDS dependence of mu in linear term
lmu30	Length dependence of mu30
wmu30	Width dependence of mu30
mu3b	VBS dependence of mu3
lmu3b	Length dependence of mu3b
wmu3b	Width dependence of mu3b
mu3g	VGS dependence of mu3
lmu3g	Length dependence of mu3g
wmu3g	Width dependence of mu3g
mu40	VDS dependence of mu in linear term
lmu40	Length dependence of mu40
wmu40	Width dependence of mu40
mu4b	VBS dependence of mu4
lmu4b	Length dependence of mu4b
wmu4b	Width dependence of mu4b
mu4g	VGS dependence of mu4
lmu4g	Length dependence of mu4g
wmu4g	Width dependence of mu4g
ua0	Linear VGS dependence of mobility
lua0	Length dependence of ua0
wua0	Width dependence of ua0
uab	VBS dependence of ua
luab	Length dependence of uab
wuab	Width dependence of uab
ub0	Quadratic VGS dependence of mobility
lub0	Length dependence of ub0
wub0	Width dependence of ub0
ubb	VBS dependence of ub
lubb	Length dependence of ubb
wubb	Width dependence of ubb
u10	VDS depence of mobility
lu10	Length dependence of u10
wu10	Width dependence of u10
u1b	VBS depence of u1
lu1b	Length depence of u1b
wu1b	Width depence of u1b
u1d	VDS depence of u1
lu1d	Length depence of u1d
wu1d	Width depence of u1d
n0	Subthreshold slope at VDS=0 VBS=0
ln0	Length dependence of n0

continued

BSIM2 — model input-output parameters — *continued*	
wn0	Width dependence of n0
nb	VBS dependence of n
lnb	Length dependence of nb
wnb	Width dependence of nb
nd	VDS dependence of n
lnd	Length dependence of nd
wnd	Width dependence of nd
vof0	Threshold voltage offset AT VDS=0 VBS=0
lvof0	Length dependence of vof0
wvof0	Width dependence of vof0
vofb	VBS dependence of vof
lvofb	Length dependence of vofb
wvofb	Width dependence of vofb
vofd	VDS dependence of vof
lvofd	Length dependence of vofd
wvofd	Width dependence of vofd
ai0	Pre-factor of hot-electron effect.
lai0	Length dependence of ai0
wai0	Width dependence of ai0
aib	VBS dependence of ai
laib	Length dependence of aib
waib	Width dependence of aib
bi0	Exponential factor of hot-electron effect.
lbi0	Length dependence of bi0
wbi0	Width dependence of bi0
bib	VBS dependence of bi
lbib	Length dependence of bib
wbib	Width dependence of bib
vghigh	Upper bound of the cubic spline function.
lvghigh	Length dependence of vghigh
wvghigh	Width dependence of vghigh
vglow	Lower bound of the cubic spline function.
lvglow	Length dependence of vglow
wvglow	Width dependence of vglow
tox	Gate oxide thickness in um
temp	Temperature in degree Celcius
vdd	Maximum Vds
vgg	Maximum Vgs
vbb	Maximum Vbs
cgso	Gate source overlap capacitance per unit channel width(m)
cgdo	Gate drain overlap capacitance per unit channel width(m)
cgbo	Gate bulk overlap capacitance per unit channel length(m)
xpart	Flag for channel charge partitioning

continued

BSIM2 — model input-output parameters — *continued*	
rsh	Source drain diffusion sheet resistance in ohm per square
js	Source drain junction saturation current per unit area
pb	Source drain junction built in potential
mj	Source drain bottom junction capacitance grading coefficient
pbsw	Source drain side junction capacitance built in potential
mjsw	Source drain side junction capacitance grading coefficient
cj	Source drain bottom junction capacitance per unit area
cjsw	Source drain side junction capacitance per unit area
wdf	Default width of source drain diffusion in um
dell	Length reduction of source drain diffusion

B.6. Capacitor: Fixed capacitor

Capacitor — instance parameters (input-output)	
capacitance	Device capacitance
ic	Initial capacitor voltage
w	Device width
l	Device length

Capacitor — instance parameters (output-only)	
i	Device current
p	Instantaneous device power

Capacitor — model parameters (input-only)	
c	Capacitor model

Capacitor — model parameters (input-output)	
cj	Bottom Capacitance per area
cjsw	Sidewall capacitance per meter
defw	Default width
narrow	width correction factor

B.7. CCCS: Current controlled current source

CCCS — instance parameters (input-output)	
gain	Gain of source
control	Name of controlling source

CCCS — instance parameters (output-only)	
neg_node	Negative node of source
pos_node	Positive node of source
i	CCCS output current
v	CCCS voltage at output
p	CCCS power

B.8. CCVS: Linear current controlled current source

CCVS — instance parameters (input-output)	
gain	Transresistance (gain)
control	Controlling voltage source

CCVS — instance parameters (output-only)	
pos_node	Positive node of source
neg_node	Negative node of source
i	CCVS output current
v	CCVS output voltage
p	CCVS power

B.9. CSwitch: Current controlled ideal switch

CSwitch — instance parameters (input-only)	
on	Initially closed
off	Initially open

CSwitch — instance parameters (input-output)	
control	Name of controlling source

CSwitch — instance parameters (output-only)	
pos_node	Positive node of switch
neg_node	Negative node of switch
i	Switch current
p	Instantaneous power

CSwitch — model parameters (input-output)	
csw	Current controlled switch model
it	Threshold current
ih	Hysterisis current
ron	Closed resistance
roff	Open resistance

CSwitch — model parameters (output-only)	
gon	Closed conductance
goff	Open conductance

B.10. Diode: Junction Diode model

Diode — instance parameters (input-output)	
off	Initially off
temp	Instance temperature
ic	Initial device voltage
area	Area factor

Diode — instance parameters (output-only)	
vd	Diode voltage
id	Diode current
c	Diode current
gd	Diode conductance
cd	Diode capacitance
charge	Diode capacitor charge
capcur	Diode capacitor current
p	Diode power

Diode — model parameters (input-only)	
d	Diode model

Diode — model parameters (input-output)	
is	Saturation current
tnom	Parameter measurement temperature
rs	Ohmic resistance
n	Emission Coefficient
tt	Transit Time
cjo	Junction capacitance
cj0	(null)
vj	Junction potential
m	Grading coefficient
eg	Activation energy
xti	Saturation current temperature exp.
kf	flicker noise coefficient
af	flicker noise exponent
fc	Forward bias junction fit parameter
bv	Reverse breakdown voltage
ibv	Current at reverse breakdown voltage

Diode — model parameters (output-only)	
cond	Ohmic conductance

B.11. Inductor: Inductors

Inductor — instance parameters (input-output)	
inductance	Inductance of inductor
ic	Initial current through inductor

Inductor — instance parameters (output-only)	
flux	Flux through inductor
v	Terminal voltage of inductor
volt	
i	Current through the inductor
current	
p	instantaneous power dissipated by the inductor

B.12. mutual: Mutual inductors

mutual — instance parameters (input-output)	
k	Mutual inductance
coefficient	(null)
inductor1	First coupled inductor
inductor2	Second coupled inductor

B.13. Isource: Independent current source

Isource — instance parameters (input-only)	
pulse	Pulse description
sine	Sinusoidal source description
sin	Sinusoidal source description
exp	Exponential source description
pwl	Piecewise linear description
sffm	single freq. FM description
ac	AC magnitude,phase vector
c	Current through current source
distof1	f1 input for distortion
distof2	f2 input for distortion

Isource — instance parameters (input-output)	
dc	DC value of source
acmag	AC magnitude
acphase	AC phase

Isource — instance parameters (output-only)	
neg_node	Negative node of source
pos_node	Positive node of source
acreal	AC real part
acimag	AC imaginary part
function	Function of the source
order	Order of the source function
coeffs	Coefficients of the source
v	Voltage across the supply
p	Power supplied by the source

B.14. JFET: Junction Field effect transistor

JFET — instance parameters (input-output)	
off	Device initially off
ic	Initial VDS,VGS vector
area	Area factor
ic-vds	Initial D-S voltage
ic-vgs	Initial G-S volrage
temp	Instance temperature

JFET — instance parameters (output-only)	
drain-node	Number of drain node
gate-node	Number of gate node
source-node	Number of source node
drain-prime-node	Internal drain node
source-prime-node	Internal source node
vgs	Voltage G-S
vgd	Voltage G-D
ig	Current at gate node
id	Current at drain node
is	Source current
igd	Current G-D
gm	Transconductance
gds	Conductance D-S
ggs	Conductance G-S
ggd	Conductance G-D
qgs	Charge storage G-S junction
qgd	Charge storage G-D junction
cqgs	Capacitance due to charge storage G-S junction
cqgd	Capacitance due to charge storage G-D junction
p	Power dissipated by the JFET

JFET — model parameters (input-output)	
njf	N type JFET model
pjf	P type JFET model
vt0	Threshold voltage
vto	(null)
beta	Transconductance parameter
lambda	Channel length modulation param.
rd	Drain ohmic resistance
rs	Source ohmic resistance
cgs	G-S junction capactance

continued

JFET — model input-output parameters — *continued*	
cgd	G-D junction cap
pb	Gate junction potential
is	Gate junction saturation current
fc	Forward bias junction fit parm.
b	Doping tail parameter
tnom	parameter measurement temperature
kf	Flicker Noise Coefficient
af	Flicker Noise Exponent

JFET — model parameters (output-only)	
type	N-type or P-type JFET model
gd	Drain conductance
gs	Source conductance

B.15. LTRA: Lossy transmission line

LTRA — instance parameters (input-only)	
ic	Initial condition vector:v1,i1,v2,i2

LTRA — instance parameters (input-output)	
v1	Initial voltage at end 1
v2	Initial voltage at end 2
i1	Initial current at end 1
i2	Initial current at end 2

LTRA — instance parameters (output-only)	
pos_node1	Positive node of end 1 of t-line
neg_node1	Negative node of end 1 of t.line
pos_node2	Positive node of end 2 of t-line
neg_node2	Negative node of end 2 of t-line

LTRA — model parameters (input-output)	
ltra	LTRA model
r	Resistance per metre
l	Inductance per metre
g	(null)
c	Capacitance per metre
len	length of line
nocontrol	No timestep control
steplimit	always limit timestep to 0.8*(delay of line)

continued

LTRA — model input-output parameters — *continued*	
nosteplimit	don't always limit timestep to 0.8*(delay of line)
lininterp	use linear interpolation
quadinterp	use quadratic interpolation
mixedinterp	use linear interpolation if quadratic results look unacceptable
truncnr	use N-R iterations for step calculation in LTRAtrunc
truncdontcut	don't limit timestep to keep impulse response calculation errors low
compactrel	special reltol for straight line checking
compactabs	special abstol for straight line checking

LTRA — model parameters (output-only)	
rel	Rel. rate of change of deriv. for bkpt
abs	Abs. rate of change of deriv. for bkpt

B.16. MES: GaAs MESFET model

MES — instance parameters (input-output)	
area	Area factor
icvds	Initial D-S voltage
icvgs	Initial G-S voltage

MES — instance parameters (output-only)	
off	Device initially off
dnode	Number of drain node
gnode	Number of gate node
snode	Number of source node
dprimenode	Number of internal drain node
sprimenode	Number of internal source node
vgs	Gate-Source voltage
vgd	Gate-Drain voltage
cg	Gate capacitance
cd	Drain capacitance
cgd	Gate-Drain capacitance
gm	Transconductance
gds	Drain-Source conductance
ggs	Gate-Source conductance
ggd	Gate-Drain conductance
cqgs	Capacitance due to gate-source charge storage
cqgd	Capacitance due to gate-drain charge storage
qgs	Gate-Source charge storage
qgd	Gate-Drain charge storage
is	Source current

continued

APPENDIX B: MODEL AND DEVICE PARAMETERS

MES — instance output-only parameters — *continued*	
p	Power dissipated by the mesfet

MES — model parameters (input-only)	
nmf	N type MESfet model
pmf	P type MESfet model

MES — model parameters (input-output)	
vt0	Pinch-off voltage
vto	(null)
alpha	Saturation voltage parameter
beta	Transconductance parameter
lambda	Channel length modulation parm.
b	Doping tail extending parameter
rd	Drain ohmic resistance
rs	Source ohmic resistance
cgs	G-S junction capacitance
cgd	G-D junction capacitance
pb	Gate junction potential
is	Junction saturation current
fc	Forward bias junction fit parm.
kf	Flicker noise coefficient
af	Flicker noise exponent

MES — model parameters (output-only)	
type	N-type or P-type MESfet model
gd	Drain conductance
gs	Source conductance
depl_cap	Depletion capacitance
vcrit	Critical voltage

B.17. Mos1: Level 1 MOSfet model with Meyer capacitance model

Mos1 — instance parameters (input-only)	
off	Device initially off
ic	Vector of D-S, G-S, B-S voltages

Mos1 — instance parameters (input-output)	
l	Length
w	Width
ad	Drain area
as	Source area
pd	Drain perimeter
ps	Source perimeter
nrd	Drain squares
nrs	Source squares
icvds	Initial D-S voltage
icvgs	Initial G-S voltage
icvbs	Initial B-S voltage
temp	Instance temperature

Mos1 — instance parameters (output-only)	
id	Drain current
is	Source current
ig	Gate current
ib	Bulk current
ibd	B-D junction current
ibs	B-S junction current
vgs	Gate-Source voltage
vds	Drain-Source voltage
vbs	Bulk-Source voltage
vbd	Bulk-Drain voltage
dnode	Number of the drain node
gnode	Number of the gate node
snode	Number of the source node
bnode	Number of the node
dnodeprime	Number of int. drain node
snodeprime	Number of int. source node
von	
vdsat	Saturation drain voltage
sourcevcrit	Critical source voltage
drainvcrit	Critical drain voltage
rs	Source resistance

continued

Mos1 — instance output-only parameters — *continued*	
sourceconductance	Conductance of source
rd	Drain conductance
drainconductance	Conductance of drain
gm	Transconductance
gds	Drain-Source conductance
gmb	Bulk-Source transconductance
gmbs	
gbd	Bulk-Drain conductance
gbs	Bulk-Source conductance
cbd	Bulk-Drain capacitance
cbs	Bulk-Source capacitance
cgs	Gate-Source capacitance
cgd	Gate-Drain capacitance
cgb	Gate-Bulk capacitance
cqgs	Capacitance due to gate-source charge storage
cqgd	Capacitance due to gate-drain charge storage
cqgb	Capacitance due to gate-bulk charge storage
cqbd	Capacitance due to bulk-drain charge storage
cqbs	Capacitance due to bulk-source charge storage
cbd0	Zero-Bias B-D junction capacitance
cbdsw0	
cbs0	Zero-Bias B-S junction capacitance
cbssw0	
qgs	Gate-Source charge storage
qgd	Gate-Drain charge storage
qgb	Gate-Bulk charge storage
qbd	Bulk-Drain charge storage
qbs	Bulk-Source charge storage
p	Instaneous power

Mos1 — model parameters (input-only)	
nmos	N type MOSfet model
pmos	P type MOSfet model

Mos1 — model parameters (input-output)	
vto	Threshold voltage
vt0	(null)
kp	Transconductance parameter
gamma	Bulk threshold parameter
phi	Surface potential
lambda	Channel length modulation
rd	Drain ohmic resistance

continued

Mos1 — model input-output parameters — *continued*	
rs	Source ohmic resistance
cbd	B-D junction capacitance
cbs	B-S junction capacitance
is	Bulk junction sat. current
pb	Bulk junction potential
cgso	Gate-source overlap cap.
cgdo	Gate-drain overlap cap.
cgbo	Gate-bulk overlap cap.
rsh	Sheet resistance
cj	Bottom junction cap per area
mj	Bottom grading coefficient
cjsw	Side junction cap per area
mjsw	Side grading coefficient
js	Bulk jct. sat. current density
tox	Oxide thickness
ld	Lateral diffusion
u0	Surface mobility
uo	(null)
fc	Forward bias jct. fit parm.
nsub	Substrate doping
tpg	Gate type
nss	Surface state density
tnom	Parameter measurement temperature
kf	Flicker noise coefficient
af	Flicker noise exponent

Mos1 — model parameters (output-only)	
type	N-channel or P-channel MOS

B.18. Mos2: Level 2 MOSfet model with Meyer capacitance model

Mos2 — instance parameters (input-only)	
off	Device initially off
ic	Vector of D-S, G-S, B-S voltages

Mos2 — instance parameters (input-output)	
l	Length
w	Width
ad	Drain area
as	Source area
pd	Drain perimeter
ps	Source perimeter
nrd	Drain squares
nrs	Source squares
icvds	Initial D-S voltage
icvgs	Initial G-S voltage
icvbs	Initial B-S voltage
temp	Instance operating temperature

Mos2 — instance parameters (output-only)	
id	Drain current
cd	
ibd	B-D junction current
ibs	B-S junction current
is	Source current
ig	Gate current
ib	Bulk current
vgs	Gate-Source voltage
vds	Drain-Source voltage
vbs	Bulk-Source voltage
vbd	Bulk-Drain voltage
dnode	Number of drain node
gnode	Number of gate node
snode	Number of source node
bnode	Number of bulk node
dnodeprime	Number of internal drain node
snodeprime	Number of internal source node
von	
vdsat	Saturation drain voltage
sourcevcrit	Critical source voltage
drainvcrit	Critical drain voltage

continued

Mos2 — instance output-only parameters — *continued*	
rs	Source resistance
sourceconductance	Source conductance
rd	Drain resistance
drainconductance	Drain conductance
gm	Transconductance
gds	Drain-Source conductance
gmb	Bulk-Source transconductance
gmbs	
gbd	Bulk-Drain conductance
gbs	Bulk-Source conductance
cbd	Bulk-Drain capacitance
cbs	Bulk-Source capacitance
cgs	Gate-Source capacitance
cgd	Gate-Drain capacitance
cgb	Gate-Bulk capacitance
cbd0	Zero-Bias B-D junction capacitance
cbdsw0	
cbs0	Zero-Bias B-S junction capacitance
cbssw0	
cqgs	Capacitance due to gate-source charge storage
cqgd	Capacitance due to gate-drain charge storage
cqgb	Capacitance due to gate-bulk charge storage
cqbd	Capacitance due to bulk-drain charge storage
cqbs	Capacitance due to bulk-source charge storage
qgs	Gate-Source charge storage
qgd	Gate-Drain charge storage
qgb	Gate-Bulk charge storage
qbd	Bulk-Drain charge storage
qbs	Bulk-Source charge storage
p	Instantaneous power

Mos2 — model parameters (input-only)	
nmos	N type MOSfet model
pmos	P type MOSfet model

APPENDIX B: MODEL AND DEVICE PARAMETERS

Mos2 — model parameters (input-output)	
vto	Threshold voltage
vt0	(null)
kp	Transconductance parameter
gamma	Bulk threshold parameter
phi	Surface potential
lambda	Channel length modulation
rd	Drain ohmic resistance
rs	Source ohmic resistance
cbd	B-D junction capacitance
cbs	B-S junction capacitance
is	Bulk junction sat. current
pb	Bulk junction potential
cgso	Gate-source overlap cap.
cgdo	Gate-drain overlap cap.
cgbo	Gate-bulk overlap cap.
rsh	Sheet resistance
cj	Bottom junction cap per area
mj	Bottom grading coefficient
cjsw	Side junction cap per area
mjsw	Side grading coefficient
js	Bulk jct. sat. current density
tox	Oxide thickness
ld	Lateral diffusion
u0	Surface mobility
uo	(null)
fc	Forward bias jct. fit parm.
nsub	Substrate doping
tpg	Gate type
nss	Surface state density
delta	Width effect on threshold
uexp	Crit. field exp for mob. deg.
ucrit	Crit. field for mob. degradation
vmax	Maximum carrier drift velocity
xj	Junction depth
neff	Total channel charge coeff.
nfs	Fast surface state density
tnom	Parameter measurement temperature
kf	Flicker noise coefficient
af	Flicker noise exponent

Mos2 — model parameters (output-only)	
type	N-channel or P-channel MOS

B.19. Mos3: Level 3 MOSfet model with Meyer capacitance model

Mos3 — instance parameters (input-only)	
off	Device initially off

Mos3 — instance parameters (input-output)	
l	Length
w	Width
ad	Drain area
as	Source area
pd	Drain perimeter
ps	Source perimeter
nrd	Drain squares
nrs	Source squares
icvds	Initial D-S voltage
icvgs	Initial G-S voltage
icvbs	Initial B-S voltage
ic	Vector of D-S, G-S, B-S voltages
temp	Instance operating temperature

Mos3 — instance parameters (output-only)	
id	Drain current
cd	Drain current
ibd	B-D junction current
ibs	B-S junction current
is	Source current
ig	Gate current
ib	Bulk current
vgs	Gate-Source voltage
vds	Drain-Source voltage
vbs	Bulk-Source voltage
vbd	Bulk-Drain voltage
dnode	Number of drain node
gnode	Number of gate node
snode	Number of source node
bnode	Number of bulk node
dnodeprime	Number of internal drain node
snodeprime	Number of internal source node

continued

Mos3 — instance output-only parameters — *continued*	
von	Turn-on voltage
vdsat	Saturation drain voltage
sourcevcrit	Critical source voltage
drainvcrit	Critical drain voltage
rs	Source resistance
sourceconductance	Source conductance
rd	Drain resistance
drainconductance	Drain conductance
gm	Transconductance
gds	Drain-Source conductance
gmb	Bulk-Source transconductance
gmbs	Bulk-Source transconductance
gbd	Bulk-Drain conductance
gbs	Bulk-Source conductance
cbd	Bulk-Drain capacitance
cbs	Bulk-Source capacitance
cgs	Gate-Source capacitance
cgd	Gate-Drain capacitance
cgb	Gate-Bulk capacitance
cqgs	Capacitance due to gate-source charge storage
cqgd	Capacitance due to gate-drain charge storage
cqgb	Capacitance due to gate-bulk charge storage
cqbd	Capacitance due to bulk-drain charge storage
cqbs	Capacitance due to bulk-source charge storage
cbd0	Zero-Bias B-D junction capacitance
cbdsw0	Zero-Bias B-D sidewall capacitance
cbs0	Zero-Bias B-S junction capacitance
cbssw0	Zero-Bias B-S sidewall capacitance
qbs	Bulk-Source charge storage
qgs	Gate-Source charge storage
qgd	Gate-Drain charge storage
qgb	Gate-Bulk charge storage
qbd	Bulk-Drain charge storage
p	Instantaneous power

Mos3 — model parameters (input-only)	
nmos	N type MOSfet model
pmos	P type MOSfet model

APPENDIX B: MODEL AND DEVICE PARAMETERS

Mos3 — model parameters (input-output)	
vto	Threshold voltage
vt0	(null)
kp	Transconductance parameter
gamma	Bulk threshold parameter
phi	Surface potential
rd	Drain ohmic resistance
rs	Source ohmic resistance
cbd	B-D junction capacitance
cbs	B-S junction capacitance
is	Bulk junction sat. current
pb	Bulk junction potential
cgso	Gate-source overlap cap.
cgdo	Gate-drain overlap cap.
cgbo	Gate-bulk overlap cap.
rsh	Sheet resistance
cj	Bottom junction cap per area
mj	Bottom grading coefficient
cjsw	Side junction cap per area
mjsw	Side grading coefficient
js	Bulk jct. sat. current density
tox	Oxide thickness
ld	Lateral diffusion
u0	Surface mobility
uo	(null)
fc	Forward bias jct. fit parm.
nsub	Substrate doping
tpg	Gate type
nss	Surface state density
vmax	Maximum carrier drift velocity
xj	Junction depth
nfs	Fast surface state density
xd	Depletion layer width
alpha	Alpha
eta	Vds dependence of threshold voltage
delta	Width effect on threshold
input_delta	(null)
theta	Vgs dependence on mobility
kappa	Kappa
tnom	Parameter measurement temperature
kf	Flicker noise coefficient
af	Flicker noise exponent

Mos3 — model parameters (output-only)	
type	N-channel or P-channel MOS

B.20. Mos6: Level 6 MOSfet model with Meyer capacitance model

Mos6 — instance parameters (input-only)	
off	Device initially off
ic	Vector of D-S, G-S, B-S voltages

Mos6 — instance parameters (input-output)	
l	Length
w	Width
ad	Drain area
as	Source area
pd	Drain perimeter
ps	Source perimeter
nrd	Drain squares
nrs	Source squares
icvds	Initial D-S voltage
icvgs	Initial G-S voltage
icvbs	Initial B-S voltage
temp	Instance temperature

Mos6 — instance parameters (output-only)	
id	Drain current
cd	Drain current
is	Source current
ig	Gate current
ib	Bulk current
ibs	B-S junction capacitance
ibd	B-D junction capacitance
vgs	Gate-Source voltage
vds	Drain-Source voltage
vbs	Bulk-Source voltage
vbd	Bulk-Drain voltage
dnode	Number of the drain node
gnode	Number of the gate node
snode	Number of the source node
bnode	Number of the node
dnodeprime	Number of int. drain node
snodeprime	Number of int. source node

continued

Mos6 — instance output-only parameters — *continued*	
rs	Source resistance
sourceconductance	Source conductance
rd	Drain resistance
drainconductance	Drain conductance
von	Turn-on voltage
vdsat	Saturation drain voltage
sourcevcrit	Critical source voltage
drainvcrit	Critical drain voltage
gmbs	Bulk-Source transconductance
gm	Transconductance
gds	Drain-Source conductance
gbd	Bulk-Drain conductance
gbs	Bulk-Source conductance
cgs	Gate-Source capacitance
cgd	Gate-Drain capacitance
cgb	Gate-Bulk capacitance
cbd	Bulk-Drain capacitance
cbs	Bulk-Source capacitance
cbd0	Zero-Bias B-D junction capacitance
cbdsw0	
cbs0	Zero-Bias B-S junction capacitance
cbssw0	
cqgs	Capacitance due to gate-source charge storage
cqgd	Capacitance due to gate-drain charge storage
cqgb	Capacitance due to gate-bulk charge storage
cqbd	Capacitance due to bulk-drain charge storage
cqbs	Capacitance due to bulk-source charge storage
qgs	Gate-Source charge storage
qgd	Gate-Drain charge storage
qgb	Gate-Bulk charge storage
qbd	Bulk-Drain charge storage
qbs	Bulk-Source charge storage
p	Instaneous power

Mos6 — model parameters (input-only)	
nmos	N type MOSfet model
pmos	P type MOSfet model

APPENDIX B: MODEL AND DEVICE PARAMETERS

Mos6 — model parameters (input-output)	
vto	Threshold voltage
vt0	(null)
kv	Saturation voltage factor
nv	Saturation voltage coeff.
kc	Saturation current factor
nc	Saturation current coeff.
nvth	Threshold voltage coeff.
ps	Sat. current modification par.
gamma	Bulk threshold parameter
gamma1	Bulk threshold parameter 1
sigma	Static feedback effect par.
phi	Surface potential
lambda	Channel length modulation param.
lambda0	Channel length modulation param. 0
lambda1	Channel length modulation param. 1
rd	Drain ohmic resistance
rs	Source ohmic resistance
cbd	B-D junction capacitance
cbs	B-S junction capacitance
is	Bulk junction sat. current
pb	Bulk junction potential
cgso	Gate-source overlap cap.
cgdo	Gate-drain overlap cap.
cgbo	Gate-bulk overlap cap.
rsh	Sheet resistance
cj	Bottom junction cap per area
mj	Bottom grading coefficient
cjsw	Side junction cap per area
mjsw	Side grading coefficient
js	Bulk jct. sat. current density
ld	Lateral diffusion
tox	Oxide thickness
u0	Surface mobility
uo	(null)
fc	Forward bias jct. fit parm.
tpg	Gate type
nsub	Substrate doping
nss	Surface state density
tnom	Parameter measurement temperature

Mos6 — model parameters (output-only)	
type	N-channel or P-channel MOS

B.21. Resistor: Simple linear resistor

Resistor — instance parameters (input-output)	
resistance	Resistance
temp	Instance operating temperature
l	Length
w	Width

Resistor — instance parameters (output-only)	
i	Current
p	Power

Resistor — model parameters (input-only)	
r	Device is a resistor model

Resistor — model parameters (input-output)	
rsh	Sheet resistance
narrow	Narrowing of resistor
tc1	First order temp. coefficient
tc2	Second order temp. coefficient
defw	Default device width
tnom	Parameter measurement temperature

B.22. Switch: Ideal voltage controlled switch

Switch — instance parameters (input-only)	
on	Switch initially closed
off	Switch initially open

Switch — instance parameters (input-output)	
pos_node	Positive node of switch
neg_node	Negative node of switch

Switch — instance parameters (output-only)	
cont_p_node	Positive contr. node of switch
cont_n_node	Positive contr. node of switch
i	Switch current
p	Switch power

Switch — model parameters (input-output)	
sw	Switch model
vt	Threshold voltage
vh	Hysteresis voltage
ron	Resistance when closed
roff	Resistance when open

Switch — model parameters (output-only)	
gon	Conductance when closed
goff	Conductance when open

B.23. Tranline: Lossless transmission line

Tranline — instance parameters (input-only)	
ic	Initial condition vector:v1,i1,v2,i2

Tranline — instance parameters (input-output)	
z0	Characteristic impedance
zo	(null)
f	Frequency
td	Transmission delay
nl	Normalized length at frequency given
v1	Initial voltage at end 1
v2	Initial voltage at end 2
i1	Initial current at end 1
i2	Initial current at end 2

Tranline — instance parameters (output-only)	
rel	Rel. rate of change of deriv. for bkpt
abs	Abs. rate of change of deriv. for bkpt
pos_node1	Positive node of end 1 of t. line
neg_node1	Negative node of end 1 of t. line
pos_node2	Positive node of end 2 of t. line
neg_node2	Negative node of end 2 of t. line
delays	Delayed values of excitation

B.24. VCCS: Voltage controlled current source

VCCS — instance parameters (input-only)	
ic	Initial condition of controlling source

VCCS — instance parameters (input-output)	
gain	Transconductance of source (gain)

VCCS — instance parameters (output-only)	
pos_node	Positive node of source
neg_node	Negative node of source
cont_p_node	Positive node of contr. source
cont_n_node	Negative node of contr. source
i	Output current
v	Voltage across output
p	Power

B.25. VCVS: Voltage controlled voltage source

VCVS — instance parameters (input-only)	
ic	Initial condition of controlling source

VCVS — instance parameters (input-output)	
gain	Voltage gain

VCVS — instance parameters (output-only)	
pos_node	Positive node of source
neg_node	Negative node of source
cont_p_node	Positive node of contr. source
cont_n_node	Negative node of contr. source
i	Output current
v	Output voltage
p	Power

B.26. Vsource: Independent voltage source

Vsource — instance parameters (input-only)	
pulse	Pulse description
sine	Sinusoidal source description
sin	Sinusoidal source description
exp	Exponential source description
pwl	Piecewise linear description
sffm	Single freq. FM descripton
ac	AC magnitude, phase vector
distof1	f1 input for distortion
distof2	f2 input for distortion

Vsource — instance parameters (input-output)	
dc	D.C. source value
acmag	A.C. Magnitude
acphase	A.C. Phase

Vsource — instance parameters (output-only)	
pos_node	Positive node of source
neg_node	Negative node of source
function	Function of the source
order	Order of the source function
coeffs	Coefficients for the function
acreal	AC real part
acimag	AC imaginary part
i	Voltage source current
p	Instantaneous power