T-Spice User’s Guide

Release 16.3
June 2015

© 1988-2015 Mentor Graphics Corporation
All rights reserved.

This document contains information that is proprietary to Mentor Graphics Corporation. The original recipient of this document may duplicate this document in whole or in part for internal business purposes only, provided that this entire notice appears in all copies. In duplicating any part of this document, the recipient agrees to make every reasonable effort to prevent the unauthorized use and distribution of the proprietary information.
This document is for information and instruction purposes. Mentor Graphics reserves the right to make changes in specifications and other information contained in this publication without prior notice, and the reader should, in all cases, consult Mentor Graphics to determine whether any changes have been made.

The terms and conditions governing the sale and licensing of Mentor Graphics products are set forth in written agreements between Mentor Graphics and its customers. No representation or other affirmation of fact contained in this publication shall be deemed to be a warranty or give rise to any liability of Mentor Graphics whatsoever.

MENTOR GRAPHICS MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

MENTOR GRAPHICS SHALL NOT BE LIABLE FOR ANY INCIDENTAL, INDIRECT, SPECIAL, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING BUT NOT LIMITED TO LOST PROFITS) ARISING OUT OF OR RELATED TO THIS PUBLICATION OR THE INFORMATION CONTAINED IN IT, EVEN IF MENTOR GRAPHICS HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

U.S. GOVERNMENT LICENSE RIGHTS: The software and documentation were developed entirely at private expense and are commercial computer software and commercial computer software documentation within the meaning of the applicable acquisition regulations. Accordingly, pursuant to FAR 48 CFR 12.212 and DFARS 48 CFR 227.7202, use, duplication and disclosure by or for the U.S. Government or a U.S. Government subcontractor is subject solely to the terms and conditions set forth in the license agreement provided with the software, except for provisions which are contrary to applicable mandatory federal laws.

TRADEMARKS: The trademarks, logos and service marks ("Marks") used herein are the property of Mentor Graphics Corporation or other parties. No one is permitted to use these Marks without the prior written consent of Mentor Graphics or the owner of the Mark, as applicable. The use herein of a third-party Mark is not an attempt to indicate Mentor Graphics as a source of a product, but is intended to indicate a product from, or associated with, a particular third party. A current list of Mentor Graphics’ trademarks may be viewed at: www.mentor.com/trademarks.

The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.

Mentor Graphics Corporation
8005 S.W. Boeckman Road, Wilsonville, Oregon 97070-7777
Telephone: 503.685.7000
Toll-Free Telephone: 800.592.2210
Website: www.mentor.com
SupportNet: supportnet.mentor.com/

Send Feedback on Documentation: supportnet.mentor.com/doc_feedback_form
# T-Spice 16 User Guide—Contents

## 1 Getting Started

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation Conventions</td>
<td>14</td>
</tr>
<tr>
<td>Special Fonts</td>
<td>14</td>
</tr>
<tr>
<td>Menu Commands and Dialog Titles</td>
<td>14</td>
</tr>
<tr>
<td>Special Keys</td>
<td>15</td>
</tr>
<tr>
<td>User Interface</td>
<td>15</td>
</tr>
<tr>
<td>Menu Bar</td>
<td>17</td>
</tr>
<tr>
<td>Toolbars</td>
<td>17</td>
</tr>
<tr>
<td>Status Bar</td>
<td>18</td>
</tr>
<tr>
<td>Display Area</td>
<td>18</td>
</tr>
<tr>
<td>Simulation Manager</td>
<td>18</td>
</tr>
<tr>
<td>Simulation Queueing</td>
<td>19</td>
</tr>
<tr>
<td>Simulation Status</td>
<td>19</td>
</tr>
<tr>
<td>Command Tool</td>
<td>20</td>
</tr>
<tr>
<td>Simulating a Design—a Simple Example</td>
<td>20</td>
</tr>
<tr>
<td>Creating a New Input File</td>
<td>20</td>
</tr>
<tr>
<td>Entering the Circuit Description</td>
<td>21</td>
</tr>
<tr>
<td>Device Statements</td>
<td>21</td>
</tr>
<tr>
<td>Simulation Commands</td>
<td>22</td>
</tr>
<tr>
<td>Running the Simulation</td>
<td>24</td>
</tr>
<tr>
<td>Setup Options</td>
<td>25</td>
</tr>
<tr>
<td>Text Editor</td>
<td>26</td>
</tr>
<tr>
<td>Modified Text Files</td>
<td>26</td>
</tr>
<tr>
<td>Text Style</td>
<td>27</td>
</tr>
<tr>
<td>Adding Keywords to a Group</td>
<td>28</td>
</tr>
<tr>
<td>T-Spice Server</td>
<td>29</td>
</tr>
<tr>
<td>External Programs</td>
<td>30</td>
</tr>
<tr>
<td>Command-Line Options</td>
<td>31</td>
</tr>
<tr>
<td>Running T-Spice From the Command-Line</td>
<td>32</td>
</tr>
<tr>
<td>Batch Simulations from the Command Line</td>
<td>32</td>
</tr>
</tbody>
</table>

## 2 Simulation Concepts

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation Algorithms</td>
<td>33</td>
</tr>
<tr>
<td>Kirchoff’s Current Law</td>
<td>33</td>
</tr>
<tr>
<td>DC Analysis</td>
<td>33</td>
</tr>
<tr>
<td>gmin Stepping</td>
<td>34</td>
</tr>
<tr>
<td>Transient Analysis</td>
<td>34</td>
</tr>
<tr>
<td>Trapezoidal Integration Method</td>
<td>35</td>
</tr>
<tr>
<td>Gear’s BDF Method</td>
<td>35</td>
</tr>
<tr>
<td>Small-Signal Analysis</td>
<td>36</td>
</tr>
<tr>
<td>Tolerances</td>
<td>36</td>
</tr>
<tr>
<td>abstol — reltol.</td>
<td>36</td>
</tr>
<tr>
<td>numnd — numnt — numntreduce</td>
<td>37</td>
</tr>
<tr>
<td>chargetol — relchargetol</td>
<td>37</td>
</tr>
<tr>
<td>Parametric Analysis</td>
<td>38</td>
</tr>
<tr>
<td>Monte Carlo Analysis</td>
<td>38</td>
</tr>
<tr>
<td>Error Types and Correction</td>
<td>39</td>
</tr>
<tr>
<td>Syntax Errors</td>
<td>39</td>
</tr>
<tr>
<td>Connectivity Errors</td>
<td>39</td>
</tr>
<tr>
<td>Convergence Errors</td>
<td>40</td>
</tr>
<tr>
<td>Multi-Threaded Processing</td>
<td>40</td>
</tr>
</tbody>
</table>
T-Spice 16 User Guide—Contents

3 Running Simulations

Input Files .................................................. 42
  File Synchronization ........................................ 42
  Opening SPICE Files ....................................... 42
Editing Text Files ........................................... 44
  Comment Delimiters ........................................ 44
  Undo and Redo ............................................. 44
  Search and Replace ....................................... 45
Command Tool ............................................... 48
Setting Simulation Options ................................. 49
Launching a Simulation ...................................... 53
  Simulation Manager ...................................... 54
  Simulation Status ........................................ 57
Encrypting Simulations .................................... 59
  Encryption Commands .................................... 59
    .protect ................................................. 59
    .unprotect ............................................. 60
    .visible ............................................... 60
    .invisible .............................................. 60
    .option password ...................................... 61
The Encryption Process .................................... 61
  1. Protect the Input File .................................. 61
  2. Encrypt the File ....................................... 61
  3. Distribute the Encrypted Files and Password .............. 64
Extent of Data Protection .................................. 64
Encrypting Verilog-A/MS .................................... 65

4 Input Conventions ........................................ 66
Names .................................................................. 66
  Reserved Names ............................................ 66
  Device Names ............................................... 66
  Hierarchical Names ....................................... 67
  Subcircuit Pin Name Aliasing ............................... 67
Comments ...................................................... 67
Line Continuation ............................................. 68
  Comments in Continued Lines ............................... 68
  Expressions and Continued Lines ......................... 69
Numbers and Units ........................................... 69
Parameters ..................................................... 70
Expressions ..................................................... 70
  Built-in Functions ......................................... 71
  Differentiation and Integration functions .................... 73

5 Simulation Commands ....................................... 75
Introduction ................................................... 75
.ac ............................................................. 76
.acmodel ....................................................... 79
.alter .......................................................... 80
.assert ........................................................ 82
.checkpoint ................................................... 84
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>.connect</td>
<td>86</td>
</tr>
<tr>
<td>.data</td>
<td>87</td>
</tr>
<tr>
<td>.dc</td>
<td>89</td>
</tr>
<tr>
<td>.del lib</td>
<td>90</td>
</tr>
<tr>
<td>.end</td>
<td>91</td>
</tr>
<tr>
<td>.enddata</td>
<td>92</td>
</tr>
<tr>
<td>.endi</td>
<td>93</td>
</tr>
<tr>
<td>.ends</td>
<td>94</td>
</tr>
<tr>
<td>.four</td>
<td>95</td>
</tr>
<tr>
<td>.global</td>
<td>97</td>
</tr>
<tr>
<td>.hdl</td>
<td>98</td>
</tr>
<tr>
<td>.ic</td>
<td>100</td>
</tr>
<tr>
<td>.if</td>
<td>101</td>
</tr>
<tr>
<td>.include</td>
<td>103</td>
</tr>
<tr>
<td>.lib</td>
<td>104</td>
</tr>
<tr>
<td>.load</td>
<td>106</td>
</tr>
<tr>
<td>.macro .eom</td>
<td>107</td>
</tr>
<tr>
<td>.malias</td>
<td>108</td>
</tr>
<tr>
<td>.measure</td>
<td>109</td>
</tr>
<tr>
<td>.model</td>
<td>116</td>
</tr>
<tr>
<td>.nodeset</td>
<td>119</td>
</tr>
<tr>
<td>.noise</td>
<td>120</td>
</tr>
<tr>
<td>.op</td>
<td>121</td>
</tr>
<tr>
<td>.optgoal</td>
<td>122</td>
</tr>
<tr>
<td>.optimize</td>
<td>123</td>
</tr>
<tr>
<td>.options</td>
<td>125</td>
</tr>
<tr>
<td>.param</td>
<td>131</td>
</tr>
<tr>
<td>.paramlimits</td>
<td>135</td>
</tr>
<tr>
<td>.power</td>
<td>136</td>
</tr>
<tr>
<td>.print</td>
<td>137</td>
</tr>
<tr>
<td>.probe</td>
<td>152</td>
</tr>
</tbody>
</table>
T-Spice 16 User Guide—Contents

(continued)

6 Device Statements

Introduction ................................................................. 171
BJT (q) ................................................................. 172
Capacitor (c) ............................................................... 174
Coupled Transmission Line (u) ......................................... 177
Current Source (i) .......................................................... 178
Exponential Waveform ...................................................... 178
Pulse Waveform .......................................................... 179
Piecewise Linear Waveform .............................................. 179
Piecewise Linear Waveform File ........................................ 180
Frequency-Modulated Waveform ........................................ 180
Sinusoidal Waveform ...................................................... 180
Vectorized Waveform .................................................... 181
Current-Controlled Current Source (f) ................................. 183
Current-Controlled Voltage Source (h) ................................. 185
Diode (d) ................................................................. 187
Inductor (l) ................................................................. 188
Instance (x) ................................................................. 190
JFET (j) ................................................................. 191
MESFET (z) ................................................................. 192
MOSFET (m) ................................................................. 193
Mutual Inductor (k) ........................................................ 197
Resistor (r) ................................................................. 198
Voltage- or Current-Controlled Switch (s) .............................. 201
Voltage-Controlled Switch ................................................ 201
Current-Controlled Switch ............................................... 202
Transmission Line (t) ........................................................ 203
“Lossless” Transmission Line ............................................ 203
“Lossy” Transmission Line .............................................. 203
Voltage Source (v) .......................................................... 205
Exponential Waveform ...................................................... 205
Pulse Waveform .......................................................... 205
Piecewise Linear Waveform .............................................. 206
Piecewise Linear Waveform File ........................................ 206
Frequency-Modulated Waveform ........................................ 207
Sinusoidal Waveform ...................................................... 207
Vectorized Waveform .................................................... 207
Voltage-Controlled Current Source (g) ................................. 210
Linear Functions .......................................................... 211
Piecewise Linear Functions .............................................. 211
Laplace Functions ........................................................ 212
# 7 Simulation Options

## Accuracy and Convergence Options

<table>
<thead>
<tr>
<th>Option</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>absi</td>
<td>abstol</td>
</tr>
<tr>
<td>absv</td>
<td>vntol</td>
</tr>
<tr>
<td>accurate</td>
<td></td>
</tr>
<tr>
<td>autoconverge</td>
<td></td>
</tr>
<tr>
<td>bypass</td>
<td></td>
</tr>
<tr>
<td>bytol</td>
<td></td>
</tr>
<tr>
<td>cshunt</td>
<td></td>
</tr>
<tr>
<td>dchromotopy</td>
<td></td>
</tr>
<tr>
<td>dcmethod</td>
<td></td>
</tr>
<tr>
<td>dcstep</td>
<td></td>
</tr>
<tr>
<td>extraiter</td>
<td>ations</td>
</tr>
<tr>
<td>fast</td>
<td></td>
</tr>
<tr>
<td>gmin</td>
<td></td>
</tr>
<tr>
<td>gmindc</td>
<td></td>
</tr>
<tr>
<td>gramp</td>
<td></td>
</tr>
<tr>
<td>gshunt</td>
<td></td>
</tr>
<tr>
<td>kcltest</td>
<td></td>
</tr>
<tr>
<td>kvlttest</td>
<td></td>
</tr>
<tr>
<td>maxdcfailures</td>
<td></td>
</tr>
<tr>
<td>mindcratio</td>
<td></td>
</tr>
<tr>
<td>minsrcstep</td>
<td></td>
</tr>
<tr>
<td>numnd</td>
<td>it11</td>
</tr>
<tr>
<td>numndset</td>
<td></td>
</tr>
<tr>
<td>numns</td>
<td>it6</td>
</tr>
<tr>
<td>numnx</td>
<td>it2</td>
</tr>
<tr>
<td>numnxramp</td>
<td></td>
</tr>
<tr>
<td>precise</td>
<td></td>
</tr>
<tr>
<td>recycleOP</td>
<td></td>
</tr>
<tr>
<td>reli</td>
<td>reftol</td>
</tr>
</tbody>
</table>
8 Device Models

### Introduction

Displaying Supported Models and Parameters
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIPOLAR Level/Model Cross Reference</td>
<td>367</td>
</tr>
<tr>
<td>Diode Level/Model Cross Reference</td>
<td>367</td>
</tr>
<tr>
<td>JFET &amp; MESFET Level/Model Cross Reference</td>
<td>368</td>
</tr>
<tr>
<td>MOSFET Level/Model Cross Reference</td>
<td>368</td>
</tr>
<tr>
<td>Philips NXP Model Cross Reference</td>
<td>369</td>
</tr>
<tr>
<td>Model Descriptions</td>
<td>371</td>
</tr>
<tr>
<td><strong>BJT Level 1 (Gummel-Poon)</strong></td>
<td>373</td>
</tr>
<tr>
<td>DC Current Parameters</td>
<td>374</td>
</tr>
<tr>
<td>Base Charge Parameters</td>
<td>374</td>
</tr>
<tr>
<td>Parasitic Resistor Parameters</td>
<td>375</td>
</tr>
<tr>
<td>Parasitic Capacitance Parameters</td>
<td>375</td>
</tr>
<tr>
<td>Junction Capacitance Parameters</td>
<td>375</td>
</tr>
<tr>
<td>Transit Time parameters</td>
<td>376</td>
</tr>
<tr>
<td>Noise Parameters</td>
<td>377</td>
</tr>
<tr>
<td>Temperature Effect Parameters</td>
<td>377</td>
</tr>
<tr>
<td>Geometry Consideration</td>
<td>380</td>
</tr>
<tr>
<td>Current Equations (Level 1)</td>
<td>382</td>
</tr>
<tr>
<td>Substrate Current Equations</td>
<td>384</td>
</tr>
<tr>
<td>Variable Base Resistance Equations</td>
<td>384</td>
</tr>
<tr>
<td>Capacitance Equations</td>
<td>385</td>
</tr>
<tr>
<td>Temperature Dependence</td>
<td>387</td>
</tr>
<tr>
<td><strong>BJT Level 6 (Mextram)</strong></td>
<td>393</td>
</tr>
<tr>
<td><strong>BJT Level 9 (VBIC)</strong></td>
<td>394</td>
</tr>
<tr>
<td><strong>BJT Level 10 (Modella)</strong></td>
<td>399</td>
</tr>
<tr>
<td><strong>Capacitor</strong></td>
<td>400</td>
</tr>
<tr>
<td><strong>Coupled Transmission Line (Level 1)</strong></td>
<td>402</td>
</tr>
<tr>
<td><strong>Diode</strong></td>
<td>403</td>
</tr>
<tr>
<td>Model Selectors</td>
<td>403</td>
</tr>
<tr>
<td>Geometric and Scaling Parameters</td>
<td>403</td>
</tr>
<tr>
<td>DC Parameters</td>
<td>404</td>
</tr>
<tr>
<td>Capacitance Parameters</td>
<td>404</td>
</tr>
<tr>
<td>Noise Parameters</td>
<td>405</td>
</tr>
<tr>
<td>Temperature Parameters</td>
<td>405</td>
</tr>
<tr>
<td>Fowler-Nordheim Model Parameters (level=2)</td>
<td>406</td>
</tr>
<tr>
<td>Level 1 and Level 3</td>
<td>406</td>
</tr>
<tr>
<td>Diffusion Capacitance</td>
<td>408</td>
</tr>
<tr>
<td>Junction Capacitance (Depletion Capacitance)</td>
<td>408</td>
</tr>
<tr>
<td>Metal (Contact Electrode) Capacitance</td>
<td>409</td>
</tr>
<tr>
<td>Polysilicon (Contact Electrode) Capacitance</td>
<td>409</td>
</tr>
<tr>
<td>Level 1</td>
<td>409</td>
</tr>
<tr>
<td>Level 3</td>
<td>410</td>
</tr>
<tr>
<td>Energy Gap</td>
<td>411</td>
</tr>
<tr>
<td>Saturation Current</td>
<td>412</td>
</tr>
<tr>
<td>Breakdown Voltage</td>
<td>412</td>
</tr>
<tr>
<td>Transit Time</td>
<td>412</td>
</tr>
<tr>
<td>Contact Potential</td>
<td>413</td>
</tr>
<tr>
<td>Junction Capacitance</td>
<td>414</td>
</tr>
<tr>
<td>Grading Coefficient</td>
<td>414</td>
</tr>
<tr>
<td>Resistance</td>
<td>414</td>
</tr>
<tr>
<td>Current Equations</td>
<td>415</td>
</tr>
<tr>
<td>Capacitance</td>
<td>415</td>
</tr>
<tr>
<td><strong>JFET</strong></td>
<td>417</td>
</tr>
<tr>
<td>Currents</td>
<td>418</td>
</tr>
<tr>
<td>Charges</td>
<td>419</td>
</tr>
</tbody>
</table>
MESFET ................................................................. 420
  Submodel Selectors ............................................ 420
  DC Parameters ................................................... 420
  Capacitance Parameters ...................................... 421
  Noise Parameters ............................................... 421
  Geometry Parameters .......................................... 422
  Area Calculation Method (ACM) Parameters ................ 422
  Temperature Dependence Parameters ....................... 423
  Currents ....................................................... 424
  Capacitances .................................................. 425
  Temperature Dependence Equations ......................... 426

MOSFET Levels 1/2/3 (Berkeley SPICE 2G6) ................. 430
  Current .......................................................... 432
  Charge ........................................................... 432
  Capacitance .................................................... 432
  Threshold Voltage .............................................. 433
  Subthreshold Region .......................................... 434
  Linear and Saturation Regions ............................... 436
  Second-Order Effects .......................................... 439
  Ward-Dutton Charge Model .................................... 439

MOSFET Levels 4 and 13 (BSIM1) ................................ 447

MOSFET Level 5 (Maher-Mead) .................................. 451

MOSFET Levels 8, 49 and 53 (BSIM3 Revision 3.3) .......... 453
  Model Selectors ................................................ 453
  Basic Model Parameters ...................................... 453
  AC and Capacitance Parameters ............................... 457
  Length and Width Parameters ................................ 457
  Temperature Parameters ....................................... 458
  Bin Description Parameters .................................. 459
  Process Parameters ............................................ 459
  NonQuasi-Static Parameters .................................. 460
  Drain Current .................................................. 460
  Gate Charge .................................................... 461

MOSFET Levels 9 and 50 (Philips MOS 9) ....................... 462
MOSFET Levels 11 and 63 (Philips MOS 11) ...................... 463
MOSFET Levels 14 and 54 (BSIM4 Revision 5) .................. 465
MOSFET Levels 15 and 61 (RPI Amorphous-Si TFT Model) .... 467
MOSFET Levels 16 and 62 (RPI Poly-Si TFT Model, 1.0 and 2.0) 470
MOSFET Level 20 (Philips MOS 20) ............................. 474
MOSFET Level 28 (Extended BSIM1) ............................ 475
  Process Parameters ............................................ 476
  Drain Current .................................................. 476
  Subthreshold Current ......................................... 478

MOSFET Level 30 (Philips MOS 30) ............................. 479
MOSFET Level 31 (Philips MOS 31) ............................. 480
MOSFET Level 40 (Philips MOS 40) ............................. 481
MOSFET Levels 44 and 55 (EKV Revision 2.6) .................. 482
MOSFET Level 47 (BSIM3 Revision 2) ........................... 483
  Drain Current .................................................. 486

MOSFET Level 57 / 70 (BSIM3SOI and BSIM4SOI) ............ 488
  SOI Modes ...................................................... 489
  Body and Temperature Nodes .................................. 490

MOSFET Level 100 (Penn State & Philips PSP Model) .......... 491

Additional MOSFET Parameters .................................. 492
T-Spice 16 User Guide—Contents

Parasitic Resistances .................................................. 494
Parasitic Diodes ......................................................... 494
Effective Areas and Perimeters .................................. 495

Resistor ................................................................. 498
  Resistance ........................................................... 499
  Capacitance ......................................................... 500

Switch ................................................................. 502
  Transmission Line .................................................. 504

9  Small-Signal and Noise Models ................................. 506
  Introduction ......................................................... 506
  Small-Signal Models .............................................. 506
  Noise Models ...................................................... 506

Diode ................................................................. 508
  Small-Signal Model ............................................... 508
  Noise Model ....................................................... 508

BJT Level 1 (Gummel-Poon) ....................................... 509
  Small-Signal Model ............................................... 509
  Gummel-Poon Noise Model ..................................... 510
    Thermal Noise .................................................. 510
    Shot Noise ..................................................... 510
    Flicker Noise .................................................. 510

JFET/MESFET .......................................................... 512
  Small-Signal Model ............................................... 512
  Noise Model ....................................................... 513

MOSFET .............................................................. 514
  Small-Signal Model ............................................... 514
  Noise Model ....................................................... 515

References .......................................................... 517

10  Parametric Analysis ............................................... 518
  Introduction ......................................................... 518
  Output File Formats .............................................. 518

Parameter Sweeps ................................................... 519
  Example 1: Parametric Sweep .................................. 519
    T-Spice Input .................................................. 519
    Output .......................................................... 520
    Waveform ....................................................... 521

Monte Carlo Analysis ............................................... 522
  Example 2: Monte Carlo Analysis .............................. 522
    Input .......................................................... 523
    Output .......................................................... 523
    Waveform ....................................................... 525

Optimization ........................................................ 525
  Defining Optimization Parameters ............................. 526
  Defining Optimization Goals .................................. 526
  Invoking Optimization .......................................... 526
  Example 3: Optimization ........................................ 526
    Output .......................................................... 534

  Example 4: Optimization Using HSPICE-Compatible Commands 535
    Input .......................................................... 535
11 References 537

Credits 574
1 Getting Started

This chapter describes the T-Spice documentation conventions and user interface, and provides a simple tutorial on basic T-Spice usage.

Documentation Conventions

This section contains information about the typographical and stylistic conventions used in this user guide.

Special Fonts

The following references in the text are represented by a bold font:

- Menu and simulation commands (For example: .print tran v(out).)
- Literal user input (For example: Enter 14.5.)
- Program output (For example: S-Edit generates names for the ports on the symbol based on the PAD string.)
- All dialog elements—fields, checkboxes, drop-down menus, titles, etc. (For example: Click Add.)

Freestanding quotations of input examples, file listings, and output messages are represented by a constant-width font—for example:

.ac DEC 5 1MEG 100MEG

Variables for which context-specific substitutions should be made are represented by bold italics—for example, myfile.tdb.

Sequential steps in a tutorial are set off with a checkbox (☑) in the margin.

References to mouse buttons are given in all capitals—for example, MOVE/EDIT. When a key is to be pressed and held while a mouse button is used, the key and button are adjoined by a plus sign (+). For example, Shift+SELECT means that the Shift key is pressed and held while the SELECT mouse button is used.

The terms “left-click,” “right-click,” and “middle-click” all assume default mappings for mouse buttons.

Menu Commands and Dialog Titles

Elements in hierarchical menu paths are separated by a > sign. For example, File > Open means the Open command in the File menu.
Tabs in dialog boxes are set off from the command name or dialog box title by a dash. For example, Setup > Layers—General and Setup Layers—General both refer to the General tab of the Setup Layers dialog.

**Special Keys**

Special keys are represented by the following abbreviations:

<table>
<thead>
<tr>
<th>Key</th>
<th>Abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shift</td>
<td>Shift</td>
</tr>
<tr>
<td>Enter</td>
<td>Enter</td>
</tr>
<tr>
<td>Control</td>
<td>Ctrl</td>
</tr>
<tr>
<td>Alternate</td>
<td>Alt</td>
</tr>
<tr>
<td>Backspace</td>
<td>Back</td>
</tr>
<tr>
<td>Delete</td>
<td>Del</td>
</tr>
<tr>
<td>Escape</td>
<td>Esc</td>
</tr>
<tr>
<td>Insert</td>
<td>Ins</td>
</tr>
<tr>
<td>Tab</td>
<td>Tab</td>
</tr>
<tr>
<td>Home</td>
<td>Home</td>
</tr>
<tr>
<td>End</td>
<td>End</td>
</tr>
<tr>
<td>Page Up</td>
<td>PgUp</td>
</tr>
<tr>
<td>Page Down</td>
<td>PgDn</td>
</tr>
<tr>
<td>Function Keys</td>
<td>F1 F2 F3 ...</td>
</tr>
<tr>
<td>Arrow Keys</td>
<td>⏯️ ⬅️ ⬆️ ⬅️</td>
</tr>
</tbody>
</table>

When keys are to be pressed simultaneously, their abbreviations are adjoined by a plus sign (+). For example, Ctrl+R means that the Ctrl and R keys are pressed at the same time.

When keys are to be pressed in sequence, their abbreviations are separated by a space ( ). For example, Alt+E R means that the Alt and E keys are pressed at the same time and then released, immediately after which the R key is pressed.

Abbreviations for alternative key-presses are separated by a slash (/). For example, Shift↑ / ↓ means that the Shift key can be pressed together with either the up (↑) arrow key or the down (↓) arrow key.

**User Interface**

The T-Spice user interface consists of the following elements:

- Title bar
- Menu bar
- Toolbars
- Status bar
- Simulation Manager
- Simulation Status window

Commands on the View menu toggle display of the last four items.

Text windows in the display area display the contents of input and output files.
Menu Bar

The menu bar contains the names of the T-Spice command menus. The **Edit** and **Window** menus are only available when the active window contains a text file.

 Commands for reading and writing input files
    Commands for text editing
    Commands for displaying interface components
    Commands for operating the simulation engine

 File   Edit  View  Simulation  Setup  Window  Help

 Commands for setting application-level options

 Commands for arranging and showing windows

 Commands for accessing online documents

Toolbars

The Standard toolbar contains buttons with icons representing the most commonly used menu commands.

![Standard Toolbar Diagram]
The Simulation toolbar contains buttons with icons representing simulation commands.

**Status Bar**

When the pointer is positioned over a button in the toolbar, the left side of the status bar contains a short description of the button’s function. When a text window in the display area is active, the status bar provides information useful for editing.

**Display Area**

The *display area* consists of the entire T-Spice window not occupied by the title, menu bar, toolbars, Simulation Manager, Simulation Status window, or status bar. In this area, input files are viewed and edited and simulation status information is displayed.

**Simulation Manager**

The Simulation Manager allows you to control and monitor all T-Spice simulations. Use **View > Simulation Manager** to display or hide the Simulation Manager. You can use the Simulation Manager to monitor multiple simulations at one time. Each simulation occupies one row, and each row has four attributes.
Simulation Queueing

You can submit multiple simulations which will run in the order they were submitted.

You can also run batch simulations from a DOS or Unix command-line using T-Spice’s -batch option, which passes the simulation commands listed in a text batch file to the T-Spice engine. See “Command-Line Options” on page 31 for a description of command-line options.

Simulation Status

The Simulation Status window displays output messages from the T-Spice simulation engine. When you select a simulation in the Simulation Manager, the Simulation Status window displays the results for that simulation.
Command Tool

T-Spice also provides a Command Tool, which presents a categorized listing of T-Spice simulation commands and options. You can use this tool as a guide in composing commands for the input file.

Simulating a Design—a Simple Example

This section illustrates some basic principles of T-Spice operation using a simple example.

Creating a New Input File

Create a new input file by clicking the New File button ( ) or selecting File > New. When you click the New File button, T-Spice opens a new text window with the default filename T-Spice1. T-Spice increments the default name for additional new files, assigning them T-Spice2, T-Spice3, etc.

When you select File > New, T-Spice prompts you to specify the type of file you wish to create. For each file type, T-Spice has settings for color-coding how commands and comments are displayed.
Select the option **SPICE netlist**. An empty text window opens in the display area with the provisional name T-Spice1.

Use the **File > Save As** command (or click ![save](image) in the toolbar).

Type **test2** in the **File name** field of the **Save As** dialog and press **Save**.

T-Spice updates the filename display accordingly.

**Entering the Circuit Description**

**Device Statements**

Type the following text in the **test2.sp** window, pressing **Enter** after each line:

```plaintext
* Test circuit
vv top gnd SIN(0 1 10MEG)
r1 top out 1
r2 out gnd 2
```

The first line is a **comment**. T-Spice always treats the first line of an input file as a comment, whether or not it begins with a comment symbol. The next three lines, beginning with **vv**, **r1**, and **r2**, are **device statements** comprising a SPICE description of the elementary voltage divider schematically represented below.

![Diagram of voltage divider](image)

For information on comments, see “Input Conventions” on page 66. For information on device statements, see “Device Statements” on page 171.
**Simulation Commands**

With the cursor at the beginning of the next blank line in the `test2.sp` netlist, use the Edit > Insert Command command (shortcuts Ctrl+M or the toolbar icon ) to open the Command Tool.

The first category on the left is Analysis. Expand that category and select Transient analysis. T-Spice displays the appropriate parameter options in the right-hand pane. Enter the following values.

- In Maximum Time Step, enter 5n.
- In Simulation Length, enter 500n.

Click Insert Command and then the Close button to bring the netlist window into focus. The full command is placed as a line of text in `test2.sp`

Click at the end of this line, then press Enter to create another blank line.
Reopen the **Command Tool**, select the **Output** category and the **Transient results** command.

- For **Plot** type, select **Voltage**.
- For **Node name** enter **top**, then click the **Add** button.
- Replace the **Node name** value **top** with **out**, and click the **Add** button again.
- Click **Insert Command** button and then the **Close** button to bring the netlist into focus.

**test2.sp** now looks like this:

```
* Test circuit
vv top gnd SIN(0 1 10M)
r1 top out 1
r2 out gnd 2
.tran 5n 500n
.print tran v(top) v(out)
```

The last two lines (beginning with `.tran` and `.print`) are the simulation command sequence, directing T-Spice to perform a transient analysis for 500 nanoseconds with a maximum time step of 5 nanoseconds, and to report the results of the transient analysis for the voltages at nodes **top** and **out**.

(Note that you can also use Tanner’s S-Edit schematic entry tool to incorporate simulation commands directly into a circuit schematic so the commands will be present in the netlist when it is exported.)

Save **test2.sp**. The circuit description is now complete and the simulation can be run.
Running the Simulation

Use the **Simulation > Run Simulation** command (or click ▶️ in the simulation toolbar) to initiate simulation. T-Spice displays the results in the **Simulation Status** window. Note that text in the scrolling portion of the **Simulation Status** window can be copied for pasting into another input window.
At the same time, the output of the simulation is shown graphically in W-Edit. The voltage at node `out` is, correctly, two-thirds of the voltage at node `top`.

Setup Options

The tabs in Setup > Application control global options in T-Spice.
Text Editor

Determines T-Spice behavior in file saving operations. The most recently used settings are kept as defaults.

**Auto-Load**
When checked, T-Spice automatically loads changes whenever a text file is modified outside T-Spice. No warning will be provided.

**Modified text files**
Controls how files modified within T-Spice will be saved prior to simulation operations. Options are:

- **Save all changes**—automatically saves all active windows when one of the above operations is invoked.
- **Prompt to save changes**—T-Spice will display a prompt when there are unsaved changes in the simulation input file. Select **Yes** to save the input file and proceed with simulation. If you select **No**, the **Run Simulation** command is ignored.
- **Don't save changes**—modified files will not be saved and the operation will use the stored version of those files.

**Modified Text Files**

When Auto-Load is disabled, T-Spice will open a checklist of all the files open in the T-Spice text editor that have been modified elsewhere. Files that have also been modified in the text editor will be highlighted.
Similarly, when **Prompt to save changes** is selected, T-Spice will open a checklist of the modified files associated with the operation you are running. For example, you might open a text file in T-Spice, then later overwrite it by reexporting a netlist. In such a case, T-Spice will warn you that the file on disk has changed since you opened it in T-Spice. If you click **Yes**, T-Spice updates the file in memory.

In **Text Files Modified**, the options **Do Not Reload** or **Reload** apply to all the checked files.

![Text Files Modified](image)

**Text Style**

*Keyword groups* are categories of text for which you can set display characteristics in the T-Spice text editor. You can set text file formatting for SPICE netlists, VerilogA netlists, C macro and text files. Each file type has a set of predefined keyword groups that cannot be edited or deleted.

For example, you can define the keyword group “warnings” for SPICE netlist files, and set T-Spice to display warnings in red text against a yellow background.
Use the **Text Styles** tab to view and change keyword settings, and to add or remove your own keyword groups with customized characteristics.

![Text Styles Tab](image)

**File Type**
A drop down list of the file types for which keywords are or can be defined.

**Font**
Allows you to set the typeface (**Face Name**) and point **Size** in which a given keyword group will appear.

**Paragraph**
Allows you to set the increment, in number of spaces, of the **Tab Size** used by the text editor. / Enter a positive integer value to set the increment, in spaces, of the tab spacing the text editor uses.

**Groups**
Displays the keyword groups defined for the active file type.

**Keyword Group**
- Use **Add** to enter the name of a new keyword group.
- Use **Edit** to enter the keywords belonging to a group.
- Use **Remove** to delete an entire keyword group.

See “Adding Keywords to a Group,” below.

**Colors**
Use **Foreground** and **Background** to set the respective colors for a keyword group.

## Adding Keywords to a Group

**Add** under keyword group opens the **Keyword Section** dialog, which allows you name a new keyword group. Once a group is defined, you can use **Edit** under keyword group to enter keywords and specify
whether the case is evaluated (Case sensitive keywords checkbox enabled) when highlighting is applied.

**T-Spice Server**

An internal http server in T-Spice allows other Tanner applications to launch it and run a simulation. This setup tab indicates the status of the server, the port on which it is active and whether to log simulation jobs sent to the server.
When a remote request is sent to launch T-Spice, it initiates a sequence of checks to determine whether T-Spice is installed on the local PC, whether the user has a current license and whether the version of the remote application is compatible with the version of T-Spice.

**Status**
Indicates the status of the internal T-Spice server. Use the **Start**, **Pause** and **Stop** buttons to control the simulation job.

**Network protocol settings**
Enter the port to use to search for the T-Spice executable.

**Logging**
When **Enable logging** is checked, you can enter a name in the **Log file** field and location (using the browse button) for a log file.

### External Programs

This tab is used to configure external executables used by T-Spice.

The **Diff program** that can compare two (or three) text files is specified here.
The Digital simulator path is used for setting the path to the digital simulator executable for Verilog-AMS co-simulations.

Command-Line Options

T-Spice supports command-line options that allow you to alter simulation commands or options without changing the input netlist. There are two ways to use command-line options in T-Spice:

- You can use any command-line option in conjunction with the tspcmd.exe executable to run simulations in a command-line environment, such as DOS or Unix.
- You can use command-line options that are available within the T-Spice user interface by typing them in the Simulation > Run Simulation dialog before you begin a simulation.

Command descriptions follow these conventions:

- Variables to be replaced by actual names, numbers, or expressions are indicated by italics.
- Square brackets [] enclose items that are not required. The brackets should not be typed on the command line.

Some command-line options allow you to specify an included or referenced file. Use the following rules to specify filenames in the command line:

- Specify filenames relative to the working directory, or using a fully qualified pathname. The working directory is the one that contains the main input file.
- Filenames containing spaces must be enclosed in single or double quotes.

T-Spice supports the following command-line options:

<table>
<thead>
<tr>
<th>Option</th>
<th>When on</th>
<th>When off (default)</th>
</tr>
</thead>
<tbody>
<tr>
<td>-batch file</td>
<td>Processes the simulations listed in the text file file in series.</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td>See “Running T-Spice From the Command-Line” on page 32 for file syntax.</td>
<td></td>
</tr>
<tr>
<td>-C</td>
<td>Disable the connectivity check.</td>
<td>Enable the connectivity check.</td>
</tr>
<tr>
<td>-h headerfile</td>
<td>Specify a header file to be processed at the start of T-Spice simulation.</td>
<td>Parse only the contents of the current T-Spice input file.</td>
</tr>
<tr>
<td>-i &quot;TSCommand&quot;</td>
<td>Executes the T-Spice command enclosed in double quotes. Input commands added with -i are parsed after the header file, but before other input files.</td>
<td>—</td>
</tr>
<tr>
<td>-l</td>
<td>Echoes all parsed input lines to the Simulation Status window or stderr output. This is equivalent to setting .options echo=1.</td>
<td>Input lines are not shown in simulation output.</td>
</tr>
</tbody>
</table>
Chapter 1: Getting Started

Command-Line Options

Running T-Spice From the Command-Line

The `tspcmd.exe` executable file allows you to run the T-Spice engine, without a graphical interface, from a command-line environment such as DOS or Unix. To run T-Spice from a command-line, use the following syntax:

```
tspcmd [-aqlCUV] [-batch batchfile] [-P parameter=value] [-o outfile] [-i "command"] netlistfile
```

The `-batch` option is available only in command-line environments; you cannot use this option in the T-Spice Run Simulation dialog.

Batch Simulations from the Command Line

The command-line option:

```
-batch batchfile
```

allows you to pass a list of simulations to T-Spice for serial execution. The `batchfile` specifies a text file that contains a list of simulations and accompanying command-line options. Each simulation and its associated options must be listed on a separate line, using the following syntax:

```
[-acqruCMTUV] [-P parameter=value] [-o outfile] [-i "TScommand"] netlistfile
```
2 Simulation Concepts

Simulation Algorithms

T-Spice is designed to solve a wide variety of circuit problems. Its flexibility is due to robust algorithms which can be optimized by means of user-adjustable parameters. This chapter contains an overview of T-Spice’s algorithms and parameters.

In what follows, when reference is made to an “option” (such as the numnd option) it means that the corresponding quantity is controlled with the .options command. For information on the .options command, see “Simulation Commands” on page 75.

Kirchoff’s Current Law

T-Spice uses Kirchoff’s Current Law (KCL) to solve circuit problems. To T-Spice, a circuit is a set of devices attached to nodes. The circuit's state is represented by the voltages at all the nodes. T-Spice solves for a set of node voltages that satisfies KCL (implying that the sum of the currents flowing into each node is zero).

In order to evaluate whether a set of node voltages is a solution, T-Spice computes and sums all the currents flowing out of each device into the nodes connected to it (its terminals). The relationship between the voltages at a device’s terminals and the currents through the terminals is determined by the device model. For example, the device model for a resistor of resistance $R$ is $i = \Delta v / R$, where $\Delta v$ represents the voltage difference across the device.

DC Analysis

Most T-Spice simulations start with a DC operating point calculation. A circuit’s DC operating point is its steady state, which would in principle be reached after an infinite amount of time if all inputs were held constant. In DC analysis, capacitors are treated as open circuits and inductors as short circuits.

Because many devices, such as transistors, are described by nonlinear device models, the KCL equations that T-Spice solves in DC analysis are nonlinear and must therefore be solved by iteration. On each iteration, T-Spice tries to find a set of node voltages that satisfies KCL more closely than the previous set. When the KCL equations are satisfied “well enough” (the sums of currents into nodes are small enough), the process stops.

The abstol and reltol options determine how closely KCL must be satisfied. The numnd option imposes a limit on the number of iterations. If numnd iterations are reached without a solution being found, then nonconvergence is declared.

T-Spice sometimes uses a technique called source stepping or source ramping to find a circuit’s DC operating point. In source stepping, all voltage and current sources are ramped up from zero to their final values. This allows T-Spice to find the DC operating points of difficult-to-converge circuits. Source stepping is used only in non-converging cases of initial DC operating point calculations, and not during DC analysis sweeps. The smallest source step that T-Spice will take is controlled by the minsrcstep option.
**g**ₘᵢₙ Stepping

Some non-convergence errors can be eliminated by ensuring a sufficient conductance across capacitors. The option \textit{gmindc} specifies a conductance that is added in parallel with all \textit{pn} junctions during DC analysis. T-Spice applies the \textit{gmindc} conductance to various elements as follows:

- diode—conductance is added across the positive/negative terminals.
- BJT—conductance is added across the base/emitter and the base/collector terminals.
- MOSFET—conductance is added across the source/bulk, drain/bulk, and the source/drain terminals.
- MESFET—conductance is added across the source/gate, drain/gate, and source/drain terminals.

The default value for \textit{gmindc} is 10⁻¹².

When a DC operating point non-convergence occurs, T-Spice can begin a \textit{g}ₘᵢₙ stepping algorithm to find the minimum conductance that yields a convergent solution. The \textit{g}ₘᵢₙ stepping algorithm is triggered when a non-convergence occurs and the value of option \textit{gramp} is greater than zero. Together, the options \textit{gmindc} and \textit{gramp} specify a search range for the minimum required conductance, \textit{g}ₘᵢₙ:

\[
\text{gmindc} \leq g_{\text{min}} \leq \text{gmindc} \cdot 10^{\text{gramp}}
\]

T-Spice’s \textit{g}ₘᵢₙ stepping algorithm searches the specified conductance range in two steps. First, T-Spice performs a binary search between \textit{gmindc} and \textit{gmindc} \cdot 10^{\text{gramp}}. T-Spice searches for the smallest value of \textit{g}ₘᵢₙ that results in a converged solution. T-Spice automatically ends the binary search when it reaches a \textit{Δg}ₘᵢₙ that is less than or equal to a factor of 10.

Starting with binary search results, T-Spice then begins reducing the value of \textit{g}ₘᵢₙ by a factor of 10 in each iteration. Once a non-convergence occurs, the previous convergent iteration provides the final solution.

**Transient Analysis**

In transient analysis, T-Spice solves for a circuit’s behavior over some time interval. In this mode, T-Spice takes small time steps, solving for the circuit’s state at each step. At each time step, two approximations are made.

First, a small error — the discretization error — is introduced because T-Spice cannot take infinitely small time steps. The \textit{charge}tol and \textit{relcharge}tol options determine the acceptable limits of discretization error. In general, taking smaller time steps decreases the discretization error, so tightening the tolerances has the effect of higher accuracy at the expense of smaller time steps and therefore longer simulation times and larger output files. The discretization error is also affected by the order of the time integration method used, adjusted with the \textit{maxord} option.

Second, just as in DC analysis, T-Spice solves the nonlinear KCL equations iteratively at each time step. The accuracy is affected by an iteration stopping criterion. The same tolerances as in DC analysis — \textit{abstol} and \textit{reltol} — affect this solution process. The iteration count limit for a transient analysis time step is \textit{numnt}, which is typically much smaller than \textit{numnd}; the previous time step always provides a good initial guess for a transient analysis Newton iteration, so that fewer iterations are typically required than for DC analysis, where a good initial guess is usually not available. Another iteration limit, \textit{numntreduce}, affects time step selection after a successful time step. If T-Spice took less than \textit{numntreduce} iterations to find the solution at a time step, the next time step is adjusted (often increased) according to the discretization error tolerances \textit{charge}tol and \textit{relcharge}tol. But if the number of iterations required is between \textit{numntreduce} and \textit{numnt}, then the time step is always decreased on the next step (even if the step was successful).
As in DC analysis, some non-convergence errors can be avoided by adding a small conductance across capacitors. For transient analysis, the option \texttt{gmin} specifies a conductance that is added in parallel with all \textit{pn} junctions. The default value of \texttt{gmin} is $10^{-12}$.

\textbf{Trapezoidal Integration Method}

T-Spice’s default method for transient analysis uses trapezoidal integration with the \texttt{lvltim=1} delta-voltage time step control algorithm.

The trapezoidal formula calculates the average slope of the present and next time point to approximate the value of the integral of the differential equations used in the time range calculations, as follows in simplified form. The following approximation is used to discretize the differential equation:

\begin{equation}
V_{n+1} = V_n + \frac{h}{2} \left( \frac{dV_{n+1}}{dt} + \frac{dV_n}{dt} \right)
\end{equation}

where

\begin{align*}
V_{n+1} &= \text{present unknown voltage value} \\
V_n &= \text{previous time-point solution} \\
h &= \text{time step length} \\
n &= \text{time interval}
\end{align*}

\textbf{Gear's BDF Method}

T-Spice’s alternate method for transient analysis uses Gear's backward differentiation formulas (BDF). In this method, the time derivative of charge in the KCL equations is replaced by an approximation involving the solution at the last few time points. The first-order BDF method uses only one previous time point, and it is equivalent to the well-known Backward Euler method. In this method, the discretization error is a linear function of the step size. The second order method uses two previous time points, and its discretization error is proportional (for small time step sizes) to the time step size squared. In general, the \textit{k}th order BDF method uses \textit{k} previous time points.

T-Spice uses a variable-step-size, variable-order, and variable-coefficient implementation of the BDF method. T-Spice automatically adjusts the time step size and BDF order (between 1 and 4) to minimize the number of time steps required to meet the given error tolerances. The maximum order used can be adjusted with the \texttt{maxord} option. The variable-coefficient implementation was chosen over the fixed-coefficient and fixed-leading-coefficient methods because it offers the best stability properties, especially with frequently varying time step sizes.

At each time step, the BDF discretization results in a nonlinear system of equations (representing KCL) which is solved iteratively as described above. If the iteration succeeds, the discretization error is examined (by comparison with an explicit predictor). For example, in the order 1 case, the difference between the Forward Euler predictor and the computed BDF (Backward Euler) solution provides a bound on the discretization error. If the error is within the prescribed tolerance (defined by \texttt{chargetol} and \texttt{relchargetol}), the step is accepted, and the error is used to adjust the step size for the next time step. If the error is too large, the time step is rejected and reattempted with a smaller step size. This will produce answers which approach a more stable numerical solution. Gear integration often produces superior results for power circuitry simulations, due to the fact that high frequency ringing and long simulation periods are often encountered.
Small-Signal Analysis

Some of T-Spice’s analysis commands use small-signal models. Small-signal analysis linearizes the KCL equations about an operating point. Subsequent computations are performed on the linearized circuit, which can be solved in one matrix-vector operation. In AC analysis, for example, one matrix-vector solve is done at each frequency point to find the AC solution; no iteration is necessary. The linearized small-signal model is valid only locally, so if the operating point changes, then a new linearized model has to be computed.

Tolerances

T-Spice’s simulation speed and accuracy are controlled by various tolerance values. Computer-based simulators like T-Spice solve circuit equations using finite precision arithmetic. This means that numerical approximations are made at several steps of the solution process. The errors introduced by these approximations in T-Spice are bounded by tolerance settings.

Each approximation is controlled by a relative tolerance $t_{rel}$ and an absolute tolerance $tabs$. In most cases, the relative tolerance is used — the approximation error may not exceed $t_{rel} \times |v|$, where $v$ is the value of the quantity to be approximated. The absolute tolerance is used when the approximated quantity’s value is close to zero; in that case, the error may not exceed $tabs$. In general, the error must be less than the maximum of $t_{rel} \times |v|$ and $tabs$.

$abstol$ — $reltol$

Corresponding to each node in a circuit is an equation which expresses Kirchoff’s Current Law (KCL), according to which the branch currents flowing into the node must be zero. The actual sum of these branch currents at a node is called the “residual current” for that node, and it is a function of all node voltages. In a sense, then, the value of this current at a given node is a measure of how well KCL holds at that node. T-Spice attempts to find a solution (a set of node voltages) that causes KCL to be satisfied at all nodes. The correctness of the solution is measured by the norm of the vector of residual currents at all nodes.

In general, the KCL equations are complicated and nonlinear, and T-Spice solves them by numerical iteration. Each iteration results in an improved approximation of the true solution of the KCL equation, in the sense that the norm of the residual currents decreases with each iteration. However, it is often impossible to make the residual current exactly zero, because of the finite precision arithmetic used. Even if infinite precision arithmetic were available, it might take infinitely many iterations to make the residual current zero. Therefore, the iteration is considered to have converged to a solution when the residual current is within the tolerances defined by $abstol$ and $reltol$. Thus, $abstol$ and $reltol$ control how small the residual currents must be — how far from satisfying KCL the nodes can be — before the system is considered solved.

To be more precise, $reltol$ and $abstol$ are applied as relative and absolute tolerances, as described above, to the residual currents at each node. The tolerance used at a particular node is max($abstol$, $reltol \times imax$), where $imax$ is the largest branch current (in absolute value) flowing into the node in question. The $abstol$ and $reltol$ tolerances are used whenever T-Spice solves the KCL equations for a DC solution or a transient analysis time step.

The default value for $reltol$ is $1 \times 10^{-4}$, or 0.01%. The default value of $abstol$ is 0.5 nanoamps. These values should be reduced for sensitive analog designs.
**numnd — numnt — numntreduce**

`numnd` defines the maximum number of iterations allowed during the solution of the KCL equations during a DC analysis. If, after `numnd` iterations, the `abstol/reltol` tolerances have not been satisfied, the iteration is considered to have failed and nonconvergence is declared. T-Spice then stops the iteration and reacts appropriately:

- On initial DC operating point calculations, source stepping is invoked.
- During DC transfer analysis, the transfer step size is reduced.

During source stepping, the maximum number of iterations for each source step is `numnd/10`. If `numnd/10` iterations are exceeded during source stepping, the source step size is reduced. If the minimum source step size (`minsrcstep` option) is violated, then nonconvergence is declared for the DC operating point computation.

If the initial DC operating point computation fails at the beginning of a transient analysis, T-Spice attempts a powerup simulation. In a powerup simulation, all voltage and current sources are slowly ramped up from zero to their actual values. The default ramp period is 0.1% of the transient simulation final time, and it can be overridden using the `poweruplen` option.

`numnt` determines the maximum number of iterations allowed during the solution of the KCL equations for a transient analysis time step. Another option, `numntreduce`, is used in the time step size after a successful time step. Together, `numnt` and `numntreduce` work as follows. If a time step requires more than `numnt` iterations, the iteration is considered to have failed, and the same time step is reattempted with a smaller step size. If fewer than `numntreduce` iterations are needed for a time step, the next time step is adjusted (often increased) according to the discretization error tolerances (`chargetol` and `relchargetol`). If the number of iterations required is between `numntreduce` and `numnt`, the step size for the next time step is always reduced.

The number of iterations it takes to converge to a circuit solution depends heavily on the circuit to be simulated. Stable circuits with well-defined, steady-state conditions generally require fewer iterations, whereas circuits with poorly defined or unstable steady-state conditions require more iterations. Convergence is also sensitive to the starting point (initial guess) of the iteration. If the starting point is close enough to the operating point, then the iteration will eventually converge (although it might take many iterations in some cases). But if the starting point is not close to the solution, then the iteration may not converge at all. Different starting points (initial guesses) may be specified with the `.nodeset` command. For information on the `.nodeset` command, see “Simulation Commands” on page 75.

**chargetol — relchargetol**

Kirchoff’s Law contains a term that represents the time derivative (rate of change) of charge. In transient simulations, T-Spice must replace this derivative by a divided difference approximation. The approximation becomes more accurate as the time step size decreases. T-Spice chooses its time step size so that the error caused by this approximation remains below another tolerance called the “charge tolerance.” You can specify both an `absolute` and a `relative` charge tolerance with the `chargetol` and `relchargetol` options. By decreasing (or increasing) these tolerances, you can force T-Spice to take smaller (or larger) time steps if you believe that the results of transient simulation are not accurate enough (or too accurate for your time constraints).

The value of `relchargetol` is by default the same as `reltol`, so that `reltol` controls the overall relative simulation accuracy. For example, if you would like 0.001% accuracy, simply set `reltol` to $1 \times 10^{-5}$; this also sets `relchargetol` to $1 \times 10^{-5}$. The `relchargetol` option should be used only to override the general relative tolerance `reltol`. 
In general, the KCL tolerances `abstol` and `reltol`, as well as the charge tolerances `chargetol` and `rechargetol`, trade off speed for accuracy such that tightening these tolerances increases simulation accuracy at the expense of speed. However, if the KCL tolerances are too loose relative to the charge tolerances, the simulator may take small time steps because of numerical noise introduced by residual currents. These residual currents exist only because the KCL equations are not being solved exactly, but they may cause the charge tolerances to be violated, leading to excessively small time steps. If this happens, reducing `abstol` and/or `reltol` will result in larger time steps and faster (as well as more accurate) simulation.

**Parametric Analysis**

Under many circumstances, T-Spice will be required to study the effects on circuit performance of variations in parameter values. For example, parametric analysis can be used to evaluate multidimensional trends in the output over defined ranges of input values, or the sensitivity of circuit behavior to random fluctuations in fabrication conditions.

A large range of parameters may be systematically and automatically varied:

- External parameters (such as temperature)
- Simulation parameters (such as tolerances)
- Device parameters (such as input voltage level)
- Model parameters (such as transistor length)

Three types of parametric analysis are made possible by T-Spice: parameter sweeping, Monte Carlo analysis, and optimization. Discussions of T-Spice syntax and corresponding examples are included for all three analysis types in the chapter titled “Parametric Analysis” on page 518.

In a parameter sweep, a specified parameter is held or initialized at a given value, on the basis of which all analyses requested by the input file are performed, and the results recorded. Then the parameter is incremented by a set amount, and the same analyses are repeated. This cycle is continued while the parameter is incremented through a defined range of values.

Parameter values may be swept linearly — in identical increments, typically through a limited range — or logarithmically — in exponential increments, typically through a range spanning multiple orders of magnitude.

An example illustrating T-Spice input for parameter sweeping is given in “Parameter Sweeps” on page 519.

**Monte Carlo Analysis**

Monte Carlo analysis generates “random” variations in parameter values by drawing them probabilistically from a defined distribution. For each value thus chosen, all analyses requested by the input file are performed, and the results recorded. Monte Carlo analysis is performed using the keyword `sweep`; syntax is described in “.step” (page 158).

Parameter values may be drawn from tunable uniform, Gaussian, or random limit distributions.

For a more detailed description of Monte Carlo analysis, see “Monte Carlo Analysis” on page 522.

If an aspect (or aspects) of the desired circuit performance can be specified quantitatively, T-Spice can search through a multidimensional “space” of parameters — that is, vary several parameters...
systematically and simultaneously — to determine the combination of parameter values that optimizes the specified performance measure (or set of measures).

Each run, using a particular combination of parameter values, produces a new value for each performance measure studied. Optimization is achieved by varying parameters in an attempt to minimize

$$\sum_{i} w_i \frac{(G_i - R_i)^2}{G_i}$$

(2.3)

where \(G_i\) is the goal or desired value of the \(i\)th performance measure; \(R_i\) is the result or actual value of the \(i\)th performance measure, for a particular combination of parameters; and \(w_i\) is the weight or importance assigned to the \(i\)th performance measure relative to the other measures used. The quantity \(w_i(G_i - R_i)/G_i\) is also called the error.

The choice of the next combination of parameters to test after each run, on the basis of the current total error, is the heart of the optimization algorithm. The algorithm employs gradient descent; that is, it attempts to find the steepest (fastest) “path” through the “space” of parameters that will lead to the minimum, by estimating the gradient in various directions.

For a description of T-Spice input needed to set up and invoke optimization, see “Optimization” on page 525. This section also includes a tutorial example illustrating optimization (see “Example 3: Optimization” on page 526).

Error Types and Correction

The highly compressed, text-based nature of the SPICE circuit description language, while rendering it efficient, portable, and flexible, also makes it prone to user errors of various kinds — errors that, while they will seldom crash the program, often result in wrong answers or problems with convergence.

T-Spice makes some corrections automatically, replacing improbable values with default ones or ignoring improper commands and specifications. In other cases, the Simulation Window shows the line number of the input file on which the error was found, or the name of the device whose specification T-Spice could not parse properly, and a brief description of the error. The most common errors can be divided into several categories.

Syntax Errors

- Unsupported commands or statements.
- Wrong spelling of commands, statements, or options.
- Failure to abide by conventions for names, comments, line continuation, numeric formats, unit abbreviations, or expressions.
- Wrong number or order of arguments on a simulation command or device statement.
- Numbers out of range.

Connectivity Errors

- Floating (unconnected) nodes.
- Nodes connected to only one device (except power supply and output nodes).
• Identical nodes referenced by different names.
• Different nodes referenced by identical names.
• Devices referred to that have not been previously defined.

Unless naming inconsistencies interfere with proper connectivity — producing floating nodes or devices with unconnected terminals — they will not be caught by T-Spice; it will simply assume that what is written is what is meant, and will produce misleading, implausible, or impossible results.

**Convergence Errors**

• Wrong metric prefix (order-of-magnitude error). For example, forgetting the \( p \) on a number intended to represent picofarads will not be caught by the syntax or connectivity checks but will probably lead to wildly wrong results.
• Ill-chosen tolerances.

To help prevent errors, use a **schematic export tool** to automate part of the process of writing a circuit description. These translate a **graphical** description of the circuit into a **textual** (SPICE-format) description. As long as the schematic has been drawn consistently (and this is easier to do graphically than textually), connections will be specified and nodes and devices named properly.

In addition, **use comments liberally**. Take advantage of the multiple ways by which comments can be indicated in a T-Spice input file to add structure and clarity to the circuit description. This will make later use of the file, whether for further study or for debugging, much simpler.

To deal with a **syntax error**, note the line number or device name in the input file at which T-Spice found the error, and check the syntax there.

To deal with a **connectivity error**, check the structure of the circuit, as described in the input file, carefully against the original schematic or plan. (This will be easier if T-Spice's own connectivity checker detected the error and provided the appropriate line number or device name.)

To deal with a **convergence error**, check the input file carefully, making sure that any metric prefixes used are plausible. Adjust the tolerances if necessary.

For information on comments, see “Input Conventions” on page 66. For information on syntax specifications, see “Simulation Commands” on page 75 and “Device Statements” on page 171.

**Multi-Threaded Processing**

T-Spice offers multi-threading options to provide accelerated performance on shared memory multi-processor or multiple core computers.

Traditional SPICE simulation runtimes are dominated by two major computational areas: first, the transistor and device model evaluations, in which all terminal currents, charges, and derivatives are computed as a function of the device terminal voltages. Secondly, a sparse linear system is formed and solved, in order to compute iterates to the Newton-Raphson solution of the overall nonlinear algebraic circuit equations.

The model evaluations and linear system solutions comprise well over 95% of the solution time for T-Spice. For medium size circuits, the model evaluation time will be greater than the linear system solution time, but with larger circuits the linear solution component will grow to dominate the overall solution time. The multi-threading feature of T-Spice provides for computational operations to be decomposed and solved in parallel during both stages of processing.
Model evaluation parallel processing is controlled via a single option, `.option threads`. T-Spice requires a circuit to have over 100 devices for multi-threading to be enabled. When the threads option is enabled, T-Spice will automatically decompose the workload into small tasks, and dynamically distribute these tasks to multiple threads for processing. The number of threads and size of the tasks are controlled via internal variables that you can set as shown in the following table. The linear system solution, or matrix factorization and solve, is performed using a multi-threaded sparse linear solver.

The threads option works as follows:

- `threads=0` Disable multi-threading.
- `threads=1` Enable threading, using a number of threads to match the number of processors available, i.e. on dual-core processors use 2 threads, on quad-core use 4 threads, etc.
- `threads=2` Enable threading with two threads.
- `threads=x` Enable threading with `x` number of threads, where `x` is a positive integer.

`Threads=1` is the typical setting you should use to enable multi-threading. However, if you have a quad-core system and want to use the computer for additional work while the simulation is running, then it is advisable to use `threads=2`. 
# 3 Running Simulations

## Input Files

At the heart of T-Spice’s operation is the input file (also known as the circuit description, the netlist, or the input deck). This is a plain text file that contains the device statements and simulation commands, drawn from the SPICE circuit description language, with which T-Spice constructs a model of the circuit to be simulated. Input files can be created and modified with any text editor, though the text editor integrated with T-Spice is ideal as it includes default and fully-customized syntax highlighting.

Input files can be very long and complex, but they do not have to be written from scratch; they can be efficiently created using the export facility of a schematic editor (Tanner’s S-Edit™), or the extraction facility of a layout editor (Tanner’s L-Edit™). In addition, T-Spice includes a “Command Tool” (page 20) that automates error-free SPICE language entry.

Any number of text files can be open at once, each in its own window in the display area. However, only one window can be “active” at any given time, and only an input file displayed in an active window can be edited and simulated.

### File Synchronization

T-Spice has file synchronization features that allow you to control how files are saved and updated.

For input files, you can set an application-level default (auto-load) so that text files open within T-Spice are automatically reloaded if they are modified outside of T-Spice. You can also set a default for text files that have been modified inside T-Spice so that prior to being used they are automatically saved (or not, or with a prompt). Similarly, you can set a global default for output files so that they will always be overwritten without a prompt (see “Setup Options” on page 25).

If the file synchronization feature poses an inconvenience due to a network configuration or other issues, you can disable it by starting T-Spice with the `-y` command-line flag (see “Command-Line Options” on page 31).

Opening SPICE Files

You can use File > New or File > Open to launch the text editor window in T-Spice. There is also a special command File > Open Folder Containing (filename), that opens a browser window directly to the folder containing the result files for the active SPICE file as well as the folder path.
The `.tsim` simulation data file generated by S-Edit or T-Spice is what W-Edit opens and reads.

The SPICE netlist and remaining files required by W-Edit are located in the folder that was specified in S-Edit, in this case `../../SimulationResults/${Design}/${Cell}/${Name}.sp`. The simulation data itself is stored in a separate `.tsdat` file, and all other application settings are saved in various files in the same folder for a given simulation run.
Editing Text Files

When the pointer is in an active text window, it becomes an I-beam. The position in the input file at which text is to be added is marked by a blinking cursor, and the status bar displays the cursor position, editing mode, and other information as described in “Status Bar” on page 18. You can toggle between the two editing modes by pressing the Ins key. In Insert mode, text is added between the characters separated by the blinking cursor. In Overwrite mode, text is added in place of the character to the right of the blinking cursor.

The cursor can be moved by clicking the pointer at the desired location. Sections of text can be selected by clicking and dragging the pointer. Double-clicking selects a word.

Alternatively, all these functions are accessible from the keyboard, as follows.

<table>
<thead>
<tr>
<th>Keys</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>↑ ↓ ← →</td>
<td>Move the cursor in the indicated direction</td>
</tr>
<tr>
<td>Home / End</td>
<td>Move the cursor to the beginning or end of the line</td>
</tr>
<tr>
<td>PgUp / PgDn</td>
<td>Move the cursor up or down one page (scrolling the window with it)</td>
</tr>
<tr>
<td>Shift + ↑ ↓ ← →</td>
<td>Extend the selection in the indicated direction</td>
</tr>
<tr>
<td>Shift + Home / End</td>
<td>Extend the selection to the beginning or end of the line</td>
</tr>
<tr>
<td>Shift + PgUp / PgDn</td>
<td>Extend the selection up or down one page</td>
</tr>
<tr>
<td>Shift + Ctrl + Home / End</td>
<td>Extend the selection to the beginning or end of the file</td>
</tr>
</tbody>
</table>

Text selections can be manipulated with the **Cut**, **Copy**, **Paste**, and **Clear** commands in the **Edit** menu (or, in the case of the first three, by clicking , , and in the toolbar). The **Cut** and **Copy** commands put deleted or duplicated text onto the clipboard, and from there the text can be placed elsewhere with the **Paste** command. The **Clear** command simply deletes text without adding it to the Clipboard.

**Comment Delimiters**

T-Spice allows several different characters to be used as comment delimiters, including the asterisk (*), dollar sign ($), semicolon (;), and C-language style slash (/ or \\). However, the T-Spice text editor will only color-code comment text when:

- An asterisk (*) used as delimiter is placed in the first column of the text editor
- A dollar sign ($) is used as delimiter in any column of the text editor

C-style comments delimited by \* or \* and midline comments delimited by an asterisk will not be color-coded. The T-Spice simulation engine will correctly interpret them as comments, however.

**Undo and Redo**

The **Edit > Undo** command (Ctrl+Z or ) reverses changes made to the text of the input file.

**Undo** reverses the most recent of the editing operations stored in the undo buffer. The previous 100 editing operations are stored in the undo buffer; they are of the following types:

- Typing, including delete and backspace keystrokes.
Edits made with the **Cut**, **Copy**, **Paste**, or **Clear** commands.

Edits made with **Insert Command** (see “Simulation Commands” on page 22).

**Undo** is unavailable under the following circumstances:

- Immediately after T-Spice is launched.
- Immediately after an input file is created or opened.

The **Edit > Redo** command (Ctrl+Y or ) restores changes reversed with a previous **Undo** command. Each of the events stored in the undo buffer can be redone one at a time in reverse order.

**Search and Replace**

The T-Spice text editor supports string and regular expression search and replace operations.

**Edit > Goto Line**

Prompts for a line number, then places the cursor at the beginning of the corresponding line in the active window.

**Edit > Find**

The **Edit > Find** (Ctrl+F) command opens the **Find** dialog which prompts for text to be searched for (the *target string*).

<table>
<thead>
<tr>
<th>Find what</th>
<th>String to be searched for in the text file. Use the button to insert the special character codes used to search for such things as a manual line break, tab break, white space, or the carat (^) character.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Match whole word only</td>
<td>T-Spice searches only for whole words that match the specified search string.</td>
</tr>
<tr>
<td>Match case</td>
<td>Causes T-Spice to find only strings whose case matches that of the search string.</td>
</tr>
</tbody>
</table>
Regular expression

Activates Unix-style regular expression searching in the target string. The Match whole word only option is not available in Regular expression mode. See “Regular Expression Rules,” below for further information.

(Special)

Opens a submenu of special character codes that can be inserted in the target string. These codes are prefixed by the caret character (^) unless they are UNIX-style regular expressions. Options include:

- Manual Line Break
- Tab Break
- White Space
- Caret Character

Direction

- Up—searches backward in the active window.
- Down—searches forward in the active window.

Find Next

Finds the next occurrence of the target string in the active window and closes the dialog.

Replace

Opens the “Edit > Replace” dialog.

Edit > Replace

Prompts for text to be searched for (the target string) and replaced (the replace string) in the active window. The Replace dialog provides one input field and two options additional to the fields in the Find dialog (see “Edit > Find”):
Find Next
Finds the next occurrence of the target string in the active window and closes the dialog.

Replace
Replaces the next instance of the search string with the replace string.

Replace All
Replaces every instance of the search string with the replace string.

Match whole word only
T-Spice searches only for whole words that match the specified search string.

Match case
Causes T-Spice to find only strings whose case matches that of the search string.

Regular expression
Activates Unix-style regular expression searching in the target string. The Match whole word only option is not available in Regular expression mode. See “Regular Expression Rules,” below for further information.

Replace in
- Selection replaces only the instances of the search string in the text that is selected.
- Whole file replaces all instances of the search string in the file.

Regular Expression Rules

The Regular expression option in the Find Item or Replace dialogs causes T-Spice to interpret the search string as a Unix-style regular expression. Instead of interpreting the caret combinations ^t, ^l, and ^w as special sequences, T-Spice replaces them with Unix-style combinations that use the backslash (\) escape character.

The following table lists the rules T-Spice will follow when searching in regular expression mode:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>\n</td>
<td>Line break.</td>
<td></td>
</tr>
<tr>
<td>\t</td>
<td>Tab character.</td>
<td></td>
</tr>
<tr>
<td>^=</td>
<td>Beginning of line.</td>
<td></td>
</tr>
<tr>
<td>$</td>
<td>End of line.</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td>Any character except line break.</td>
<td>p.n matches pin and pan.</td>
</tr>
<tr>
<td>[]</td>
<td>One of the characters enclosed in the brackets.</td>
<td>p[ai]n matches pin and pan but not pun.</td>
</tr>
<tr>
<td>[^set]</td>
<td>Any character not enclosed in square brackets.</td>
<td>p[^i]n matches pan but not pin.</td>
</tr>
<tr>
<td>[set]</td>
<td>A set of characters including any character from the set enclosed in square brackets.</td>
<td>[0-9] matches any digit. [spice] matches any of the characters (s p i c e).</td>
</tr>
<tr>
<td>[set]^</td>
<td>Zero or more occurrences of the set enclosed in square brackets.</td>
<td>[0-9]^1 matches 1 and 11 and 381.</td>
</tr>
<tr>
<td>[set]+</td>
<td>One or more occurrences of the set enclosed in square brackets.</td>
<td>[0-9]+ matches 2 and 4532.</td>
</tr>
<tr>
<td>-</td>
<td>Optional match.</td>
<td>12- matches 1 and 12.</td>
</tr>
</tbody>
</table>
### Command Tool

The T-Spice **Command Tool** automates the insertion of T-Spice commands and device statements, in correct SPICE format, into the active window. You also can use it to specify filenames and command-line options before launching a T-Spice simulation.

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>\1</td>
<td>Begin tag.</td>
<td>\A([0-9]+) matches \A123 and substitutes <strong>123</strong> for the first tag (\1) in the replacement string.</td>
</tr>
<tr>
<td>)</td>
<td>End tag.</td>
<td></td>
</tr>
<tr>
<td>\n\m</td>
<td>Text matching the *th parenthesized component of the regular expression, where * is a single digit.</td>
<td>If the search string is (\text{ab})(\text{cd})) and the replacement string is \2\1, T-Spice will replace \text{abcd} with \text{cdab}.</td>
</tr>
<tr>
<td>&amp;</td>
<td>The entire matched regular expression.</td>
<td>If the search string is \text{Windows} and the replacement string is \text{MS-&amp;}, T-Spice will replace \text{Windows} with \text{MS-\text{Windows}}.</td>
</tr>
<tr>
<td>\0</td>
<td>The entire matched regular expression.</td>
<td></td>
</tr>
<tr>
<td>\1</td>
<td>Literal backslash</td>
<td>\a\n matches \a\n.</td>
</tr>
<tr>
<td>&amp;</td>
<td>Literal ampersand, to avoid expression substitution</td>
<td>If the search string is <strong>123</strong> and the replacement string is &amp;&amp;&amp;&amp;, T-Spice will replace <strong>123</strong> with <strong>123&amp;123</strong>.</td>
</tr>
</tbody>
</table>
Use **Edit > Insert Command** or the toolbar button to open the Command Tool.

The right pane lists general categories that expand into individual T-Spice simulation commands when you expand a category. When you click on a command name in the left-hand tree, fields representing individual fields open in the right-hand pane of the dialog.

Commands are inserted to the right of the cursor position, or replacing highlighted text (above the current line if nothing is selected and below the current line if something is selected.)

**Setting Simulation Options**

Also accessible from the toolbar button ( ), the **Simulation Settings** dialog allows you to specify options that will be used for all subsequent simulations.

If the netlist name is not user-set in S-Edit (in the "Spice File Name field" of **Setup > SPICE Simulation > General**), but a "Simulation Results Folder" is set, the netlist is by default stored in that folder.

The results folder contains the tsim file and database folder simulation results. The database folder contains all the binary and text results files created by the simulation except the tsim file.
Simulation > Simulation Settings—Options

Command line arguments: Use this field to enter command-line options, which modify a simulation without altering the input file. You can enter as many options as desired, separating each with a space. Refer to “Command-Line Options” on page 31 for a description of available options and their proper syntax.

Include/Lib search path: A semicolon-separated list of folders, used to search for include files, model files, and subcircuit definitions.

Additional SPICE commands: Commands that are prepended to the user’s spice deck. These are often useful for setting simulator options, and/or including specific header files.
Simulation > Simulation Settings—Output

**Location**

The output file(s) are created in the specified directory. This directory can be an absolute path, or a path relative to the input file.

**Keep all simulation results**

If this option is selected, each simulation run creates a new subfolder in the output folder; output files for that simulation run are placed in this subfolder. The subfolder name is based on the input file name, and includes additional timestamp information. A copy of the input file is also kept in this subfolder.

**Always overwrite T-Spice output files without prompting**
Simulation > Simulation Settings—W-Edit

**W-Edit Waveform Display**
Options for displaying simulation progress in W-Edit. Check an option to make it active.

- **Show during simulation**—displays traces in W-Edit while the simulation is running.
- **Show after simulation completes**—displays traces in W-Edit once the simulation is complete.
- **Do not show**—does not open W-Edit.
Launching a Simulation

Simulation > Run Simulation

The **Simulation > Run Simulation** command (shortcut F5 or toolbar button ➤), launches a T-Spice simulation on the currently active document. If a simulation is already in progress, the new simulation is added to the queue of jobs awaiting simulation. This queue is displayed in the **Simulation Manager** window.
**Simulation Manager**

The Simulation Manager allows you to control and monitor all T-Spice simulations. It queues files for simulation, displays their processing status, and allows you to stop or pause a simulation. You can also highlight one or more files in this window to view simulation results in the Simulation Status window or the W-Edit waveform viewer.

You can right-click on any simulation entry to access the pop-up menu with shortcuts to the controls described above, as well as additional menu items that open a new file (New), specify whether the Simulation Manager will be visible (Hide) and whether it will be docked (Docking view).

There is only one Simulation Status window, and all simulations display their output in this window. When you select a simulation in the Simulation Manager, the Simulation Status window displays the results for that simulation.

**Docking and Undocking the Simulation Manager**

The Simulation Manager can also be docked or undocked by right-clicking in any field and checking or unchecking Docking view in the pop-up menu. When it is docked, you can double-click on any edge of the dialog box or click-and-drag one of the sides to undock it.

**View > Simulation Manager**

Use View > Simulation Manager to display or hide the Simulation Manager. Each simulation occupies one line.

<table>
<thead>
<tr>
<th>Status</th>
<th>Input file</th>
<th>Start Date/Time</th>
<th>Elapsed Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>finished</td>
<td>SParameters Measurement_GaAsAmp_Teobench_GaAsAmp.sp</td>
<td>29 December</td>
<td>00:00:05</td>
</tr>
<tr>
<td>finished</td>
<td>TransferAnalysis_GaAsAmp_Teobench_GaAsAmp.sp</td>
<td>29 December</td>
<td>00:00:06</td>
</tr>
<tr>
<td>finished</td>
<td>ACAnalysis GaAsAmp_Teobench GaAsAmp.sp</td>
<td>30 December</td>
<td>00:00:08</td>
</tr>
<tr>
<td>finished</td>
<td>DCAnalysis GaAsAmp_Teobench GaAsAmp.sp</td>
<td>30 December</td>
<td>00:00:06</td>
</tr>
</tbody>
</table>

**Status**

Simulation status. Possible states include:

- **Queued**—the simulation is in the queue and will run when the simulation engine is available.
- **Running**—the simulation is underway.
- **Paused**—simulation has been suspended by the user.
- **Finished**—the simulation ran and output is available.
- **Stopped**—the simulation was stopped by the user.
- **Failed**—the simulation failed to run.

**Input file**

Full pathname of the input file

**Start Date/Time**

The date and time at which the simulation began execution.
The total simulation run time is in \texttt{hh:mm:ss} format. Pausing a simulation will not interrupt the measurement of elapsed time.

When the Simulation Manager is undocked, the following buttons are available to control queued simulations.

<table>
<thead>
<tr>
<th>Status</th>
<th>Input File</th>
<th>Start Time/Date</th>
<th>Elapsed Time</th>
<th>Run Simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finished</td>
<td>ParameterMeasurement_GaAsAmp_Testbench_GaAsAmp.sp</td>
<td>29 December ...</td>
<td>00:03:05</td>
<td></td>
</tr>
<tr>
<td>Finished</td>
<td>TransientAnalysis_GaAsAmp_Testbench_GaAsAmp.sp</td>
<td>26 December ...</td>
<td>00:03:05</td>
<td></td>
</tr>
<tr>
<td>Running</td>
<td>ACAnalysis_GaAsAmp_Testbench_GaAsAmp.sp</td>
<td>30 December ...</td>
<td>00:04:17</td>
<td></td>
</tr>
<tr>
<td>Queued</td>
<td>DCAnalysis_GaAsAmp_Testbench_GaAsAmp.sp</td>
<td>36 December ...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

- **Elapsed Time**: The total simulation run time in \texttt{hh:mm:ss} format. Pausing a simulation will not interrupt the measurement of elapsed time.

- **Run Simulation**: Invokes “Simulation > Run Simulation” (page 53). If a file is highlighted, the appropriate file and path names will automatically populate the Input file and Output file fields.

- **Stop**: Stops simulation processing for the highlighted file. Once a simulation is stopped, it cannot be resumed.

- **Pause/Resume**: Pauses simulation when the status of the highlighted file is Running, and resumes simulation when the status of the highlighted file is Paused.

- **Delete**: Removes the highlighted file from the simulation queue. If the simulation is running or paused, you will be prompted to stop processing on the file before it is deleted.

- **Empty List**: Removes all files from the simulation queue. If any simulations are running or paused, you will be prompted to stop processing before they are deleted.

- **Show Waveform**: Invokes the W-Edit waveform viewer for the selected simulation. Double clicking on a line in the simulation manager will also open W-Edit and load the corresponding results.

- **Show Netlist**: Opens a window with the selected input file. If the selected file is already open, makes that window active.

- **Show Output**: Opens a window with the output files corresponding to the selected simulation. If already open, makes that window active.
Simulation Manager Right-Click Menu

Right-clicking in any field of the Simulation Manager (either docked or undocked) opens a pop-up menu with the following options:

<table>
<thead>
<tr>
<th>Setup Simulation...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remove</td>
</tr>
<tr>
<td>Empty List</td>
</tr>
<tr>
<td>Open Waveform...</td>
</tr>
<tr>
<td>Open Netlist...</td>
</tr>
<tr>
<td>Open Operating Point Results...</td>
</tr>
<tr>
<td>Open Simulation Log...</td>
</tr>
<tr>
<td>Open Assert Log...</td>
</tr>
<tr>
<td>Open Measure Log...</td>
</tr>
<tr>
<td>Open CSV Results...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Compare Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Output Folder...</td>
</tr>
<tr>
<td>Rename Output Folder...</td>
</tr>
<tr>
<td>Delete Results</td>
</tr>
<tr>
<td>Hide</td>
</tr>
<tr>
<td>Docking view</td>
</tr>
</tbody>
</table>

**Setup Simulation...**
Opens the Setup Simulation dialog. Options selected here will apply to subsequent simulation jobs.

**Remove**
Removes the highlighted line(s) from the Simulation Manager. No files are deleted.

**Empty List**
Removes all lines from the Simulation Manager.

**Open Waveform...**
Show the output data file in W-Edit.

**Open Netlist...**
Open the input netlist file.

**Open Operating Point Results...**
Opens the operating point results .op file if one is present in the database folder.

**Open Simulation Log...**
Opens the simulation results .log file.

**Open Assert Log...**
Opens the assert command .assert file if one is present in the database folder.

**Open Measure Log...**
Opens the measurement results .measure file.

**Open CSV Results...**
Opens the .print output .csv file when the CSV option is enabled.

**Compare Results**
Textually compares files from two simulations. Select two rows in the Simulation Manager, then right-click on Compare Results to open the sub-menu.

This option is only available if the files are located in different directories. In the case where simulation output subfolders are automatically created, the diff runs on the copied input files that are created in each folder.

For this option to work you must have specified a Diff Program in the T-Spice Setup > Application > External Programs dialog.
Open Output Folder... Opens a Windows Explorer window to the database folder, which contains all the binary and text results files created by the simulation except the tsim file.

Rename Output Folder... Allows you to rename the database output folder.

Delete Results Removes the highlighted file from the simulation queue.

Hide Hide or show the Simulation Manager.

Docking view Works as a toggle to dock and undock the Simulation Manager. When checked, the Simulation Manager docks at the bottom of the T-Spice window frame.

Simulation Status

The Simulation Status window shows simulation statistics and progress information, as well as any warnings or error messages, for the currently running or most recently completed simulation.

View > Simulation Status

Use View > Simulation Status to toggle visibility of the Simulation Status window.

![Simulation Status Window](image)

Input file Name of the input file.

Progress The type of simulation, duration in nanoseconds, and percentage of the simulation completed.

Total nodes Total number of nodes simulated.

Total devices Total number of devices simulated.

Active devices Total number of active devices simulated.

Passive devices Total number of passive devices simulated.
<table>
<thead>
<tr>
<th>Source Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent sources</td>
<td>Total number of independent sources simulated.</td>
</tr>
<tr>
<td>Controlled sources</td>
<td>Total number of controlled sources simulated.</td>
</tr>
</tbody>
</table>
Encrypting Simulations

The T-Spice encryption feature lets you protect your models, subcircuits, parameters, libraries and netlists so you can distribute them to others without revealing proprietary information.

When recipients of encrypted netlists and components run T-Spice simulations, the T-Spice simulator will not print any listings of the components contained within the encrypted blocks, and protected data is not written into any file.

The encrypted library is essentially a black box into which the user cannot see. Note that encryption is a one-way process! There is no way to decrypt a file. However, if you overwrite an encrypted file with an unencrypted version, the encryption will not be retained.

**Note:** Only T-Spice can read an encrypted file. They cannot be read by other Tanner tools or by applications from other companies.

Encryption Password Methods

There are three ways to assign an encryption password.

**Assign a password to each encrypted block**

The `.protect` and `.unprotect` T-Spice commands are used to delimit data blocks that are to be encrypted. An optional `password` argument to the `.protect` command allows for a password to be assigned on a per-block basis.

**Assign a unique global password**

During the encryption process, you can set one password that will be used for all encrypted data blocks.

**Use the default password**

If you do not specify an encryption password during the encryption process, then T-Spice will use a default password string. Any T-Spice user will be able to simulate using the encrypted library, but the data will, as always, be hidden.

**Note:** It is a very good practice to test your SPICE file so that it is free of any errors prior to encrypting it!

Encryption Commands

`.protect`

Specifies the beginning of an encrypted block of the input file (netlist).

When the netlist is encrypted using the T-Spice File > Encrypt command, all of the text between the `.protect` and the `.unprotect` commands will be encrypted using 256 bit AES encryption. The text in the newly encrypted file will then appear as a large block of hexadecimal characters. The `.protect` command will be preserved in the encrypted file, however the optional `password=password` argument will be removed.
When you run simulations, all model, parameter, device, node and subcircuit listings will be suppressed for the commands and device declarations that are contained within the .protect block.

This is true whether or not the file has actually been encrypted.

- **.protect** can be abbreviated as **.prot**
- Each **.protect** command must have a matching **.unprotect** command
- If you modify the encrypted text in the **.protect** block, the T-Spice decryption process and the simulation will fail.

**Syntax**

```
.protect  [password=password]
or
.prot  [password=password]
```

**.unprotect**

Specifies the end of an encrypted block of the input file.

- **.unprotect** can be abbreviated as **.unprot**

**Syntax**

```
.unprotect
or
.unprot
```

**.visible**

Specifies the beginning of a sub-section of a protected block that should be visible to the user. This temporarily disables the data hiding aspect of the **.protect** command.

The **.visible** command is used to delimit the start of a block of code that is contained within a **.protect** block and should not be hidden from the user. The block of code will still be encrypted, but the models, devices, params, subcircuits, and any other elements contained within the visible block will not be hidden from the user, as they normally are with encryption.

- The **.visible** command does not require a matching **.invisible** command
- The visible block will continue until the **.invisible, .lib, .include, or .unprotect** command appears in the file

**Syntax**

```
.visible
```

**.invisible**

Specifies the end of a sub-section of a protected block that should be visible to the user. The statements and components following the **.invisible** command will be hidden from the user, in the normal manner for protected/encrypted blocks - probing is not allowed, and the encrypted data will not be listed or displayed in any tool
Syntax

.invisible

.option password

T-Spice uses the password option to permit simulation of a netlist that contains encrypted .protect blocks.

This password is identical to the encryption password that is entered during the T-Spice File > Encrypt command execution, and should be distributed in a safe and appropriate manner to intended library users so that they can perform simulations using the encrypted libraries.

The T-Spice input netlist may contain more than one password option in order to establish a list of available passwords.

When T-Spice encounters a protected block of code in the netlist during a simulation, it tests each available password to see if it is the correct one. In this way, a single simulation can use encrypted libraries that were provided by multiple vendors, or that have independent licensing and access restrictions.

The password option can be globally assigned for all simulations in the Additional SPICE commands field of the T-Spice dialog Simulation > Simulation Settings > Options.

Note: The exact password that was used for netlist encryption must be specified in the password option.

The Encryption Process

1. Protect the Input File

T-Spice uses the .protect and .unprotect commands to distinguish portions of input files that will be encrypted and hidden from the end-user.

You can place the protect/unprotect commands around any block of T-Spice commands. However, the following code blocks may not be separated – that is, they may not be partially within and partially without of a protected block:

- Subcircuit definitions: .subckt ... .ends
- Conditional statement blocks: .if ... .endif
- Library modules: .lib sectionname ... .endl
- Data blocks: .data ... .enddata

2. Encrypt the File

The protected blocks of code will be encrypted when you save the file using File > Encrypt from the T-Spice menu. Note that there is no special extension for an encrypted file.
The **Save Encrypted Spice File** dialog allows you to use either the default public password by checking "Use default password," or to enter a custom encryption password with the "Password" field. If you use a custom password, you must confirm it. Passwords are hidden as they are entered.

You can also choose to include a specific password for one or more protected blocks when you encrypt them, as shown below:

```
.protect password=judy
.subckt DiffCell Inn Inp Outm Outp VTune Vbl Vb2 Gnd Vdd W=5.00u
```

If you have already specified a password in the .protect command, T-Spice opens a conventional Save As dialog instead, as shown below.
**Warning:**

The file browser dialog will allow you to save the file with any name. We strongly recommend that you save the file with a different name than the unencrypted input file. Once it has been encrypted, there is no way to decrypt the file to its original state.
3. Distribute the Encrypted Files and Password

If you used a custom encryption password, then you must distribute this password to the end-user of the encrypted library. The user of the encrypted library must use the password option to inform T-Spice of the password value(s), at the beginning of the netlist file.

![T-Spice netlist example with password option]

Extent of Data Protection

The intent of T-Spice data encryption is to hide proprietary data from the end user. As such, log file listings of netlist information and the simulation database of solutions are limited in their content. The simulation solutions are clipped to remove all netlist components that are contained within protected blocks. This has the following effects for other Tanner Tools:

- W-Edit will not be able to display any traces for protected nodes or elements. Node voltages, device currents, device charges, and other device state values will not be stored in the solutions database, and cannot be viewed.
- S-Edit will not be able to back-annotate any node voltages, device terminal currents and charges, or any small-signal values for protected blocks.
- Model and device parameter listings will not be available in either S-Edit or the T-Spice user interface.
- The `.print` and `.probe` commands will not work for any encrypted components. Other simulation output reports such as operating point AC small-signal values and AC noise analysis values will not contain any protected components.
- The T-Spice log file and Simulation Status window contents will not contain any information about protected components.
- The following commands will not include any references to or information about protected components:

  `.option node` listings of nodes and device terminal node connections
.option nomod=0 model parameter listings

.option list device parameter listings

.option xref subcircuit parameter and .param listings

.option params listing of model and device parameters

Of course, encryption will fail if a file has no .protect / .unprotect blocks, if each .protect command does not have a matching .unprotect command (or vice-versa), or if the .protect blocks are already encrypted.

You are allowed to encrypt a file that contains both encrypted and unencrypted .protect blocks, but only the unencrypted block will be encrypted.

Encrypting Verilog-A/MS

The T-Spice Verilog-A/MS system supports processing of encrypted Verilog-A and Verilog-AMS files.

The process for encrypting and simulating Verilog-A/MS is as follows:

- In the T-Spice UI, select the menu item File > Open..., and a file browser will appear. Change the file type drop-down list to enable browsing for Verilog-A/MS files, Verilog Files (*.v;*.va;*.vams;*.verilog). Locate and open the file that you wish to encrypt.

- Select the menu item File > Encrypt...
  This will cause 128-bit encryption to be applied to the Verilog file, and for a new encrypted file to be created. The new file is named the same as the original file, except for an added character ’p’ at the end of the extension name, e.g. a2d.va encryption results in new file a2d.vap.

- Simulation with the encrypted file is accomplished in the same manner as unencrypted files, using the .hdl command. e.g. .hdl a2d.vap
4 Input Conventions

When you run a simulation, the T-Spice parser interprets the input file. For T-Spice to function efficiently and as compatibly as possible with other versions of SPICE, the parser must enforce a number of language conventions. This chapter summarizes those conventions. For more information, see “Simulation Commands” on page 75 and “Device Statements” on page 171.

Names

All nodes and devices in the circuit must be identified uniquely by their names. Node and device names have the following features.

- Their length is unlimited (except by hardware constraints).
- They can include all characters except tabs, spaces, semicolons (;), single quotes (’), curly braces ({}), parentheses and equal signs (=).
- The dollar sign ($) can appear in names, but it cannot be the first character of the name or be a name by itself.
- They are case insensitive.

Model names cannot start with digits.

The following examples are all valid names:

```
in Alpha16 ONE[21]3_72
```

Reserved Names

T-Spice uses reserved node names for the default system ground: gnd, gnd!, ground, and 0 (zero). All instances of these nodes are connected and treated as the same node, which is fixed at a potential of 0.0 volts.

The following keywords (in any combination of uppercase and lowercase letters) cannot be used as names:

```
ac busi off piei pulse rounding transfer
bit dc noise onoise pwl r sffm
biti exp params pwlfile repeat sin
bus inoise pie sini
```

Device Names

Most device statements are of the form
where the variable Z represents the required key letter which uniquely specifies the device type, and the variable xxx represents a user-supplied alphanumeric string.

For example, MOSFET statements have a form following this example:

```
mtran1 d g s b nmos l=2um w=2um
```

In this example, m is the required key letter and tran1 is the user-supplied string. The device’s name is \textit{mtran1} (not \textit{tran1}).

A particular device may be indicated in the input file by \textit{either} case, for example, m or M, of the key letter — but the case must remain constant for the same device throughout the file.

### Hierarchical Names

Hierarchical node names are used to refer to nested subcircuit nodes. Each level in the name of a node is separated by a period (\textendash).

For example, the internal node \texttt{ing} in the \texttt{xnand} subcircuit contained in the \texttt{xadder} subcircuit is specified as

```
xadder.xnand.ing
```

### Subcircuit Pin Name Aliasing

T-Spice recognizes subcircuit pin node names by their internal names as well as their global names.

For example, the netlist

```
.subckt test a b
r1 a c 100
r2 c b 100
.ends
x1 a1 b1 test
```

creates three nodes: \texttt{a1}, \texttt{b1}, and \texttt{x1.c a1} and \texttt{b1} are globally recognized node names; but they may also be referred to by aliases \texttt{x1.a} and \texttt{x1.b}, respectively — for example, within a command such as \texttt{.print dc v(x1.a)}.

### Comments

Comments provide information about the circuit, but are not processed as part of the formal circuit description. Comments are generally indicated by the presence of special characters called \textit{delimiters}.

However, T-Spice always treats the \textit{first line of the input file as a comment}, even without comment delimiters.

Other comments may be placed anywhere in the circuit description.

Several comment styles are allowed, for compatibility with other versions of SPICE:
- An asterisk (*), dollar sign ($), or semicolon (;) in the first column of a line indicates that the entire line is a comment.
- Any value after an asterisk is ignored if the expression is not enclosed in single quotes, or beginning with version 16, in parentheses.
- A dollar sign or semicolon, but not an asterisk, anywhere in a line other than in the first column indicates that the rest of the line is a comment.
- C-style comments, enclosed by the delimiters /* and */, can be used anywhere, except in the middle of multi-word commands (such as .print tran) or arguments. A C-style comment is not restricted to one line.

The comments in the following examples are highlighted:

```
* Lines beginning with asterisks, $ or dollar signs, ; or semicolons are ignored.
r1 node1 node2 4k $ This comment can follow a command
c2 node3 node4 100f ; This is like a '$' comment
v1 node5 GND /* This is a C-style comment */ 3volt
```

A C-style comment that interrupts a command or argument may cause an error message. However, comments may appear between arguments.

The first two lines in the following examples would cause error messages:

```
.print /* wrong */ tran v(1)
.options prtdel /* wrong */ = 0.01
.options abstol=1e-8 /* OK */ reltol=1e-4
```

### Line Continuation

Input file lines may be of any length; however, it is often convenient to break up long lines for readability.

A plus sign (+) in the first column denotes line continuation. For example:

```
.model nmes NMF
+ vto=-2.5 rs=100 rd=200 pb=0.7
+ alpha=2.0 cgs=500.0f cgd=100.0f
```

### Comments in Continued Lines

Comments may appear between continued lines. However, blank lines may not appear between continued lines.

A continued line may include a dollar sign ($) or semicolon (;) comment symbol, in which case the rest of the line is ignored.

No plus sign is needed for a comment line, a line continuing a C-style comment, or a line continuing an expression (see “Parameters” on page 70).

For example:

```
.options
```
* now we declare several options, continuing the line
  + prtdel = 10ms $ (; would work too)
  + abstol = 1e-10 /* we can put a
    C-style comment here */

Expressions and Continued Lines

Plus signs cannot be used to indicate addition in the first column within multi-line expressions.

For example:

```
.options numnd = '7 +
log(10.1)   $ will be added to previous term
+ 2'       $ will NOT be added - '+' is ignored
```

Numbers and Units

Some commands and statements require arguments representing physical quantities with attached units (such as seconds or volts). Such numbers can be expressed in floating point, scientific, or fixed-point notation, and can be followed by metric abbreviations indicating order of magnitude.

The base units (s, v, a, f, h) are implicit from the context and are optional. For example, the following expressions can all specify 12 nanoseconds in the appropriate context:

```
12ns 12e-9 .012u 12000p
```

Acceptable metric abbreviations are as follows.

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Prefix</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>t or T</td>
<td>tera-</td>
<td>10^{12}</td>
</tr>
<tr>
<td>g or G</td>
<td>giga-</td>
<td>10^{9}</td>
</tr>
<tr>
<td>meg or MEG</td>
<td>mega-</td>
<td>10^{6}</td>
</tr>
<tr>
<td>x or X</td>
<td>mega-</td>
<td>10^{6}</td>
</tr>
<tr>
<td>k or K</td>
<td>kilo-</td>
<td>10^{3}</td>
</tr>
<tr>
<td>m or M</td>
<td>milli-</td>
<td>10^{-3}</td>
</tr>
<tr>
<td>'u or U</td>
<td>micro-</td>
<td>10^{-6}</td>
</tr>
<tr>
<td>n or N</td>
<td>nano-</td>
<td>10^{-9}</td>
</tr>
<tr>
<td>p or P</td>
<td>pico-</td>
<td>10^{-12}</td>
</tr>
<tr>
<td>f or F</td>
<td>femto-</td>
<td>10^{-15}</td>
</tr>
</tbody>
</table>

The abbreviation f is ambiguous because it can mean either the scale indicator “femto-” or the unit “farad.” T-Spice employs the following convention: f by itself means “femto-.” Thus 100f means “100 femto-,” where the unit is clear from the context. Where f precedes a base unit, as in ff and fs, or follows a metric abbreviation, as in uf, there is no ambiguity. When “farad” by itself is meant, no unit should be used.

A commonly used unit abbreviation is mil (or MIL), representing 10^{-3} inch.
Parameters

You can declare parameters and assign them values with the `.param` command. Parameters cannot be reassigned values within an input file.

Parameter names can contain any characters except tabs, spaces, commas, curly braces, parentheses, single quotes, square brackets, equal signs, and algebraic operators (+ – * / ^).

Expressions

Any number in a command or statement may be replaced by an algebraic expression. Expressions must conform to the following conventions.

- They must be enclosed by single quotes (' or, with version 16, by parentheses. The outer parentheses behave as quotes, so you can have embedded parentheses and spaces, and can continue to the next line.
- They may span several lines. The plus sign (+) is ignored if it appears in the first column of a continued expression.
- They may include comments that do not interrupt numeric values or parameter names.

Expressions may involve any valid combination of numbers, parameters (".param" (page 131)), operations, algebraic functions, and user-defined functions. T-Spice evaluates expressions according to a standard mathematical operator precedence, shown below. Level 1 has the highest operator precedence, and level 10 has the lowest.

Operators

The operations and functions available in T-Spice are summarized in the following tables (in which the variables x and y represent numbers, parameter names, or subexpressions). All angles are in radians.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x)</td>
<td>Parentheses — to override operator precedence</td>
</tr>
<tr>
<td>x + y</td>
<td>Addition</td>
</tr>
<tr>
<td>x – y</td>
<td>Subtraction</td>
</tr>
<tr>
<td>x * y</td>
<td>Multiplication</td>
</tr>
<tr>
<td>x / y</td>
<td>Division</td>
</tr>
<tr>
<td>x^y, x**y</td>
<td>Exponentiation (x^y)</td>
</tr>
<tr>
<td>–x</td>
<td>Unary negation</td>
</tr>
<tr>
<td>x &lt; y, x &lt;= y, x &gt; y, x &gt;= y</td>
<td>Relational operators - return 1 if the relation is true, otherwise 0</td>
</tr>
<tr>
<td>x == y, x != y</td>
<td>Equality operators - return 1 if the (in)equality is true, otherwise 0</td>
</tr>
<tr>
<td>x &amp;&amp; y</td>
<td>Logical AND - returns 1 if x and y are true (non-zero), otherwise 0</td>
</tr>
<tr>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>
Operator Precedence

The following table lists operator precedence. Operators with the highest precedence are evaluated first. Operators with equal precedence are evaluated left to right.

<table>
<thead>
<tr>
<th>First (highest)</th>
<th>(x)</th>
<th>parentheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Second</td>
<td>f(x)</td>
<td>function call</td>
</tr>
<tr>
<td></td>
<td>[...]</td>
<td>TCL evaluation</td>
</tr>
<tr>
<td>Third</td>
<td>X^Y</td>
<td>exponentiation</td>
</tr>
<tr>
<td>Fourth</td>
<td>X*Y</td>
<td>multiplication</td>
</tr>
<tr>
<td></td>
<td>X/Y</td>
<td>division</td>
</tr>
<tr>
<td></td>
<td>X%Y</td>
<td>modulo</td>
</tr>
<tr>
<td>Fifth</td>
<td>X + Y</td>
<td>addition</td>
</tr>
<tr>
<td></td>
<td>X - Y</td>
<td>subtraction</td>
</tr>
<tr>
<td></td>
<td>-X</td>
<td>unary negation</td>
</tr>
<tr>
<td>Sixth</td>
<td>&lt;</td>
<td>relational operators</td>
</tr>
<tr>
<td></td>
<td>&lt;=</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&gt;=</td>
<td></td>
</tr>
<tr>
<td>Seventh</td>
<td>==</td>
<td>equality operators</td>
</tr>
<tr>
<td></td>
<td>!=</td>
<td></td>
</tr>
<tr>
<td>Eighth</td>
<td>X&amp;&amp;Y</td>
<td>logical AND</td>
</tr>
<tr>
<td>Ninth</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Tenth</td>
<td>?:</td>
<td>conditional</td>
</tr>
</tbody>
</table>

**Built-in Functions**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs(x)</td>
<td>Absolute value of x (same as fabs)</td>
</tr>
<tr>
<td>acos(x)</td>
<td>inverse cosine of x (error if</td>
</tr>
<tr>
<td>asin(x)</td>
<td>Inverse sine of x (domain error if</td>
</tr>
<tr>
<td>atan(x)</td>
<td>Inverse tangent of x (range: [-p/2, p/2])</td>
</tr>
<tr>
<td>atan2(x,y)</td>
<td>Inverse tangent of y/x (range: [-π, π])</td>
</tr>
<tr>
<td>ceil(x)</td>
<td>Smallest integer not less than x</td>
</tr>
<tr>
<td>cos(x)</td>
<td>Cosine of x</td>
</tr>
<tr>
<td>cosh(x)</td>
<td>Hyperbolic cosine of x</td>
</tr>
<tr>
<td>db(x)</td>
<td>x in decibels: (sign of x)-20·log10(</td>
</tr>
<tr>
<td>err(x,y)</td>
<td>error analysis; abs(x-y) / max(x,y)</td>
</tr>
</tbody>
</table>
### Function Descriptions

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \exp(x) )</td>
<td>( e^x )</td>
</tr>
<tr>
<td>( \text{fabs}(x) )</td>
<td>Absolute value of ( x ) (same as ( \text{abs} ))</td>
</tr>
<tr>
<td>( \text{floor}(x) )</td>
<td>Largest integer not greater than ( x )</td>
</tr>
<tr>
<td>( \text{fmod}(x,y) )</td>
<td>Remainder of ( x/y ) (error if ( y = 0 ))</td>
</tr>
<tr>
<td>( \text{if}(c,a,b) )</td>
<td>Conditional statement, if ( c ) is true, then ( a ), else ( b ).</td>
</tr>
<tr>
<td>( \text{or} ) or ( c ? a : b )</td>
<td>Two syntax variations are supported for the conditional statement - the ( \text{if()} ) function call, and the C style of conditional expression</td>
</tr>
<tr>
<td>( \text{int}(x) )</td>
<td>Convert ( x ) to an integer, removing the fractional portion</td>
</tr>
<tr>
<td>( \text{ldexp}(x,y) )</td>
<td>( x \times 2^y ) for integer ( y )</td>
</tr>
<tr>
<td>( \log(x) )</td>
<td>Natural logarithm of ( x ) (error if ( x \leq 0 ))</td>
</tr>
<tr>
<td>( \log10(x) )</td>
<td>Logarithm (base 10) of ( x ) (error if ( x \leq 0 ))</td>
</tr>
<tr>
<td>( \log2(x) )</td>
<td>Logarithm (base 2) of ( x ) (error if ( x \leq 0 ))</td>
</tr>
<tr>
<td>( \text{max}(x1, x2) )</td>
<td>Evaluates to the maximum of the two arguments.</td>
</tr>
<tr>
<td>( \text{min}(x1, x2) )</td>
<td>Evaluates to the minimum of the two arguments.</td>
</tr>
<tr>
<td>( \text{pow}(x,y) )</td>
<td>In HSPICE compatibility mode (default): Integral power function. Returns the real value ( x ) raised to the integer part of ( y ). In PSPICE compatibility mode (.option compatibility=pspice): Power function. Returns ( x ) raised to the power ( y ).</td>
</tr>
<tr>
<td>( \text{pwr}(x,y) )</td>
<td>In HSPICE compatibility mode (default): ( \text{(sign of } x) \cdot \text{sign}(x)^y )</td>
</tr>
<tr>
<td>( \text{pwr}(x,y) )</td>
<td>In PSPICE compatibility mode (.option compatibility=pspice) exponentiation ( x^y ), equivalent to ( \text{pow}(x,y) )</td>
</tr>
<tr>
<td>( \text{random}(x) )</td>
<td>Generates a random number in the range ( 0 \ldots x )</td>
</tr>
<tr>
<td>( \text{sgn}(x) )</td>
<td>Sign of ( x ): -1 if ( x &lt; 0 )</td>
</tr>
<tr>
<td></td>
<td>0 if ( x = 0 )</td>
</tr>
<tr>
<td></td>
<td>1 if ( x &gt; 0 )</td>
</tr>
<tr>
<td>( \text{sign}(x,y) )</td>
<td>( \text{(sign of } y) \cdot \text{sign}(x)^y )</td>
</tr>
<tr>
<td>( \sin(x) )</td>
<td>Sine of ( x )</td>
</tr>
<tr>
<td>( \sinh(x) )</td>
<td>Hyperbolic sine of ( x )</td>
</tr>
<tr>
<td>( \sqrt{x} )</td>
<td>Square root of ( x ) (-( \sqrt{</td>
</tr>
<tr>
<td>( \text{stp(expression)} ) or ( \text{stp(expression1, expression2)} )</td>
<td>The first syntax evaluates to 0 if expression is negative, and 1 otherwise. The second syntax evaluates to 0 if ( \text{expression1} ) is less than ( -\text{expression2} ), to 1 if ( \text{expression1} ) is greater than ( \text{expression2} ), and to ( (\text{expression1}+\text{expression2})/(2*\text{expression2}) ) otherwise. The second syntax thus provides a continuous approximation to a step function.</td>
</tr>
</tbody>
</table>
Differentiation and Integration functions

In addition to the Standard math library functions, T-Spice provides a set of functions for computing integrals and derivatives of data.

The \( \text{ddt}(f) \), \( \text{d2dt}(f) \), and \( \text{idt}(f) \) (aka \( \text{sdt}(f) \)) functions compute the time derivatives and integrals of transient simulation data.

The \( \text{ddx}(f,x) \), \( \text{d2dx}(f,x) \), and \( \text{idx}(f,x) \) (aka \( \text{sdx}(f,x) \)) functions compute the derivatives and integrals of the variable or expression \( f \) with respect to the independent variable or expression \( x \).

The time-based integration and differentiation functions, and the generic \( x \) dependent forms of the functions, both use polynomial fits to the data for computing the integrals and derivatives. In the case of the transient functions, the order of the polynomial tracks that of the transient simulation engine, which is controlled by the “maxord” (page 270) option. The generic \( x \) dependent functions use a second-order polynomial fit, by default.

The order of the function equations can be changed using the \texttt{const_dt_maxord}, and \texttt{const_dx_maxord} options:

\begin{verbatim}
.options const_dt_maxord=[0-4] ; default value is 0, selecting transient order
.options const_dx_maxord=[1-4] ; default value is 2
\end{verbatim}

\textbf{Note:}\n
The quality of integrals is highly dependent upon the order of the fitting polynomial. It is important to select an order of equations which is an appropriate fit to the data. If the dependent function \( f \) is highly irregular or is a square-shaped digital signal, then a 1st or 2nd order integration is best. Higher order equations should only be used for very smooth data.

If the integration solutions are suspicious, perhaps containing very large integral values, then the solutions should be verified by re-running the simulation with the function integral order set to 1, which results in a piecewise summation of integrands, without higher order effects.

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\text{ddt}(f)</td>
<td>Derivative of ( f ) with respect to time</td>
</tr>
<tr>
<td>\text{d2dt}(f)</td>
<td>Second derivative of ( f ) with respect to time</td>
</tr>
</tbody>
</table>
### Function | Description
--- | ---
idt(f) | Integrate f in time
sdt(f) |  
ddx(f,x) | Derivative of f with respect to x
 d2dx(f,x) | Second derivative of f with respect to x
idx(f,x) | Integrate f in x
sdx(f,x) |  
5 Simulation Commands

Introduction

This chapter describes features, outlines syntax, and gives examples for the simulation commands of the T-Spice circuit description language.

The commands are listed in alphabetical order. Many commands have options, which branch to different modes, and arguments, which indicate expressions, nodes, or devices to be operated on.

Options and arguments must be separated by spaces or new lines (with line continuation).

The Syntax sections follow these conventions:

- *Italics* indicate variables to be replaced by actual names, numbers, or expressions.
- Curly braces {} indicate alternative values for the same option or argument.
- Square brackets [] enclose items that are not required.
- Vertical bars | separate alternative values for the same option or argument.
- Ellipses ... indicate items that may be repeated as many times as needed.

Brackets, vertical bars, and ellipses are not typed in the input file. All other characters are typed as shown. For further information, see “Input Conventions” on page 66 and “Device Statements” on page 171.
.ac

Performs AC analysis to characterize the circuit’s dependence on small-signal input frequency: the DC operating point is computed, a linearized small-signal model is constructed at the DC operating point, and the circuit’s response over a range of frequencies is measured.

- Small-signal parameters are reported to the main simulation output file. You can disable this reporting or specify a different output file using the command “.acmodel” (page 79). Reporting is automatically disabled if the simulation contains more than 1000 nodes.
- AC analysis results can be reported with the commands .print ac, .probe ac (for binary output), or .measure ac. For additional information on these commands, see “.print” (page 137), “.probe” (page 152), and “.macro .eom” (page 107).
- The frequency can be varied linearly, by octaves, or by decades, by specifying a total number of test points, or by listing specific test frequencies.

Syntax

```
.ac {lin|oct|dec} num start stop [sweep swinfo] [analysisname=name]
```

or

```
.ac list f1 f2 ...fn [sweep swinfo] [analysisname=name]
```

or

```
.ac poi num f1 f2 ...fn [sweep swinfo] [analysisname=name]
```
Parameters

lin | oct | dec Frequency variation mode.
- lin: linear sweep.
- oct: logarithmic sweep by octaves.
- dec: logarithmic sweep by decades.

list Using the list keyword with .ac allows the user to specify a list of frequencies (values) for which the analysis is to be performed.

poi Using the poi keyword with .ac allows the user to specify a list of frequencies (values) for which the analysis is to be performed. The poi mode of processing is the same as the list mode, except that the syntax for poi requires that the number of frequencies, N, be specified next.

num Frequency count.
- Linear mode: Number of frequencies between start and stop.
- Octave mode: Number of frequencies per octave between start and stop.
- Decade mode: Number of frequencies per decade between start and stop.
- poi mode: Total number of frequency values.

start First frequency. (Unit: Hertz.)

stop Last frequency. (Unit: Hertz.)

sweep Specifies the parameter values of the sweep for which analysis will be performed. The sweep keyword is equivalent to “.step” (page 158) and uses the same parameter syntax. However, sweep applies only to one analysis command, while .step applies to all analysis commands in the input file. If sweep is specified on an analysis command and .step is present, the sweep sweep is nested inside the .step sweep. The parameter sweep may be used to specify a parametric sweep, Monte Carlo analysis, or optimization.

analysisname Specifies an analysis name that will be referenced by an .optimize simulation command. For further information, see “.optimize” (page 123).

Examples

.ac DEC 5 1MEG 100MEG

Defines a frequency sweep from 1 MHz to 100 MHz by decades, with 5 points per decade.

.ac LIN 100 10K 100MEG

Defines a linear frequency sweep from 10 kHz to 100 MHz with 100 equally spaced points.

.ac list 5 50 500 5000 sweep rval dec 10 1 1000
Performs a logarithmic sweep of the parameter \texttt{ rval} from 1 to 1000, using 10 points per decade; also performs an AC analysis at the four specified frequencies for each value of \texttt{ rval}. 
.acmodel

Modifies reporting of small-signal model parameters and operating points for specified devices in conjunction with AC analysis or DC operating point analysis. Small-signal parameters are automatically reported to the main simulation output file with the use of either .ac or .op in the input file.

- If "ac" (page 76) and "op" (page 121) are missing from the input file, .acmodel is ignored.
- No small-signal data is reported if the simulation model has more than 1000 nodes.
- Small-signal data are available for diodes, resistors, BJT s, JFETs, MESFETs, MOSFETs, and Verilog-A devices.

Syntax

.acmodel [device [[,] device ...]]]

device Device(s) for which small-signal model parameters and operating points are to be given. If no devices are specified, then no small-signal data is reported.

Examples

.acmodel {mt1 mt2}

Prints data for devices mt1 and mt2.

.acmodel {}

Turns off all small-signal parameter reporting.
.alter

Causes a simulation to be repeated with slight changes as specified after the .alter command. Multiple .alter commands can appear in the same netlist. T-Spice performs the first simulation with all commands that occur before the first .alter command; the second simulation incorporates changes between the first and second .alter commands; the third simulation incorporates changes between the second and third .alter commands, and so on. The optional altername string identifies the .alter block that follows and is used to identify the corresponding simulation output in output files.

A .alter block can contain any legal T-Spice statement. This command occurs at the end of a complete input file.

- Elements, option values, parameter values, and "connect" (page 86) and "model" (page 116) statements in a .alter block replace equivalent statements in the main netlist if the name of the element, parameter, option, data set, or model parameter set matches. If no name match can be made, the statement in the .alter block is simply added to the simulation.
- Statements are cumulative and progressive from one .alter block to the next such that any additions or changes made in block N will also occur (unless superseded) in block N+1.
- Initial conditions (".hdl" (page 98)) and "nodeset" (page 119) replace equivalent commands that operate on the same nodes or devices. If no equivalent command is found in the main netlist, the command is simply added.
- If a .alter block contains a "temp" (page 163) command, any .temp commands in the original netlist are replaced with the new .temp command.
- The commands "lib" (page 104) and "if ... / elseif ... / else / endif" (page 101) can be used within "alter" (page 80) blocks to include command and element statements.
- The "del lib" (page 90) command can be used to delete a library section included in the original input file. All commands and elements in the library section are ignored during the "altered" simulation.
- Simulation commands such as "ac" (page 76), "tran" (page 165), and "step" (page 158) do not replace commands in the original input file, but are simply added on as new commands.
- Since trace expressions in W-Edit use the comma as the primary delimiter and equal sign as the secondary delimiter, using either of these characters in an alter name will interfere with parsing of the subsequent trace expression. Instead of commas in the alter names in .inc files, you can use the @ character. Do not use commas or equal sign characters in .alter names.

Syntax

.alter [altername]

Examples

```
v1 1 0 1
r1 1 2 1k
r2 2 0 1k
.op
.alter
r1 1 2 2k
.alter r2_4k
r2 2 0 4k
.end
```
This performs three simulations:

The first uses \( r1=1k, \ r2=1k \), and is identified by alter=0; the output is \( v(2)=0.5 \).

The second uses \( r1=2k, \ r2=1k \), and is identified by alter=1; the output is \( v(2)=0.333333 \ (1/3) \).

The third uses \( r1=2k, \ r2=4k \), and is identified by alter=r2_4k; the output is \( v(2)=0.666667 \ (2/3) \).

The abbreviated output would be:

\[
\begin{align*}
\text{*SEDIT: Alter=0} \\
\text{*SEDIT: Analysis types DCOP 1 ACMODEL 0 AC 0 TRANSIENT 0 TRANSFER 0 NOISE 0} \\
\text{* BEGIN NON-GRAphICAL DATA} \\
\text{DC ANALYSIS - alter=0} \\
v(2) &= 5.0000e-001 \\
v(1) &= 1.0000e+000 \\
i(v1) &= -5.0000e-004 \\
\text{* END NON-GRAphICAL DATA} \\
\text{*SEDIT: Alter=1} \\
\text{*SEDIT: Analysis types DCOP 1 ACMODEL 0 AC 0 TRANSIENT 0 TRANSFER 0 NOISE 0} \\
\text{* BEGIN NON-GRAphICAL DATA} \\
\text{DC ANALYSIS - alter=1} \\
v(2) &= 3.3333e-001 \\
v(1) &= 1.0000e+000 \\
i(v1) &= -3.3333e-004 \\
\text{* END NON-GRAphICAL DATA} \\
\text{*SEDIT: Alter=2} \\
\text{*SEDIT: Analysis types DCOP 1 ACMODEL 0 AC 0 TRANSIENT 0 TRANSFER 0 NOISE 0} \\
\text{* BEGIN NON-GRAphICAL DATA} \\
\text{DC ANALYSIS - alter=r2_4k} \\
v(2) &= 6.6667e-001 \\
v(1) &= 1.0000e+000 \\
i(v1) &= -1.6667e-004 \\
\text{* END NON-GRAphICAL DATA}
\end{align*}
\]
You can characterize the safe operating area (SOA) of your circuit using the assert command. Assert establishes upper and lower limits on values of netlist parameters, model parameters, device instance parameters, and simulation electrical state variables. Violations of the established SOA limits will be logged to the Simulation Status window and to the safe operating area violations log file, *.assert.

**Syntax**

```
.assert name
    [ sub=subcircuit ]
    [ dev=instance ]
    [ mod=model ]
    [ node=node ]
    [ primitive=primitive ]
    [ expr='expression' ]
    [ param=param ]
    [ modelparam=modelparam ]
    [ modeltype=modeltype ]
    [ modelclass=modelclass ]
    [ min=value ]
    [ max=value ]
    [ abs_min=value ]
    [ abs_max=value ]
    [ start=x_start_value ]
    [ stop=x_stop_value ]
    [ duration=x_duration_value ]
    [ message="message"]
    [ level= warning | error ]
    [ info= yes | no ]
    [ when= ac | dc | noise | op | setup | tf | tran ]
    [ measure= measurename ]
    [ value= value ]
```

- **name**: A label for the assert item.
- **subcircuit**: Constrains the SOA limits to elements within a particular subcircuit and its sub-hierarchy.
- **instance**: Device instance name for device parameter checking.
- **model**: Model name for model parameter checking.
- **node**: Node name for node value checking.
- **primitive**: Device type specifier, which may be either the standard initial character of the name - d (diode), m (mosfet), q (bjt), etc., the full device name - diode, mosfet, bjt, etc., or the keyword node.
- **expression**: A device parameter name.
- **param**: A model parameter name.
- **modelparam**: A model parameter name.
### Examples

```plaintext
.assert CapLimit primitive=c param=c max=1n
.assert MosCap primitive=m param=capbd max=10p
.assert Isupply dev=vdd param=i1 abs_max=1A
.assert TotalPower max=.1w expr='power()' when=tran start=100n
.assert Vout node=out param=v max=3.5
.assert model_Vsat mod=nmos8 modelparam=vsat max=10
.measure TRAN FallTime TRIG v(Out) VAL='(Vpwr-0)*90/100+0' TD=0 Fall=1 TARG v(Out) VAL='(Vpwr-0)*10/100+0' TD=0 FALL=1
.assert Vout_measure measure=falltime max=20n
```

#### Note:

SOA limits may be specified as expressions, e.g. `max='i1(r1)*v(r1)'`
.checkpoint

Enable transient analysis checkpoint-restart capability. When the .checkpoint command is used, transient analysis information will be saved (checkpointed) so that subsequent simulation jobs may continue the simulation from the checkpoint time (restarted).

The restart portion of checkpoint-restart is accomplished by running a subsequent simulation job with added qualifiers on the .tran statement: restart, restartfile=filename, or restarttime=time. The simulation results file for restarted jobs will only contain data beginning at the checkpoint time, which is also the restart time.

Restrictions and limitations for restarting simulations

You will not be able to restart a simulation if the topology has been changed. This includes:

- adding or deleting nodes or devices
- changing names of components

You may change model and device parameters, with the caveat that significant changes might lead to convergence problems at restart.

With the same warning about potential convergence problems, you can also modify:

- global and device temperatures
- simulator tolerances and settings (except integration method and order)

New output commands, such as .assert, .print, .measure can safely be added to the restart simulation.

Note: The same version of T-Spice must be run for the checkpoint and the restart jobs.

Syntax

.checkpoint [ clock = clock ] [ file = filename ] [ period = period ] [ time = t1, t2, ... tn ]

clock

walltime value at which the simulation will continually be checkpointed, at every integral multiple of the time (units: seconds).

filename

name of the binary data file that will contain all simulation state information needed to restart the simulation at the checkpoint time.

period

transient time period at which checkpointing will continually be performed, at every integral multiple of period (units: seconds).

t1, t2, ... tn

a list of transient analysis timepoints at which checkpointing is to be performed (units: seconds).
Examples

Suppose that a very large circuit transient analysis is being run, and the user wants to guard against losing results if the job is stopped. The following command could be added to the input netlist in order to save checkpoint data every hour of simulation time.

```
.checkpoint file=D:\simulations\bigsim.chk clock=3600
```

The user can stop the job after it has simulated for several days, and then start a new job close to where the first was left off. The restart job will begin at the final checkpoint time, which will probably not be exactly the same as the first simulation stop time. The restart command might look like this:

```
.tran 10n 500m restart restartfile=D:\simulations\bigsim.chk
```

Checkpoint-restart capability can also be used to run multiple variations of an original simulation. This is useful if the original transient analysis requires a very long time to reach steady-state, and then a number of short-running variations are to be performed.

```
.checkpoint file=C:\tmp\mytran.chk time=1m 1.1m 1.2m 1.3m 1.4m 1.5m
```

The following restart simulations might contain the command:

```
.tran 10n 2m restarttime=1.5m
```
.connect

This command will connect two nodes in your circuit, so that the two nodes will be simulated as only one node. In essence, one node name becomes an alias for another node name.

Both nodes must be at the same level in the circuit design that you are simulating: you cannot connect nodes that belong to different subcircuits.

Syntax

.connect node1 node2

If you connect node2 to node1, you can then refer to either node1 or node2 in other simulation commands, and it will refer to the same node.

Examples

vcc 0 cc 5v
r1 0 1 5k
r2 1 cc 5k
.tran 1n 10n
.print i(vcc) v(1)
.alter
.connect cc 1
.end

The first .tran simulation includes two resistors. Later simulations have only one resistor, because r2 is shorted by connecting cc with 1.

You may also use multiple .connect statements to connect several nodes together:

.connect node1 node2
.connect node2 node3

connects both node2 and node3 to node1. The T-Spice simulation evaluates node voltages and related terms only for node1; node2, and node3 are the same node as node1.

Note: If you set .option node, T-Spice prints out a node connection table.
.data

Used to incorporate external numerical data into simulations. The data can be used to specify reference data for error measurements, which is often used in conjunction with optimization for model parameter extraction. Also used to specify parametric sweeps in which several variables are swept simultaneously.

Syntax

```
.data dataname
  + colname [colname [...]]
  + value [value [...]]
  + value [value [...]]
  ...
.enddata
```

The number of columns is equal to the number of values per row.

dataname Name assigned to the data set.
colname Name assigned to a column of data within the data set.
value Numeric parameter value. These values may be expressions, but must not depend on other parameter values.

A data set can be used with the “.macro /eom” (page 107) command to compute differences between simulation output curves and corresponding external data.

A data set can also be used to define a parameter sweep on an “.ac” (page 76), “.dc” (page 89), “.step” (page 158), or “.tran” (page 165) command.

The syntax for the sweep parameter on those commands is as follows:

data=dataname

where dataname refers to a .data statement of the same name. Each row of numbers in the .data statement corresponds to a sweep step, and each column refers to a sweep parameter (which must be a global parameter defined using .param) and the values which the parameter takes on. The colname strings identify the parameters to be swept. For each sweep step, the sweep parameters take on the values listed in a row of data.

Examples

```
.data idsdata
  + time        v1dat
  + 0           0
  + 1u          0
  + 1.1u        5
  + 2u          5
.enddata
.tran 0.1u 2u
.measure tran v1fit err1 v1dat v(1)
```

This example computes the difference between the externally supplied data curve v1dat and the simulated transient analysis response voltage v(1).
This example computes the (relative) difference between the externally supplied data curve $\text{id}_s$ and the simulated MOSFET drain current $\text{id}(\text{m1})$. This is particularly useful in conjunction with optimization for model parameter extraction, where the purpose of the simulation is to select model parameter values which yield the best match between simulated and measured data curves.
.dc

Performs DC transfer analysis to study the voltage or current at one set of points in a circuit as a function of the voltage or current at another set of points. Can also be used for linear or logarithmic sweeps of DC voltage or current; for sweeps of parameters other than voltage and current source values; and for Monte Carlo analysis or optimization.

- Transfer analysis is done by sweeping the source variables over specified ranges, and recording the output.
- Up to three parameters can be specified per .dc command.
- When two or more sources are specified, the last-named source “controls” the sweeping process (see “Examples,” below).
- The specified current or voltage sources must exist—that is, be defined by $i$ or $v$ device statements elsewhere in the input file.
- DC transfer analysis results can be reported with the “.print” (page 137) dc, “.probe” (page 152) dc, and “.macro /eom” (page 107) dc commands.

Syntax

```
/dc swinfo [[sweep] swinfo [[sweep] swinfo]]
```

Refer to “.step” (page 158) for a syntax description of swinfo.

Examples

```
.dc isrc 0 1e-6 0.1e-6
```

Current source isrc is swept from 0 to 1 microampere in 0.1-microampere steps.

```
.dc vin 0 5 0.05 VCC 4 6 0.5
```

Names two voltage sources: vin, to be swept from 0 to 5 volts in 0.05-volt steps, and VCC, to be swept from 4 to 6 volts in 0.5-volt steps. The second source “controls” the sweep: VCC is initially set to 4 volts, while vin is swept over its specified range. Then VCC is incremented to 4.5 volts, and vin is again swept over its range. This process is repeated until VCC reaches the upper limit of its specified range.

```
.data sweep_params
+ vds r2 length
+ 0.0 1k 10u
+ 1.0 500 12u
+ 2.0 100 14u
.enddata
.dc data=sweep_params
```

This example performs a DC sweep defined using a .data statement. There are three sweep steps, and the parameters vds, r2, and length are varied in the sweep. On the first sweep step, vds=0, r2=1000, and length=10u; on the second sweep step, vds=1, r2=500, and length=12u; on the third sweep step, vds=2, r2=100, and length=14u.
.del lib

Used to delete a section from a library file previously included using the ".lib" (page 104) command. The .del lib command is used in ".alter" (page 80) blocks, typically to replace a library section with a different one.

Syntax

```
.del lib filename section
```

- **filename**: Name of the referenced library file. If the referenced filename or path contains a space, enclose the entire path in single or double quotation marks.
- **section**: Name of the library section in `filename` that is to be deleted.

Examples

```
.lib bsim3model.md typical
.alter
.del lib bsim3model.md typical
.lib bsim3model.md fast
.alter
.del lib bsim3model.md fast
.lib bsim3model.md slow
.end
```

T-Spice input such as shown above might be used to run a simulation three times, first with `typical`, then with `fast`, and finally with `slow` model parameters. First, the library section called `typical` is loaded for the first simulation. The second simulation incorporates changes between the first and second `alter` commands, so that the `typical` library section is deleted and replaced with the `fast` library section. Similarly, the third simulation replaces the `fast` library section with the `slow` one.
.end

Signifies the end of the circuit description.

- Any text in the input file after the `.end` command is ignored.
- The `.end` command is optional in T-Spice but is included for compatibility with generic SPICE.

Syntax

```plaintext
.end [comment]
```
.enddata

Signifies the end of a .data statement.

- The .enddata command must accompany a “.data” (page 87) command.

Syntax

.enddata [comment]
.endl

Signifies the end of a library definition.

- The .endl command must accompany a `.lib` (page 104) command.

Syntax

```
.endl [comment]
```
.ends

Signifies the end of a subcircuit definition.

- The .ends command must accompany a `.subckt` (page 162) command.

Syntax

```
.ends [comment]
```
.four

Performs Fourier analysis on transient analysis data.

- Fourier components (magnitude and phase) are computed for a given fundamental frequency and corresponding to a specified number of integer multiples of the fundamental frequency.
- The DC Fourier component is computed, as well as the total harmonic distortion, defined as

\[
\frac{1}{R_1} \cdot \left( \sum_{m=2}^{nfreqs} \frac{R_m}{m} \right)
\]

where \( R_m \) is the magnitude of the \( m \)th Fourier component.

- The .four command is ignored if no .tran command is found.

Syntax

```
.four F list [nfreqs=N] [npoints=P] [interpolate=I]
```

- **F**: Fundamental frequency.
- **list**: Output variables for which the analysis is to be performed. Each of these can be any valid output item from a “.print” (page 137) tran command, including output expressions.
- **N**: Number of frequencies for which Fourier components are determined. (Default: 9.)
- **P**: Number of points over which transient analysis data is interpolated to fit. These points equally divide the analysis interval \((T-J,T)\), where \( T \) is the final time specified on the corresponding “.tran” (page 165) command, and \( J = 1/F \). Increasing \( P \) improves accuracy but increases simulation time and memory use. (Default: 100.)
- **I**: If 0, T-Spice inserts an actual computed time point at each place where a Fourier analysis time point is needed without interpolating transient data to fit on \( np \). If 1, Fourier analysis is based on interpolated data. (Default: 1.)

Examples

```
v1 2 0 sin (4 10 9e6 0 0 20)
v2 1 2 sin (0 3 3e6 0 0 -50)
.tran 1u 10u
.four 3e6 v(1) npoints=1000
```
The formula for \(v(1)\) is:

\[
v(1) = 4 + 10\sin[2\pi(9 \times 10^6 t + 20/360)] + 3\sin[2\pi(3 \times 10^6 t - 50/360)]
\]

(5.2)

The analytic Fourier response for this formula is as follows:

- The DC component is 4.
- The first harmonic (at a frequency of 3 MHz) has a magnitude of 3 and a phase of -50 degrees.
- The third harmonic (at a frequency of 9 MHz) has a magnitude of 10 and a phase of 20 degrees.
- All other harmonics have a zero Fourier component.

The output will be:

```
* BEGIN NON-GRAPHICAL DATA
FOURIER ANALYSIS RESULTS
Fourier components of transient response v(1)

DC component = 4.0011e+000

Harmonic no Frequency<Hz> Fourier comp Normalized FC Phase<deg> Normalized phase
1 3.0000e+006 2.9989e+000 1.0000e+000 -4.9651e+001 0.0000e+000
2 6.0000e+006 2.4224e+002 8.0777e-003 3.5132e+001 8.4783e+001
3 9.0000e+006 9.9949e+000 3.3328e+000 2.0564e+001 7.0215e+001
4 1.2000e+007 3.4401e-002 1.1471e-002 -1.6494e+002 -1.1529e+002
5 1.5000e+007 1.8803e-002 6.2701e-003 -1.6786e+002 -1.1821e+002
6 1.8000e+007 1.0534e-002 3.5126e-003 -1.7085e+002 -1.2120e+002
7 2.1000e+007 8.7570e-003 2.9201e-003 -1.7169e+002 -1.2204e+002
8 2.4000e+007 7.5266e-003 2.5098e-003 -1.7229e+002 -1.2264e+002
9 2.7000e+007 6.3695e-003 2.1385e-003 -1.7283e+002 -1.2319e+002

Total harmonic distortion = 333.3 percent
```

where the DC component is 4.0011e+000, the first harmonic has a magnitude of 2.9989 and a phase of -4.9651e+001, the third harmonic has a magnitude of 9.9949e+000 and a phase of 2.0564e+001.

Note that harmonics one and three have the exponent e+000, while the magnitude of the other harmonics—e-002 or even e-003—is quite small in comparison.
.global

Specifies nodes with global scope.

- Global node names refer to the same nodes both inside and outside subcircuit definitions.
- Ground (0, gnd, gnd!, or ground) is automatically defined to be a global node.

Syntax

```
.global node1 [,,] node2 ...
```
.hdl

Loads a Verilog-A module into the simulator, and optionally declares that the module will be used to simulate all devices of a specified type, level, and version.

When the type, level, and version parameters are used in the .hdl command, T-Spice will use the Verilog-A module as the modeling code for all matching devices, i.e. devices of the specified type which reference a .model of the specified level and version. This capability may be used to either introduce new model levels and versions into T-Spice, or to replace the existing built-in T-Spice models with your own.

Syntax

```
.hdl filename [ modulename [ type=type [ level=level [ version=version ]]]]
```

- **filename**: The Verilog-A file to be loaded. It must exist in the current directory or in the T-SpiceVerilog-A search path. Absolute or relative path names (according to the conventions of the operating system) can be used. If the referenced filename or path contains a space, enclose the entire path in single or double quotation marks.
- **modulename**: The name of a specific module within the file. Only this one module will be loaded into the simulator, and other modules within the file will be ignored.
- **type**: The type of devices that will use this module
- **level**: The level number of the models that will use this module
- **version**: The version number of the models that will use this module

The device type keyword is limited to the following:

<table>
<thead>
<tr>
<th>Type keyword</th>
<th>Device</th>
</tr>
</thead>
<tbody>
<tr>
<td>C or capacitor</td>
<td>Capacitors</td>
</tr>
<tr>
<td>D or diode</td>
<td>Diodes</td>
</tr>
<tr>
<td>J or jfet</td>
<td>JFETs</td>
</tr>
<tr>
<td>R or resistor</td>
<td>Resistors</td>
</tr>
<tr>
<td>M or mosfet</td>
<td>MOSFETs</td>
</tr>
<tr>
<td>Q or bipolar</td>
<td>BJTs</td>
</tr>
<tr>
<td>Z or mesfet</td>
<td>MESFETs</td>
</tr>
</tbody>
</table>

Examples

```
.hdl pll.va
.hdl bsim3v34.va bsim3 type=mosfet level=49 version=3.4
```
The second of the above examples demonstrates how a Verilog-A module can be used instead of the built-in device evaluation code. In this case, a Verilog-A representation of the BSIM3 model will be used instead of T-Spice’s internal BSIM3 analysis code for those MOSFETs which reference a model that is level 49 and version 3.4.

**Note:** If the **level** or **version** is not specified or has a value of zero, then all levels and versions will be matched and simulated using the Verilog-A module.
.ic

Sets node voltages or inductor currents for the duration of a DC operating point calculation.

- DC operating points are calculated by the “.ac” (page 76), “.dc” (page 89), “.op” (page 121), “.tf” (page 164), and “.tran” (page 165) commands.
- The .ic command adds a voltage source present only in DC (not transient) simulations.
- The specified nodes are allowed to float if a transient analysis is subsequently requested in the input file. For further information on transient analysis, see “.tran” (page 165).
- Nodes and devices within subcircuits can be accessed with hierarchical notation in the form xinstance.xinstance.node.
- To set initial guesses for node voltages, use the “.nodeset” (page 119) command.
- .ic commands within subcircuit definition (“.subckt” (page 162)/“.ends” (page 94)) blocks are replicated for each subcircuit instance.

Syntax

```
.ic node=X [[,] node=X ...]
.ic v(node [,node])=X [[,] v(node [,node])=X ...]
.ic i(inductor)=X [[,] i(inductor)=X ...]
```

**node**

Node whose voltage is to be initialized. (Default reference node: ground.)

**inductor**

Inductor whose current is to be initialized.

**X**

Node-to-node or node-to-ground voltage or inductor current value. (Unit: volts or amperes.)

Examples

```
.ic a=5, b=5, c=5
```

Assigns initial voltages of 5 volts (relative to ground) to nodes a, b, and c.

```
.ic v(a,b)=5
```

Sets the initial voltage between nodes a and b to 5.
The `.if`, `.elseif`, `.else`, and `.endif` conditional statements may be used in a netlist to control which simulation commands, device statements, and device models will be included in the simulation.

The `.if` and `.elseif` commands have required conditional statements which will be evaluated to either true or false (non-zero or zero). Processing will then proceed, according to the conditional value. The first condition block which evaluates to true will be the selected block, or the `.else` block will be evaluated if no other blocks are true.

- Conditional statements may be nested.
- You can have an unlimited number of `.elseif` statements in your conditional.
- Parameter assignments that are contained within a condition’s statement block do not effect the condition evaluation.
- Statements that are part of a condition statement block are only evaluated if and when the containing condition statement is evaluated to true.

Syntax

```
.if condition1
  < statement block1 >
[ .elseif condition2 ]
  < statement block2 >
[ .elseif condition3 ]
  < statement block3 >
[ .elseif ... ]
...
[ .else ]
  < statement blockn >
.endif
```

- `condition` A value or an expression, enclosed in parentheses or quotes, which will be evaluated to control the selection of one of the statement blocks. Any non-zero expression value will be considered true, and a zero value is false.
- `statement block` Any valid T-Spice statements

Examples

The simplest example of a conditional statement has a single conditional `.if` block:

```
.param debug=1
.if (debug)
  .options echo=1
  .options verbose=2
  .options list nomod=0 node
.endif
```
An example which demonstrates all types of conditional statements:

```plaintext
.if (technology==49 && fast)
   .lib mos49.md FF
.if (technology==49 && fast==0)
   .lib mos49.md TT
.elseif (technology==53)
   .lib mos53.md TT
.else
   .lib mos2.md TT
.endif
```
.include

Includes the contents of the specified file in the input file.

- The .include command can be nested (included files can include other files, and so on) as deeply as hardware and operating system constraints permit.

Syntax

.include filename

filename

The file to be included. It must exist in the current directory or in the T-Spice search path. Absolute or relative path names (according to the conventions of the operating system) can be used. If the referenced filename or path contains a space, enclose the entire path in single or double quotation marks.
.lib

Within a SPICE or included file, specifies a library file or section to be included. Within a library file, indicates the beginning of a library section.

T-Spice accepts two different library file formats. The .lib command is used to access library files of both formats. It is also used to delimit library sections within library files if the first library format is used. Specific model and subcircuit definitions are read in only if needed.

Syntax

*Library File Format I:*

The first T-Spice library file format is a sequence of library sections. Each library section begins with a .lib command and ends with a .endl command. The .lib command assigns a name to each section. Within each library section any sequence of SPICE circuit elements or commands may occur.

When .lib is used with both a file name and section name, it is equivalent to .include except that only the part of the file within the specified library section is included.

```
.lib filename [section]
```

*filename* The file to be included. It must exist in the current directory or in the T-Spice search path. If the referenced filename or path contains a space, enclose the entire path in single or double quotation marks.

*section* If specified, designates a section of the library file to be searched.

Examples

A file *test.lib* might contain:

```
.lib sub1
.subckt s1 a b
rl a b 1k
.ends
.endl

.lib sub2
.subckt s2 a b
rl a b 2k
.ends
.endl

.lib sub3
.subckt s3 a b
rl a b 3k
.ends
.endl

The command:

.lib test.lib sub2
```
would cause T-Spice to include the library section \texttt{sub2} and therefore the definition for subcircuit \texttt{s2}.

\textbf{Library file format II:}

The second T-Spice library file format consists simply of a sequence of \texttt{.model} commands and \texttt{.subckt} definition blocks. A library file of this second format may be included in a simulation by using the \texttt{.include} command in a main input file or included file. T-Spice will search the specified file for device model and subcircuit definitions if they are not found in the main input file or included files, and read in only those that are needed.

\texttt{.include filename}

\texttt{filename}

The library file to be searched for \texttt{“.model”} (page 116), \texttt{“.param”} (page 131), and \texttt{“.subckt”} (page 162) definitions. The file must exist in the current directory or in the T-Spice search path.

If the referenced filename or path contains a space, enclose the entire path in single or double quotation marks.

\textbf{Examples}

A file \texttt{test2.lib} might contain:

\begin{verbatim}
.subckt s1 a b
r1 a b 1k
.ends

.subckt s2 a b
r1 a b 2k
.ends

.subckt s3 a b
r1 a b 3k
.ends
\end{verbatim}

Suppose the main input file contains:

\begin{verbatim}
.include test2.lib
x1 1 0 s3
\end{verbatim}

The \texttt{.include} command would cause the file \texttt{test2.lib} to be searched for model and subcircuit definitions. The instance \texttt{x1} references subcircuit \texttt{s3}, causing T-Spice to read and include the subcircuit definition for \texttt{s3} in \texttt{test2.lib}. Assuming that subcircuits \texttt{s1} and \texttt{s2} are not referenced elsewhere, their definitions in \texttt{test2.lib} would not be read in.
.load

Input the contents of the specified file. The file presumably was created using the ".save" (page 155) command, and contains either ".ic" (page 100) or ".nodeset" (page 119) commands for restoring the bias point of the circuit. The .save and .load commands can be used in combination to reduce simulation time by performing a compute-intensive operating point calculation once, saving the bias information, and then using the .load command in subsequent simulations to initialize the circuit to that state.

Syntax

.load [file=filename]

filename Name of the file to be read. If the file parameter is not entered, then the filename is derived from the simulation input filename, with a .ic file extension.

Examples

.load file=baseline.ic
.macro /.eom

.macro is synonymous with .subckt, and .eom is synonymous with .ends.

The .macro ... .eom naming convention for defining subcircuits is provided for compatibility with other simulators which use this syntax rather than .subckt ... .ends.
.malias

Assigns an alias to a model name which was defined in a .model command. Device references to model names may use either the original model name or the alias name throughout the circuit netlist.

Alias name assignments may also refer to binned model base names. That is, given binned models *nch.1, nch.2, nch.3*, etc. you may declare an alias name *nmosmodel* for model name *nch*.

**Syntax**

```
.malias ModelName=AliasName1 [ AliasName2 [ AliasName3 [ ... ] ] ]
```

**Examples**

```
.model nch.1 nmos level=49 ...
.model nch.2 nmos level=49 ...
.malias nch=nmosmodel
```

or

```
.model dio d ...
.malias dio=zenerdiode
```
.measure

Used to compute and print electrical specifications of a circuit, such as delay between signals, rise and fall times, and minimum and maximum values of a signal. Also used for optimization in conjunction with the following commands:

- `.ac` (page 76)
- `.dc` (page 89)
- `.connect` (page 86)
- `.step` (page 158)
- `.tran` (page 165)

For parameter sweeps, T-Spice generates a separate output section plotting measurement results versus swept parameter values. A data set can be used with the error measurement syntax of `.measure` to compute differences between simulation output curves and corresponding external data. In this case, `out1` or `out2` in the error function measurement syntax may refer to a column name of a `.data` statement.

A `.measure` command within a `.subckt` block is replicated for each instance of the subcircuit.

For optimization, you can use `.optgoal` instead of `.measure` with the `goal` and `minval` parameters to set the minimum value for the denominator in the error expression and the scalar value that specifies the relative importance of two or more measurements.

The numerical format of `.measure` command outputs are controlled by the `ingold` option where 0 yields engineering notation and 1 yields scientific notation. See “ingold” (page 329).

T-Spice saves .measure outputs to a separate .measure file for easy access and script parsing.

---

**Note:**

When the “.options” (page 125) `autostop` field is set to 1, T-Spice automatically terminates any transient analysis when all `.measure` results have been found. The `autostop` option does not affect preview transient analyses.

---

**Syntax**

The `.measure` command syntax has several formats, each of which is described below. Each `.measure` command should be used in conjunction with DC transfer, AC, data, step, or transient analysis.

The general syntax of the `.measure` command is:

```
.measure {dc|ac|tran} result list [goal=goal] + [minval=minval] [weight=weight] [off]
```

- `dc | ac | tran` Denotes the analysis type (`dc`, `ac`, or `tran`) for which the measurement is to be done. For `dc` analysis, the independent variable is a swept parameter. For `ac` analysis, the independent variable is frequency. For transient analysis (`tran`), the independent variable is time.
Chapter 5: Simulation Commands

... .measure

Note:
The .measure keyword can be abbreviated to .meas.

Trigger/Target Measurements

The trigger/target format of the .measure command is used to make independent variable (time, frequency, or swept parameter) difference measurements. The trigger and target specifications determine the beginning and end, respectively, of the measurement. The value of the measurement is the difference in the independent variable value between the trigger and the target. Common examples of trigger/target measurements include delay time, rise time, fall time, and bandwidth measurements.

The syntax of the parameters field for trigger/target measurements is:

```
trig outvar val=val [td=td] (cross=cross | rise=rise | fall=fall)
+       targ outvar val=val [td=td] (cross=cross | rise=rise | fall=fall)
```

or

```
trig at=at_value targ outvar val=val [td=td] (cross=cross | rise=rise | fall=fall)
```

**at_value** Specifies an explicit independent variable value for the trigger.
Note:
For a particular trigger or target, only one of cross, rise, or fall may be specified.

For a particular trigger or target, only one of cross, rise, or fall may be specified.

Trigger/target measurement output reports the independent variable value difference between the trigger and the target, as well as the independent variable values at the trigger and target. The measurement result may be negative if the target independent variable value is less than the trigger independent variable value.

For syntax examples, see “Trigger/Target Example” on page 115.

Signal Statistics Measurements

Signal statistics measurements are used to perform data reduction operations on signals. T-Spice can compute average, RMS (root-mean-square), minimum, maximum, and peak-to-peak values for a signal. In addition, T-Spice can report the independent variable (argument) value at the minimum or maximum of a signal.

The syntax of the parameters field for signal statistics measurements is:

type outvar [from=from] [to=to] [output=output]

type

Specifies the type of measurement, and is one of the following:

- amax computes the independent variable value at the point where the signal’s maximum value is reached.
For syntax examples, see “Signal Statistics Example” on page 115.

Find-When and Derivative Measurements

Find-when measurements are used to measure values of dependent or independent output variables when some specific event occurs. The event specification is similar to the trigger specification in trigger/target measurements: the event occurs when a signal crosses a value, or a signal crosses another signal, or when a certain independent variable “at” value is reached. If a “find” signal is specified, the measurement output is the value of the find signal at the event. If no “find” signal is given, then the measurement result is the independent variable value at the event.

Find-when measurements can also be used to compute derivatives of functions. If the find keyword is replaced with derivative, the derivative of outvar is computed as the measurement result.

The syntax of the list field for find-when measurements is:
[find outvar1 | derivative outvar1] when outvar2=val [td=td] {cross=cross | rise=rise | fall=fall}

or

[find outvar1 | derivative outvar1] when outvar2=outvar3 [td=td] {cross=cross | rise=rise | fall=fall}

or

{find outvar1 | derivative outvar1} at=at_value

Note: For a particular trigger or target, only one of cross, rise, or fall may be specified.

- **at_value** Specifies an explicit independent variable value for the event.

- **outvar1** Specifies the output signal to be evaluated as the measurement result at the event of interest. This can be any output plot item that is legal on a .print command for the appropriate analysis type. For .measure ac commands, .print noise plot items are allowed. outvar1 may be an output expression enclosed in single quotes.

- **outvar2** Specifies an output signal to be evaluated for locating the measurement event. This can be any output plot item that is legal on a .print command for the appropriate analysis type. For .measure ac commands, .print noise plot items are allowed. outvar2 may be an output expression enclosed in single quotes.

- **val** Specifies the value of outvar2 at which the event counter for crossings, rises, or falls is incremented.

- **outvar3** Specifies a second output signal to be evaluated for locating the measurement event. This can be any output plot item that is legal on a .print command for the appropriate analysis type. For .measure ac commands, .print noise plot items are allowed. outvar3 may be an output expression enclosed in single quotes.

- **td** Specifies a time delay before the measurement is enabled and crossings, rises, and falls are counted. Default: 0.

- **cross** Indicates which occurrence of the event crossing is to be used for the measurement. A crossing occurs when outvar2 takes on the value val or the value of outvar3. The special syntax cross=last indicates that the last crossing is to be used.

- **rise** Indicates which occurrence of the trigger or target rise crossing is to be used for the measurement. A rise crossing occurs when outvar2 takes on the value val or the value of outvar3 while increasing. The special syntax rise=last indicates that the last rise crossing is to be used.

- **fall** Indicates which occurrence of the trigger or target fall crossing is to be used for the measurement. A fall crossing occurs when outvar2 takes on the value val or the value of outvar3 while decreasing. The special syntax fall=last indicates that the last fall crossing is to be used.

Expression Evaluation Measurements

T-Spice can compute expressions that are functions of previous `.measure` command results. The syntax of the `parameters` field for expression evaluation measurements is:

```
param='expression'
```

where `expression` is an algebraic expression involving `.param` parameter values, subcircuit parameter values, and previous `.measure` result names. The expression may not contain plot items such as node voltages or branch currents. The expression may contain the same operators and functions as used in `.print` output expressions.

Error Function Measurements

The error function measurement reports a relative difference between two output variables. Four methods for calculating this error are available.

The syntax of the `parameters` field for error function measurements is:

```
errtype out1 out2 [from=from] [to=to]
```

- **errtype**: Specifies the method of computing the total error. Must be one of `err`, `err1`, `err2`, `err3`.
- **out1, out2**: Specifies the output signals to be compared. These can be any output plot items that are legal on a `.print` command for the appropriate analysis type. For `.measure ac` commands, `.print noise` plot items are allowed. `out1` and `out2` may be output expressions enclosed in single quotes.
- **from**: Specifies the value of the independent variable (time, frequency, or sweep parameter) at the beginning of the measurement. The default is the beginning of the analysis.
- **to**: Specifies the value of the independent variable (time, frequency, or sweep parameter) at the end of the measurement. The default is the end of the analysis.

The error computation depends on the `errtype` and is as follows:

- **err, err1**: The error is the RMS value of the relative difference between the two signals, normalized to the length of the measurement interval.
- **err2**: The error is the integral over the measurement interval of the absolute value of the relative difference of the two signals.
- **err3**: The error is the integral over the measurement interval of the relative difference of the logs of the two signals.

The relative difference of the two signals is defined as:

```
(out1-out2)/max(minval, |out1|+|out2|)
```
The relative difference of their logs is defined as the absolute value of:

\[
\log|\frac{|\text{out1}|}{\max|\text{minval}, \text{out2}|}|/\log\max|\text{minval}, |\text{out1}|+|\text{out2}||
\]

For syntax examples, see “Error Function Example” on page 115.

Examples

**Trigger/Target Example**

```
.measure tran delaytime trig v(1) val=2.5 fall=3
+      targ v(2) val=2.5 rise=3
```

measures the time delay from the third falling edge of signal \(v(1)\) to the third rising edge of signal \(v(2)\). The measurement begins when \(v(1)\) falls through 2.5V for the third time, and ends when \(v(2)\) rises through 2.5V for the third time.

```
.measure tran risetime trig v(1) val=0.5 rise=1
+      targ v(1) val=4.5 rise=1
```

measures the first rise time of the voltage at node 1.

**Signal Statistics Example**

```
.measure ac maxgain max vm(out)
```

measures the maximum value of \(vm(out)\) over the frequency range covered during an AC analysis.

```
.measure ac resfreq amax vm(out)
```

measures the frequency at which the maximum value of \(vm(out)\) is achieved.

```
.measure ac phase_at_resonance max vm(out) output=vp(out)
```

measures the phase \(vp(out)\) at the frequency where \(vm(out)\) is at its maximum.

**Find-When and Derivative Example**

```
.measure tran v1 find v(1) when v(2)=2 cross=1
```

measures the voltage at node 1 when the voltage at node 2 crossed 2V for the first time.

```
.measure ac f1 when vm(out)=1
```

measures the frequency at which the voltage gain at node out is 1.

```
.measure tran d1 derivative v(1) at=100ns
```

measures the derivative of \(v(1)\) with respect to time at 100ns.

**Error Function Example**

```
.measure tran v1v2 err1 v(1) v(2)
```

measures the difference between the signals \(v(1)\) and \(v(2)\).
.model

Specifies device model parameters to be used by one or more devices.

Syntax

```
.model modelname type [level=L][parameter=value [parameter=value [...]]]
[ako: akomodel]
```

- **modelname**: Model name.
- **type**: Device type (see below).
- **L**: Model level, required for device models with multiple levels.
- **parameter**: The parameter list is predefined for each standard device model. (See the chapter “Device Models” on page 365.)

**type** is one of the following:

- **c**: Capacitor
- **cpl**: Coupled transmission line.
- **csw**: Current-controlled switch element.
- **d**: P-N diode.
- **npn**: NPN-type BJT.
- **pnp**: PNP-type BJT.
- **njf**: N-type JFET.
- **pjf**: P-type JFET.
- **nmf**: N-type MESFET.
- **opt**: Controls the optimization algorithm. For additional information, see “Optimization Algorithm Parameters,” below.
- **pmf**: P-type MESFET.
- **nmos**: N-type MOSFET.
- **pmos**: P-type MOSFET.
- **r**: Two-terminal resistor (or three-terminal resistor if a capacitance is specified).
- **sw**: Voltage-controlled switch element.

**Optimization Algorithm Parameters**

The T-Spice optimization algorithm is controlled using the .model command with a model type of opt. The syntax is:
.model name opt [parameter=value [parameter=value [...]]] [ako: akomodel]

**modelname**
Model name matched with the model name specified in an analysis command

**parameter**
Model parameter name

**value**
Value assigned to the model parameter

**akomodel**
Another opt model statement. If akomodel is specified, this model is considered to be “a kind of” the model defined with name akomodel; that is, all model parameters defined in the akomodel .model statement are included in this model unless overridden in this model.

Valid alternatives for value are listed below:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>cendif</td>
<td>Gradient value below which more accurate derivative computation methods are used to compute the gradient.</td>
<td>1e-9</td>
</tr>
<tr>
<td>close</td>
<td>Estimate of how close the optimization parameters’ initial value estimates are to the solution. close is a multiplier for computing new parameter estimates. Larger values result in larger steps toward the solution.</td>
<td>0.001</td>
</tr>
<tr>
<td>cut</td>
<td>Modifies close from one iteration to the next. If an iteration was unsuccessful, close is divided by cut; if an iteration is successful, close is multiplied by cut squared.</td>
<td>2</td>
</tr>
<tr>
<td>difsiz</td>
<td>Determines the increment in a parameter value used to compute numerical derivatives. The increment used is difsiz*max(value, parmin), where value is the parameter value. If delta is specified on the .param command, then delta is used as the increment.</td>
<td>1e-3</td>
</tr>
<tr>
<td>grad</td>
<td>Convergence tolerance for the gradient. If the gradient is less than grad, then convergence may have been achieved, depending on the outcome of the additional relin and relout tests.</td>
<td>1e-6</td>
</tr>
<tr>
<td>itropt</td>
<td>Maximum number of optimization loop iterations.</td>
<td>20</td>
</tr>
</tbody>
</table>
Level

Optimization method used. Level=1 selects a modified Levenberg-Marquardt algorithm.

max

Upper limit for close

parmin

Used in increment selection for derivative computation; see difsiz parameter, above

relin

Relative input parameter variation for convergence. If all optimization parameters (defined with .param) vary by less than this amount (0.1% by default) from one iteration to the next, convergence is declared.

relout

Relative output parameter variation for convergence. If the total error defined by measurement results varies by less than this amount from one iteration to the next, convergence is declared.

Note:

When no model parameters are specified, all models take on their default values.

Examples

.model nmos nmos
+ Level=2 Ld=0.0u Tox=225.00E-10
+ Nsub=1.066E+16 Vto=0.622490 Kp=6.326640E-05
+ Gamma=.639243 Phi=0.31 Uo=1215.74
+ Uexp=4.612355E-2 Ucrit=1746677 Delta=0.0
+ Vmax=177269 Xj=.9u Lambda=0.0
+ Nfs=4.55168E+12 Neff=4.68830 Nss=3.00E+10
+ Tpg=1.000 Rsh=60 Cgso=2.89E-10
+ Cgdo=2.89E-10 Cj=3.27E-04 Mj=1.067
+ Cjsw=1.74E-10 Mjsw=0.195

Specifies the parameters for an n-type MOSFET model called nmos.
.nodeset

Sets an “initial guess” for the iterative DC operating point calculation.

- DC operating points are calculated by the “.ac” (page 76), “.dc” (page 89), “.op” (page 121), “.tf” (page 164), and “.tran” (page 165) commands.
- DC operating points are calculated by iteration, and .nodeset sets starting points for iteration. Convergence properties can be very sensitive to the initial guess; .nodeset can be used (1) to enable convergence for difficult-to-converge circuits and (2) to cause T-Spice to converge to one particular solution if more than one solution exists.
- After numndset iterations, or when the convergence criteria have been met, the specified nodes are allowed to float.
- Nodes and devices within subcircuits can be accessed with hierarchical notation in the form xinstance.xinstance.node.
- To set node voltages for the duration of the operating point calculation, use the “.ic” (page 100) command.
- .nodeset commands inside subcircuit definition blocks are replicated for each subcircuit instance.

Syntax

```plaintext
.nodeset node=X [, node=X ...]  
.nodeset v(node)=X [, v(node)=X ...]
```

- **node** Node to be initialized.
- **X** Node voltage relative to ground. (Unit: volts.)

Examples

```
.nodeset n1=5V n2=2  
.nodeset v(n1)=5V v(n2)=2
```

The two examples are identical except for syntax.
.noise

Computes the effect of circuit noise on output voltage in conjunction with AC analysis.

- If the “.ac” (page 76) command is missing from the input file, the .noise command is ignored.
- Noise analysis is performed at the same frequencies as specified by the “.ac” (page 76) command.
- Noise models take the form of frequency-dependent mean-square currents (since the underlying phenomena are “random”) generated by adding a current source to the circuit for each modeled noise source.
- Noise sources at different points in the circuit are uncorrelated.
- Noise models are available for resistors and semiconductor devices (diodes, BJTs, JFETs, MESFETs, and MOSFETs). Semiconductor device models may contain noise model parameters which affect the size of noise sources.
- Verilog-A devices may also contain noise sources.
- Noise analysis results can be reported with the “.print” (page 137) noise command.
- Use the list options to display the worst noise offenders in descending order along with their names and hierarchy. You can also display the percentage a device contributes to the total noise.

Syntax

```
.noise v(node1 [, node2]) source interval
.noise v(node [, ref]) source interval
   [listcount=n][listsource=0|1|false|true] [listfloor=floor]
```

- `node1` Output node.
- `node2` Reference node. (Default: ground.)
- `source` Input voltage or current source, at which noise can be considered to be concentrated for the purposes of estimating the equivalent noise spectral density.
- `interval` Report interval. A noise report will be printed to the simulation log which lists every device and the noise contribution and noise components. This report will be printed for the first frequency and each `interval` frequency. (Default: 0)
- `listcount` Lists only the top n noise contributors.
- `listsource` Declares whether all noise components should be listed for each device; e.g. ID, FN, RS, etc. Alternatively, only device total noise and percent contribution will be listed. The default is 1.
- `listfloor` Lists only the noise contributors that have a total noise greater than floor V^2/Hz.
.op

Performs a DC operating point calculation and outputs all node voltages and voltage source currents.

- DC operating points are calculated on the assumption that there are no charge effects in the system: capacitors are open and inductors are shorted.
- The “.hdl” (page 98) command can be used to impose initial conditions on nodes. Initial conditions are represented by voltage sources present for the duration of the DC operating point calculation, and removed for transient simulations.
- The “.nodeset” (page 119) command can be used to set initial guesses for the iterative solution process for DC operating points.
- The results of the .op command are written automatically to the specified output file. Other results can be reported with the “.print” (page 137) dc command.
- Small-signal transfer function data can be reported with the “.tf” (page 164) command.
- Small-signal parameters are automatically reported to the output file. You can specify a separate output file for small-signal data with the “.acmodel” (page 79) command. The command .acmodel {} disables small-signal parameter reporting. Reporting is also disabled if the simulation model has more than 1000 nodes.

Syntax

```
.op [noprint]
```

noprint

Turns off automatic .op output.
.optgoal

Sets optimization goals. Allows the same measurement to be used in different optimization runs with different goals and weight values.

Note that during an optimization run, all .optgoal commands with the same optname are used as optimization results, in addition to any measurements specified in the results list on the sweep optimize syntax of a .step or one of the analysis commands. (See the following example.)

If the .measure command does not exist in the netlist, an error message will be returned.

The formula for optimization functions is:

\[
\sum_{k} W_k \cdot \frac{(G_k - M_k)}{\max(\text{minval}, |G_k|)}^2
\]

where \( W_k \) is the weight, \( G_k \) is the goal, and \( M_k \) is the measurement value.

Syntax

```
.optgoal optname measname=goal [minval=minval] [weight=weight]
```

- **optname**: Name of the optimization run.
- **measname**: A .measure result to be used as the goal.
- **goal**: Optimization goal value.
- **minval**: The minimum denominator value for the optimization error computation. Defaults to the value in the .measure command identified by measname, or if not specified there, to 1.0e-12.
- **weight**: A relative importance of the optimization goal with respect to other goals for the same optimization run. Defaults to the value in the .measure command identified by measname, or if not specified there, to 1.

Examples

```
.ac dec 10 1 100k sweep optimize=opt1 results=gain model=optmod
.measure ac gain max vdb(out) goal=30
.measure ac bandwidth when vdb(out)=10
.optgoal opt1 bandwidth=5k
```

This results in both the gain and bandwidth measurements contributing to the overall optimization goal.
.optimize

Invokes an optimization run using parameters and goals specified using .paramlimits and .optgoal commands with the same optname.

.ac, .dc, .step and .tran use the parameter analysis to identify analysis commands from the .optimize command. However, it is possible to avoid assigning an analysis name. If the analysis name on the .optimize command is the name of an analysis type ("ac", "tran", "dc", or "step"), and no analysis of that name exists, T-Spice will perform the optimization on the first analysis of that type. (See “Examples” on page 123.)

Syntax

```
.optimize optname model=modelname analysis=analysisname
```

<table>
<thead>
<tr>
<th>optname</th>
<th>Name of the optimization run.</th>
</tr>
</thead>
<tbody>
<tr>
<td>modelname</td>
<td>Refers to a .model command of type opt which specifies optimization algorithm parameters such as iteration count limits and convergence tolerances.</td>
</tr>
<tr>
<td>analysisname</td>
<td>Identifies a .step, .ac, .dc, or .tran command of the same name that will be performed to evaluate the measurements for the optimization.</td>
</tr>
</tbody>
</table>

Examples

```
.ac dec 10 1 100k analysisname=ac1
.optimize opt1 model=optmod analysisname=ac1
```

invokes an optimization around an AC analysis. This AC analysis optimization syntax is equivalent to:

```
.ac dec 10 1 100k
.optimize opt1 model=optmod analysisname=ac
.param r=1k c=1u
.r1 1 0 'r'
c1 1 0 'c'
.options autostop reltol=1e-6
.ic v(1)=1
.tran 0.1m 100m
.print tran v(1)
.measure tran decaytime when v(1)=0.5
```

Optimization commands:

```
.optimize opt1 model=optmod analysisname=tran
.model optmod opt level=1 itropt=40
.optgoal opt1 decaytime=300u
.paramlimits opt1 r minval=10 maxval=100k
.paramlimits opt1 c minval=0.01u maxval=100u
.end
```

This is a transient analysis of a simple RC circuit. The .measure command measures the amount of time it takes for the voltage at node 1 to decay to half its initial value. Theoretically, this decay time is equal to \( R \times C \times \ln(2) \). The .optimize command invokes an optimization run called opt1. The .optgoal
command sets the optimization goal (300 usec) for the decay time, and the .param limits commands specify ranges for the R and C parameters.

During the optimization run, R and C can be varied within their ranges in order to achieve the decay time goal of 300 microseconds. The output for this simulation is as follows:

Optimized parameter values:

\[
\begin{align*}
    r &= 6.5783 \times 10^2 \\
    c &= 6.5775 \times 10^{-7}
\end{align*}
\]

Note that the optimized R and C values achieve the optimization goal of 300u. This is consistent with theoretical prediction: for the optimized R and C values of 657.83 Ohms and 657.75 nanofarads, the theoretically predicted decay time is 299.92 microseconds.
.options

Sets global simulation options.

Syntax

```
.options field=X [field=X ...]
```

Fields that toggle actions on or off can be specified as `true/false, t/f, 1/0, or yes/no`. Specifying a `true/false` field without a value automatically sets the field to `true`.

Option fields are described in detail in the following chapter, “Simulation Options” on page 227. The following tables list a summary of simulation options; click on the option name for a full description.

**Accuracy and Convergence Options**

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>“absi</td>
<td>abstol” (page 229)</td>
<td>Maximum allowed RMS of residual branch currents when <code>kcltest = true.</code></td>
</tr>
<tr>
<td>“absv</td>
<td>vntol” (page 230)</td>
<td>Maximum absolute node voltage change allowed between iterations when <code>kvtest = true.</code></td>
</tr>
<tr>
<td>“accurate” (page 231)</td>
<td>Triggers changes to other option settings to maximize simulation accuracy.</td>
<td>false</td>
</tr>
<tr>
<td>“bypass” (page 233)</td>
<td>Controls the diode and transistor bypass algorithm</td>
<td>true</td>
</tr>
<tr>
<td>“bytol” (page 234)</td>
<td>Sets the relative tolerance for the bypass algorithm terminal voltage values</td>
<td>0.0</td>
</tr>
<tr>
<td>“cshunt” (page 235)</td>
<td>Capacitance added from each node to ground.</td>
<td>0.0 F</td>
</tr>
<tr>
<td>“dchomotopy” (page 236)</td>
<td>Algorithm used to correct DC operating point non-convergences. Can be set to none, source, gmin, pseudo, or all.</td>
<td>all</td>
</tr>
<tr>
<td>“dcmethod” (page 238)</td>
<td>Default method for solving a DC operating point problem. Can be set to standard, source, gmin, or pseudo.</td>
<td>standard</td>
</tr>
<tr>
<td>“dcstep” (page 239)</td>
<td>Controls the conductance added across the terminals of each capacitor during DC operating point computation. ($g=c/dcstep$, where $c$ is the device capacitance.)</td>
<td>0.0</td>
</tr>
<tr>
<td>“extraiter[ations]</td>
<td>newtol” (page 240)</td>
<td>Number of additional iterative steps to calculate after convergence criteria have been met.</td>
</tr>
<tr>
<td>“fast” (page 241)</td>
<td>Triggers changes to other options settings to maximum simulation speed.</td>
<td>false</td>
</tr>
<tr>
<td>“gmin” (page 242)</td>
<td>Conductance added in parallel with all $pn$ junctions during transient analysis.</td>
<td>$1 \times 10^{-12}$ $\Omega^{-1}$</td>
</tr>
<tr>
<td>“gmindc” (page 243)</td>
<td>Conductance added in parallel with all $pn$ junctions during DC operating point analysis.</td>
<td>$1 \times 10^{-12}$ $\Omega^{-1}$</td>
</tr>
</tbody>
</table>
### Timestep and Integration Options

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>**absdv</td>
<td>absvar** (page 263)</td>
<td>Threshold absolute voltage change between two consecutive time steps; used to calculate voltage variance when <code>lvltim</code>=1</td>
</tr>
<tr>
<td>**absq</td>
<td>chgtol</td>
<td>chargetol** (page 264)</td>
</tr>
<tr>
<td><strong>ft</strong> (page 266)</td>
<td>Fraction by which the timestep is reduced if a transient analysis solution does not converge within <code>numnt</code> iterations.</td>
<td>0.25</td>
</tr>
</tbody>
</table>
### Model Evaluation Options

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>“dcap” (page 283)</td>
<td>Model selector for calculating depletion capacitances.</td>
<td>2</td>
</tr>
<tr>
<td>“dccap” (page 284)</td>
<td>Flag to compute device charge and capacitance values in DC analysis</td>
<td>false</td>
</tr>
<tr>
<td>“defad” (page 285)</td>
<td>Default MOSFET drain diode area.</td>
<td>0.0 m²</td>
</tr>
<tr>
<td>“defas” (page 286)</td>
<td>Default MOSFET source diode area.</td>
<td>0.0 m²</td>
</tr>
</tbody>
</table>
### Linear Solver Options

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>“linearsolver”</td>
<td>Selects the linear equation solver — <strong>best</strong>, <strong>sparse</strong>, or <strong>superlu</strong>.</td>
<td><strong>best</strong></td>
</tr>
<tr>
<td>“pivtol”</td>
<td>Minimum pivoting tolerance for real matrices.</td>
<td>$1 \times 10^{-14}$</td>
</tr>
<tr>
<td>“zpivtol”</td>
<td>Minimum pivoting tolerance for complex matrices.</td>
<td>$1 \times 10^{-6}$</td>
</tr>
</tbody>
</table>

### General Options

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>“autopowerup”</td>
<td>Enables transient powerup mode for nonconvergent transient operating point calculations.</td>
<td><strong>true</strong></td>
</tr>
<tr>
<td>“autostop”</td>
<td>Terminates transient analysis after all <strong>.measure</strong> results have been found.</td>
<td><strong>false</strong></td>
</tr>
<tr>
<td>“casesensitive”</td>
<td>Case sensitivity for names of models, subcircuits, library sections, parameters, and nodes.</td>
<td><strong>false</strong></td>
</tr>
<tr>
<td>“compatibility</td>
<td>spicemode”</td>
<td>Specifies input syntax and option setting compatibility with other simulators - Berkeley SPICE, HSPICE, or PSPICE.</td>
</tr>
</tbody>
</table>
### Field Description Default

**“conncheck” (page 311)** | Enables connectivity checking. | true |
**“macmod” (page 312)** | Enables substitution of subcircuit definitions for missing MOSFET model references, and vice-versa. | 0 |
**“parhier” (page 313)** | Establishes the scoping algorithm for selection of parameter values in a hierarchical design. | local |
**“persist” (page 315)** | Instructs T-Spice to continue simulation when the specified levels of warnings or errors are generated. | 1 |
**“search” (page 316)** | Search path for library and include files. | |
**“spice” (page 317)** | Changes other option settings to be compatible with Berkeley SPICE. | false |
**“threads” (page 319)** | Enables parallel processing. | 0 |

### Output Options

**“acct” (page 321)** | Tracks and reports iteration counts and other accounting statistics. | false |
**“acout” (page 322)** | Calculation method for AC magnitude/phase differences. | 1 |
**“brief” (page 323)** | Minimizes the amount of diagnostics printout which is written to the simulation status window. | false |
**“captab” (page 324)** | Lists capacitances for each node in the netlist. | false |
**“csv” (page 325)** | Generates output in CSV format. | false |
**“dnout” (page 326)** | Selects noise spectral density units. | 0 |
**“echo” (page 327)** | Causes T-Spice to print each line of input to the error log as it is read. | false |
**“expert” (page 328)** | Produces a listing of node and device convergence residual information. | false |
**“ingold” (page 329)** | Controls the format of numbers printed in the AC small-signal output and the device listings. (0=engineering format, 1=g format, 2=e format) | 0 |
**“list” (page 330)** | Produces a listing of all circuit devices. | false |
**“maxmsg” (page 331)** | Sets the maximum number of duplicate warning message printouts. | 5 |
**“node” (page 335)** | Prints a node cross-reference table. | false |
**“nomod” (page 336)** | Controls the printout of diode and transistor models. Set nomod to 1 (true) to disable printout. | 1 |
**“numdgt” (page 337)** | Minimum number of decimal places included in each .print output value. | 4 |
**“opts” (page 338)** | Prints the settings of all control options. | false |
### Probing Options

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>“probei” (page 349)</td>
<td>Includes device terminal current values in the output data from .probe and .print (when used without arguments).</td>
<td>false</td>
</tr>
<tr>
<td>“probeq” (page 352)</td>
<td>Includes device terminal charge values in the output data from .probe and .print (when used without arguments).</td>
<td>false</td>
</tr>
<tr>
<td>“ probev” (page 354)</td>
<td>Includes node voltage values in the output data from .probe and .print (when used without arguments).</td>
<td>false</td>
</tr>
</tbody>
</table>
.param

Defines and assigns values to parameters, or creates a user-defined function.

- Parameters can be used in expressions to replace numeric values.
- .param statements are sequential. A parameter must be defined before it is used in the expression of another parameter value.
- User-defined functions are very similar to the set of built-in algebraic functions. User-defined functions may take any number of arguments, and are defined using an algebraic expression which performs operations on the function arguments.
- The scope of defined parameters extends to files referenced by “.if ... / .elseif ... / .else / .endif” (page 101) commands.
- Parameters placed in subcircuit definition blocks are valid only within that subcircuit definition, and override global parameters of the same name.
- Parameters placed outside subcircuit definition blocks are valid globally.
- All parameter values that are expressions must be enclosed in single quotes. With version 16, device parameters and expressions and .param and .model values can also be delineated with parentheses. The outer parentheses behave as quotes, so you can have embedded parentheses, spaces, and can continue to the next line.

**Note:**
To view a listing of all parameters that are defined in the input files, use the command .option xref.

**Syntax**

```plaintext
.param parameter={X | mc_distribution|opt_limits}[[,]
{X | mc_distribution|opt_limits} ...]```

- **parameter**
  - Parameter name.
- **X**
  - Any number or valid expression. Expressions must be enclosed by single quotes.
- **mc_distribution**
  - Defines probability distributions used in Monte Carlo iterations. For additional information, see Monte Carlo Parameters, below.
- **opt_limits**
  - Defines optimization parameters. For additional information, see “Optimization Parameters” on page 133.

**User-Defined Functions**

The syntax for defining a function is:

```plaintext
```

- **funcname**
  - The name of the user function, which may not be the same as a built-in algebraic function, or the same as a parameter name.
- **arg1 arg2 ...**
  - Function arguments which will be passed into the function from the function reference.
**Monte Carlo Parameters**

For each Monte Carlo iteration, T-Spice will report the values of all expressions evaluated using probability distributions defined by the following syntax:

\[ \text{.param parameter}= \text{unif}(\text{nominal_val}, \text{rel_variation} [, \text{multiplier}]) \]

or

\[ \text{.param parameter}= \text{aunif}(\text{nominal_val}, \text{abs_variation} [, \text{multiplier}]) \]

or

\[ \text{.param parameter}= \text{gauss}(\text{nominal_val}, \text{rel_variation}, \text{sigma} [, \text{multiplier}]) \]

or

\[ \text{.param parameter}= \text{agauss}(\text{nominal_val}, \text{abs_variation}, \text{sigma} [, \text{multiplier}]) \]

or

\[ \text{.param parameter}= \text{limit}(\text{nominal_val}, \text{abs_variation}) \]

- **body** An algebraic expression which solves the functional equation.

- **Monte Carlo Parameters**

  For each Monte Carlo iteration, T-Spice will report the values of all expressions evaluated using probability distributions defined by the following syntax:

  \[ \text{.param parameter}= \text{unif}(\text{nominal_val}, \text{rel_variation} [, \text{multiplier}]) \]

  or

  \[ \text{.param parameter}= \text{aunif}(\text{nominal_val}, \text{abs_variation} [, \text{multiplier}]) \]

  or

  \[ \text{.param parameter}= \text{gauss}(\text{nominal_val}, \text{rel_variation}, \text{sigma} [, \text{multiplier}]) \]

  or

  \[ \text{.param parameter}= \text{agauss}(\text{nominal_val}, \text{abs_variation}, \text{sigma} [, \text{multiplier}]) \]

  or

  \[ \text{.param parameter}= \text{limit}(\text{nominal_val}, \text{abs_variation}) \]

- **parameter** Name of the parameter to be varied in the Monte Carlo analysis.

- **unif** Selects a uniform distribution with relative variation specification.

- **aunif** Selects a uniform distribution with absolute variation specification.

- **gauss** Selects a Gaussian distribution with relative variation specification.

- **agauss** Selects a Gaussian distribution with absolute variation specification.

- **limit** Selects a random limit distribution function using absolute variation. The result is either **nominal_val-abs_variation** or **nominal_val+abs_variation**, with 50% probability for each.

- **nominal_val** Nominal value for the parameter.

- **abs_variation** Largest deviation from **nominal_val** that can be obtained from a uniform or limit distribution, or the standard deviation multiplied by **sigma** for a Gaussian distribution.

- **rel_variation** Relative variation specification. The corresponding absolute variation is **rel_variation*nominal_val**.

- **sigma** Sigma-level at which the absolute or relative variation is specified for a Gaussian distribution. For example, if **sigma=3**, the standard deviation is **abs_variation/3**.

- **multiplier** Number of times the distribution function is evaluated. The largest deviation from the nominal value of all evaluations is the one that is used as the result. The resulting distribution is bimodal. *(Default: 1.)*
Multiple parameters can be assigned on the same `.param` command. Probability distributions are reevaluated with every use of `paramname` in expressions.

**Optimization Parameters**

When `.param` is used to define optimization parameters, the `parameter` argument uses the following syntax:

```
.param parameter=optname(guess, min, max [, delta])
```

- **parameter**: Global parameter name.
- **name**: References a particular optimization run name.
- **guess**: Initial (nominal) value for the parameter.
- **min**: Minimum values the parameter can take on.
- **max**: Maximum values the parameter can take on.
- **delta**: Used for discrete optimization. The final parameter values must differ from the initial guess by an integer multiple of `delta`. This is useful for optimizing quantities that can only take on discrete values, such as transistor lengths and widths.

The parameter `parameter` is allowed to vary within its range when an optimization of the appropriate `name` is invoked on an analysis command. During such an optimization, the parameter is initially assigned its `guess` value, but is allowed to vary within its range (defined by `min` and `max`) during subsequent optimization iterations.

For additional information on the optimization syntax for individual analysis commands, see “.ac” (page 76), “.dc” (page 89), or “.tran” (page 165).

**Examples**

```
.param pi='4*atan(1)' tf='1E-6*sin(pi/2))'
.tran 'tf*0.01' 'tf'
```

The `.param` command defines and assigns a value to parameter `tf`, which is subsequently used (enclosed by single quotes) in place of a numeric value in the “.tran” (page 165) command.

```
.param res=agauss (100, 10, 1)
```

Specifies that the resistance is chosen from a normal distribution of mean 100 and standard deviation 10.

```
.param w1=opt1 (10u, 2u, 20u, 0.25u)
```

Specifies that `w1` is to be varied in optimization run `opt1` within the limits $2 \times 10^{-6}$ and $20 \times 10^{-6}$. The initial guess for `w1` in the optimization is $10 \times 10^{-6}$, and the final value will be a multiple of $0.25 \times 10^{-6}$. 

---

Note: Multiple optimization parameters can be assigned on the same `.param` command.
.param safedivision(a,b)=’if(abs(b)<1e-100, 1e100, a/b)’
.print tran impedance=’safedivision(v(n1), i1(dev1))’

Creates a function named safedivision which divides one number by another without a division by zero error. This function is then used in a print expression.
.paramlimits

Sets optimization parameter ranges. Allows the same parameter to be varied in multiple optimizations with different optimization run names.

This command specifies that the parameter `paramname` (specified using `.param` elsewhere) is to be varied and optimized during an optimization run `optname`. Multiple instances of `.paramlimits` may not exist in the netlist for the same optimization run and the same parameter name, but are allowed for the same `optname` but for different `paramname` values.

**Note:** T-Spice supports sequential optimization—multiple optimizations may be performed in series from one input file, and the optimization results used in subsequent optimizations.

**Syntax**

```
.paramlimits optname paramname [guessval=guess] minval=min maxval=max
[delta=delta]
```

- `optname` Name of the optimization run.
- `paramname` References a particular parameter.
- `guess` Initial (nominal) parameter value for the optimization. If not used, the initial value defaults to the value specified in the `.param` command.
- `min` Minimum value of the range in which the parameter may vary.
- `max` Maximum value of the range in which the parameter may vary.
- `delta` If specified, the parameter can change only in integer multiples of this value. This is useful for optimizing quantities which can only take on discrete values, such as transistor lengths and widths.

**Examples**

```
.paramlimits opt1 r minval=10 maxval=100k
.paramlimits opt1 m1width minval=1u maxval=10u delta=0.25u
```
.power

Computes power dissipation in conjunction with transient analysis.

- If the `.tran` (page 165) command is missing from the input file, the .power command is ignored.
- The average power consumption, the instantaneous maximum power, and the instantaneous minimum power (in watts) and the times of maximum and minimum power consumption (in seconds) are reported at the end of the transient simulation.
- The instantaneous power \( P(t) \) dissipated by a voltage source at time \( t \) is the current through the source multiplied by the voltage drop across the source. The average power \( P \) for a time interval \((t1,t2)\) is computed by using the trapezoidal rule approximation to evaluate the integral

\[
P(t_1, t_2) = \frac{1}{t_2 - t_1} \int_{t_1}^{t_2} P(\tau) d\tau
\]

- Multiple .power commands can be used in a single simulation.
- Power results can also be reported with the `.print` (page 137) tran command.

Syntax

```
.power source [A [Z]]
```

- **source**: Voltage or resistor source whose power consumption is to be computed.
- **A**: Time at which power recording begins. (Unit: seconds. Default: simulation start time.)
- **Z**: Time at which power recording ends. (Unit: seconds. Default: simulation end time.)

Examples

```
.power vtest 3e-7
```

Computes the power dissipated by voltage source `vtest` between the given time (0.3 microsecond) and the end of the simulation. It might produce the following sample output:

```
Power Results
vtest from time 0 to 3e-007
Average power consumed -> 1.249948e-002 watts
Max power 2.490143e-002 at time 9.76e-006
Min power 2.282279e-030 at time 1e-005
```
.print

Stores simulation results to the solutions database.

- Expressions can be printed by themselves, with or without reference to physical quantities at specific nodes.
- Nodes and devices within subcircuits can be accessed with hierarchical notation in the form xinstance.xinstance.node.
- .print commands inside subcircuit definition blocks are replicated for each instance.
- If no arguments are given, all node voltages and source currents are printed. If neither mode or arguments are given, .print applies to all analysis types. If arguments are given, a mode must also be specified.
- Wildcards may be entered as part of the .print command node names, devices names, terminal names, or terminal numbers. Wildcards are expanded to match any available elements which match the name specification.
- Device State print statements are a means of obtaining very detailed information about devices and device internal states. Data such as the current, charge, capacitance, and voltage values can be listed, as well as certain model evaluation variable (threshold voltage, beta, etc.).
- See also “csv” (page 325) for output file options.

Note that the difference in automatically displaying traces in W-Edit is not due to using the .probe versus .print statement. What makes the difference is whether you explicitly list a specific plot item in either a .print or .probe statement.

For instance, the following command will not result in any automatically displayed traces, even though all node voltage traces would be generated (by default, dependent upon .option probev probei probeq etc.):

.probe tran

However, the two commands below will behave exactly the same way; they will have two automatically displayed traces.

.probe tran v(in) v(out)
.print tran v(in) v(out)

Note that printing from T-Spice has been limited to fifty traces. All curves are all listed in the Traces navigator of W-Edit, but not all traces will be displayed in the chart window. Since both .probe and .print statements are loaded to the W-Edit chart window, certain simulations are likely to exceed this maximum.

Syntax

.print [mode ] [arguments ]

mode  
Analysis mode (see below).

arguments  
Information to be printed (see below). arguments may include valid expressions involving other arguments or global parameters.
**mode** is one of the following:

- **tran** Print results from transient analysis.
- **dc** Print results from DC transfer analysis and DC operating point analysis.
- **ac** Print results from AC analysis.
- **noise** Print results from noise analysis.

**arguments** take one or more of a number of values, depending on **mode**. When no arguments are given, all node voltages and source currents are printed.

When an argument includes an expression, the expression must be enclosed by single quotes ('). A string can also be used as a column heading in the output file, and the string can be followed by an optional unit specifier, enclosed by angle brackets (<>). The unit is then displayed on the W-Edit y-axis.

Some entries in the argument tables below involve the variable **z**, to be replaced by a key letter or number representing a device terminal. The key letters and numbers corresponding to particular device terminals are as follows (alternatives are separated by slashes):

- **Diodes**: anode = P/1; cathode = N/2.
- **BJTs**: collector = C/1; base = B/2; emitter = E/3; substrate = S/4.
- **JFETs/MESFETs**: drain = D/1; gate = G/2; source = S/3.
- **MOSFETs**: drain = D/1; gate = G/2; source = S/3; bulk = B/4.

Wildcards provide an easy and compact method of printing a large number of node or device values which have related names. The T-Spice .print command supports several types of wildcards in the specification of the node name, device name, terminal number, and terminal name. The '*' character (asterisk) will be expanded to match any combination of alpha-numeric characters. The '?' character (question mark) will be expanded to match any single alpha-numeric character. And, '[...]' will be expanded to match any single character enclosed within the square brackets.

The **arguments** for transient, transfer, and DC analysis (**.print tran**, **.print dc**) are as follows.

- **n** Voltage at node **n** relative to ground.
- **i(d,n)** Current at node **n** of device **d** (inward current positive).
- **iz(d)** Current at terminal **z** of device **d** (inward current positive).
- **i#(x)** Current at terminal # of subcircuit **x** (inward current positive).
- **p(d)** Power consumed by voltage source **d**. This result can also be reported with the **.power** command.
- **q(d,n)** Charge at node **n** of device **d**.
- **qz(d)** Charge at terminal **z** of device **d**.
- **v(n1[[,n2]]**) Voltage at node **n1** relative to node **n2**. (Default reference node: ground.)
The arguments for AC analysis mode (.print ac) are as follows.

- idb(d,n): Current magnitude at node n of device d. (Unit: decibels.)
- idbz(d): Current magnitude at terminal z of device d. (Unit: decibels.)
- ii(d,n): Imaginary component of the complex current at node n of device d.
- iiiz(d): Imaginary component of the complex current at terminal z of device d.
- im(d,n): Current magnitude at node n of device d.
- imz(d): Current magnitude at terminal z of device d.
- ip(d,n): Current phase at node n of device d.
- ipz(d): Current phase at terminal z of device d.
- ir(d,n): Real component of the complex current at node n of device d.
- irz(d): Real component of the complex current at terminal z of device d.
- vdb(n1[[,]n2]): Voltage magnitude at node n1 relative to node n2 (Unit: decibels. Default reference node: ground.)
- vi(n1[[,]n2]): Imaginary component of the complex voltage at node n1 relative to node n2. (Default reference node: ground.)
- vm(n1[[,]n2]): Voltage magnitude at node n1 relative to node n2. (Default reference node: ground.)
- vp(n1[[,]n2]): Voltage phase at node n1 relative to node n2. (Default reference node: ground.)
- vr(n1[[,]n2]): Real component of the complex voltage at node n1 relative to node n2. (Default reference node: ground.)
- 'frequency()': AC frequency. Must be or be part of an expression enclosed by single quotes.

The arguments for noise analysis mode (.print noise) include any of the arguments for AC analysis mode in addition to the following:

- dn(d[,t]): Output noise spectral density contributions corresponding to the noise sources associated with device d. If the noise type t (see below) is not specified, then results for all applicable noise types are printed.
- inoise: Equivalent input noise spectral density magnitude. (Unit: volts/√Hertz.)
- inoise(db): Equivalent input noise spectral density magnitude. (Unit: decibels.)
- inoise(tot): Total input noise—the integral of the input noise spectral densities over the analysis frequency interval. (Unit: volts.)
- onoise: Output noise spectral density magnitude (Unit: volts/√Hertz.)
The units for the \texttt{.print noise dn(d, t)} command are volts⁄\text{Hertz} by default. However, this can be changed to Volts²⁄\text{Hertz} by use of the option \	exttt{dnout} (page 326).

The noise types \texttt{t} available for the \texttt{.print noise dn(d, t)} command vary according to the device type (BJT, Diode, JFET, etc.) as shown in the following tables:

### BJT (Gummel-Poon) Noise Types

<table>
<thead>
<tr>
<th>noise type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FN</td>
<td>Flicker noise due to base current</td>
</tr>
<tr>
<td>IB</td>
<td>Shot noise due to base current.</td>
</tr>
<tr>
<td>IC</td>
<td>Shot noise due to collector current.</td>
</tr>
<tr>
<td>RB</td>
<td>Thermal noise due to base resistance.</td>
</tr>
<tr>
<td>RC</td>
<td>Thermal noise due to collector resistance.</td>
</tr>
<tr>
<td>RE</td>
<td>Thermal noise due to emitter resistance.</td>
</tr>
<tr>
<td>RX</td>
<td>Transresistance from flicker noise source to output.</td>
</tr>
<tr>
<td>TOT</td>
<td>Total device output noise.</td>
</tr>
</tbody>
</table>

### BJT (VBIC) Noise Types

<table>
<thead>
<tr>
<th>noise type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IBE</td>
<td>Base-Emitter shot noise</td>
</tr>
<tr>
<td>IBEFN</td>
<td>Base-Emitter flicker noise</td>
</tr>
<tr>
<td>IBEP</td>
<td>Parasitic base-emitter shot noise</td>
</tr>
<tr>
<td>IBEPFN</td>
<td>Parasitic base-emitter flicker noise</td>
</tr>
<tr>
<td>ICCP</td>
<td>Parasitic base-collector shot noise</td>
</tr>
<tr>
<td>ITZF</td>
<td>Forward transport current shot noise</td>
</tr>
<tr>
<td>RBI</td>
<td>Thermal noise due to intrinsic base resistance</td>
</tr>
<tr>
<td>RBP</td>
<td>Thermal noise due to parasitic base resistance</td>
</tr>
<tr>
<td>RBX</td>
<td>Thermal noise due to extrinsic base resistance</td>
</tr>
<tr>
<td>RCI</td>
<td>Thermal noise due to intrinsic collector resistance</td>
</tr>
</tbody>
</table>
### Noise Types

<table>
<thead>
<tr>
<th>noise type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCX</td>
<td>Thermal noise due to extrinsic collector resistance</td>
</tr>
<tr>
<td>RE</td>
<td>Thermal noise due to emitter resistance</td>
</tr>
<tr>
<td>RS</td>
<td>Thermal noise due to source resistance</td>
</tr>
<tr>
<td>RX</td>
<td>Transresistance from flicker noise source to output.</td>
</tr>
<tr>
<td>TOT</td>
<td>Total device output noise.</td>
</tr>
</tbody>
</table>

#### Diode Noise Types

<table>
<thead>
<tr>
<th>noise type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FN</td>
<td>Flicker noise</td>
</tr>
<tr>
<td>ID</td>
<td>Shot noise.</td>
</tr>
<tr>
<td>RX</td>
<td>Transresistance from flicker noise source to output.</td>
</tr>
<tr>
<td>TOT</td>
<td>Total device output noise.</td>
</tr>
</tbody>
</table>

#### JFET and MESFET Noise Types

<table>
<thead>
<tr>
<th>noise type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FN</td>
<td>Flicker noise.</td>
</tr>
<tr>
<td>ID</td>
<td>Thermal noise due to channel.</td>
</tr>
<tr>
<td>RD</td>
<td>Thermal noise due to drain resistance.</td>
</tr>
<tr>
<td>RG</td>
<td>Thermal noise due to gate resistance.</td>
</tr>
<tr>
<td>RS</td>
<td>Thermal noise due to source resistance.</td>
</tr>
<tr>
<td>RX</td>
<td>Transresistance from flicker noise source to output.</td>
</tr>
<tr>
<td>TOT</td>
<td>Total device output noise.</td>
</tr>
</tbody>
</table>

#### MOSFET Noise Types

<table>
<thead>
<tr>
<th>noise type</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FN</td>
<td>Flicker noise.</td>
</tr>
<tr>
<td>ID</td>
<td>Thermal noise due to channel.</td>
</tr>
<tr>
<td>RD</td>
<td>Thermal noise due to drain resistance.</td>
</tr>
<tr>
<td>RG</td>
<td>Thermal noise due to gate resistance.</td>
</tr>
<tr>
<td>RS</td>
<td>Thermal noise due to source resistance.</td>
</tr>
<tr>
<td>RX</td>
<td>Transresistance from channel or flicker noise source to output.</td>
</tr>
<tr>
<td>TOT</td>
<td>Total device output noise.</td>
</tr>
</tbody>
</table>
Examples

```
.print tran in out i1(r2) id(M2)
```

Prints transient analysis results: the voltages at nodes in and out and the currents into terminal 1 of device r2 and the drain terminal of device M2.

```
.print dc I10/in, I10/out, I11/in, I11/out
```

Prints DC analysis results at various subcircuit nodes.

```
.print ac im(M2,g1) vm(out) vdb(out)
```

Writes AC analysis results: the magnitude of the current flowing into node g1 of device M2, the magnitude of the voltage at node out, and the same magnitude expressed in decibels.

```
.print noise inoise transfer dn(mn1) onoise(tot)
```

Prints noise analysis results: the equivalent input noise spectral density, the input/output transfer function, all noise information corresponding to device mn1, and the total output noise.

```
.print tran 'v(out)*sin(time()*sf)'
```

Prints the transient value of an expression involving the voltage at node out, the simulation time time(), and parameter sf (defined elsewhere with a .param command).

```
.print tran diff<V>='v(2)-v(1)'
```

Prints the transient value of an expression subtracting the voltage at node 1 from the voltage at node 2. The string diff is used as a column heading with the letter V as a unit designation.

```
.print
```

Prints all node voltages and voltage source currents for all analyses to a text file.

```
.print tran
```

Prints all node voltages and voltage source currents for transient analysis to a text file.

```
.print v(n*)
```

Prints the voltages for all nodes whose name begins with the letter 'n'.

```
.print i[12](m*) i?q*(q*)
```

Prints the drain and gate currents (terminals 1 and 2) for every MOSFET device, and each terminal current for every BJT.
Device State variables are not available for all types of analysis or for all devices. In general, the state plots are only relevant to DC and transient analysis.

The format of the device state plot request is always $\text{state}(d)$, where $\text{state}$ is the state data identifier, and $d$ is the device name.

The device state data which is available for each device type is as follows:

<table>
<thead>
<tr>
<th>$\text{state}$ identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{cap_be}$</td>
<td>$cbe$ capacitance</td>
</tr>
<tr>
<td>$\text{cap_ibc}$</td>
<td>internal base-collector capacitance</td>
</tr>
<tr>
<td>$\text{cap_sbc}$</td>
<td>$csc/csb$ substrate-collector/substrate-base capacitance</td>
</tr>
<tr>
<td>$\text{cap_xbc}$</td>
<td>external base-collector capacitance</td>
</tr>
<tr>
<td>$\text{cbo}$</td>
<td>base current</td>
</tr>
<tr>
<td>$\text{cco}$</td>
<td>collector current</td>
</tr>
<tr>
<td>$\text{cexbc}$</td>
<td>base-collector equivalent current</td>
</tr>
<tr>
<td>$\text{cqbc}$</td>
<td>current due to the base-collector charge</td>
</tr>
<tr>
<td>$\text{cqbe}$</td>
<td>current due to the base-emitter charge</td>
</tr>
<tr>
<td>$\text{cqbx}$</td>
<td>current due to the base-internal base charge</td>
</tr>
<tr>
<td>$\text{cqcs}$</td>
<td>current due to the collector-substrate charge</td>
</tr>
<tr>
<td>$\text{g0}$</td>
<td>$\partial i_c/\partial v_{ce}$</td>
</tr>
<tr>
<td>$\text{gm}$</td>
<td>$\partial i_c/\partial v_{be}$</td>
</tr>
<tr>
<td>$\text{gpi}$</td>
<td>$\partial i_b/\partial v_{be}$</td>
</tr>
<tr>
<td>$\text{gu}$</td>
<td>$\partial i_b/\partial v_{bc}$</td>
</tr>
<tr>
<td>$\text{isub}$</td>
<td>substrate current</td>
</tr>
<tr>
<td>$\text{qbc}$</td>
<td>base-collector charge</td>
</tr>
<tr>
<td>$\text{qbe}$</td>
<td>base-emitter charge</td>
</tr>
<tr>
<td>$\text{qbx}$</td>
<td>base-internal base charge</td>
</tr>
<tr>
<td>$\text{qcs}$</td>
<td>collector-substrate charge</td>
</tr>
<tr>
<td>$\text{rb}$</td>
<td>base resistance</td>
</tr>
<tr>
<td>$\text{rgn}$</td>
<td>operating region: -2=inverse, -1=saturation, 0=off, 1=on</td>
</tr>
<tr>
<td>$\text{vbc}$</td>
<td>base-collector voltage</td>
</tr>
<tr>
<td>$\text{vbe}$</td>
<td>base-emitter voltage</td>
</tr>
<tr>
<td>$\text{vbei}$</td>
<td>$rb$ and $rc$ offset internal base-collector voltage</td>
</tr>
<tr>
<td>$\text{vbe}$</td>
<td>base-emitter voltage</td>
</tr>
<tr>
<td>$\text{vbei}$</td>
<td>$rb$ and $re$ offset internal base-emitter voltage</td>
</tr>
<tr>
<td>$\text{vsub}$</td>
<td>substrate voltage</td>
</tr>
<tr>
<td>state identifier</td>
<td>description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>cbco</td>
<td>parasitic Base-Collector overlap capacitance (fixed)</td>
</tr>
<tr>
<td>cbeo</td>
<td>parasitic Base-Emitter overlap capacitance (fixed)</td>
</tr>
<tr>
<td>cqbc</td>
<td>Base-Collector charge current</td>
</tr>
<tr>
<td>cqbcx</td>
<td>currents from Cbco charge</td>
</tr>
<tr>
<td>cqbec</td>
<td>currents from Cbcp charge</td>
</tr>
<tr>
<td>cqbe</td>
<td>Base-Emitter charge current</td>
</tr>
<tr>
<td>cqbeo</td>
<td>currents from Cbeo charge</td>
</tr>
<tr>
<td>cqbeo</td>
<td>currents from Cbeo charge</td>
</tr>
<tr>
<td>cqbeo</td>
<td>currents from Cbeo charge</td>
</tr>
<tr>
<td>cqbx</td>
<td>currents from Cbex charge</td>
</tr>
<tr>
<td>cqbcx</td>
<td>currents from Cbcx charge</td>
</tr>
<tr>
<td>cqbcx</td>
<td>currents from Cbex charge</td>
</tr>
<tr>
<td>cqbe</td>
<td>Base-Emitter charge current</td>
</tr>
<tr>
<td>cqbeo</td>
<td>currents from Cbeo charge</td>
</tr>
<tr>
<td>cqbeo</td>
<td>currents from Cbeo charge</td>
</tr>
<tr>
<td>cqbeo</td>
<td>currents from Cbeo charge</td>
</tr>
<tr>
<td>flxf</td>
<td>Excess phase circuit flux</td>
</tr>
<tr>
<td>ibc</td>
<td>intrinsic Base-Collector current</td>
</tr>
<tr>
<td>ibcp</td>
<td>parasitic Base-Collector current</td>
</tr>
<tr>
<td>ibe</td>
<td>intrinsic Base-Emitter current</td>
</tr>
<tr>
<td>ibep</td>
<td>parasitic Base-Emitter current</td>
</tr>
<tr>
<td>ibex</td>
<td>extrinsic Base-Emitter current</td>
</tr>
<tr>
<td>igc</td>
<td>weak avalanche current</td>
</tr>
<tr>
<td>irbi</td>
<td>intrinsic Base resistor modulated current</td>
</tr>
<tr>
<td>irbp</td>
<td>parasitic Base resistor modulated current</td>
</tr>
<tr>
<td>irbx</td>
<td>external Base resistor current</td>
</tr>
<tr>
<td>irci</td>
<td>intrinsic Collector resistor modulated current</td>
</tr>
<tr>
<td>ircx</td>
<td>external Collector resistor current</td>
</tr>
<tr>
<td>ire</td>
<td>external Emitter resistor current</td>
</tr>
<tr>
<td>ith</td>
<td>thermal (heat generation) source, power dissipation</td>
</tr>
<tr>
<td>itxf</td>
<td>forward transport current, with excess phase</td>
</tr>
<tr>
<td>itzf</td>
<td>forward transport current, zero phase</td>
</tr>
<tr>
<td>itzx</td>
<td>reverse transport current, zero phase</td>
</tr>
<tr>
<td>itzx</td>
<td>reverse transport current, zero phase</td>
</tr>
<tr>
<td>ixxf</td>
<td>forward transport current, with excess phase</td>
</tr>
<tr>
<td>ixzf</td>
<td>forward transport current, with excess phase</td>
</tr>
<tr>
<td>qbc</td>
<td>Base-Collector charge</td>
</tr>
<tr>
<td>qbc</td>
<td>Base-Collector charge (depletion)</td>
</tr>
<tr>
<td>state identifier</td>
<td>description</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>qbcx</td>
<td>parasitic Base-Collector charge (depletion)</td>
</tr>
<tr>
<td>qbeo</td>
<td>parasitic Base-Collector charge (depletion and diffusion)</td>
</tr>
<tr>
<td>qbe</td>
<td>Base-Emitter charge</td>
</tr>
<tr>
<td>qbeo</td>
<td>parasitic Base-Emitter charge (depletion and diffusion)</td>
</tr>
<tr>
<td>qbep</td>
<td>parasitic Base-Emitter charge (depletion and diffusion)</td>
</tr>
<tr>
<td>qbex</td>
<td>extrinsic Base-Emitter charge (depletion)</td>
</tr>
<tr>
<td>qcx</td>
<td>Excess phase circuit capacitance</td>
</tr>
<tr>
<td>rbi</td>
<td>intrinsic Base resistance (modulated)</td>
</tr>
<tr>
<td>rbip</td>
<td>parasitic Base resistance (modulated)</td>
</tr>
<tr>
<td>rbx</td>
<td>extrinsic Base resistance (fixed)</td>
</tr>
<tr>
<td>rci</td>
<td>intrinsic Collector resistance (modulated)</td>
</tr>
<tr>
<td>rcx</td>
<td>extrinsic Collector resistance (fixed)</td>
</tr>
<tr>
<td>re</td>
<td>Emitter resistance (fixed)</td>
</tr>
<tr>
<td>rgn</td>
<td>operating region</td>
</tr>
<tr>
<td>rs</td>
<td>Substrate resistance (fixed)</td>
</tr>
<tr>
<td>vb</td>
<td>Base voltage</td>
</tr>
<tr>
<td>vbc</td>
<td>Base-Collector voltage</td>
</tr>
<tr>
<td>vbcx</td>
<td>Rb and Rc offset internal Base-Collector voltage</td>
</tr>
<tr>
<td>vbe</td>
<td>Base-Emitter voltage</td>
</tr>
<tr>
<td>vbei</td>
<td>Rb and Re offset internal Base-Emitter voltage</td>
</tr>
<tr>
<td>vbi</td>
<td>Bi internal Base voltage</td>
</tr>
<tr>
<td>vbp</td>
<td>Bp parasitic Base voltage</td>
</tr>
<tr>
<td>vbx</td>
<td>Bx external Base voltage</td>
</tr>
<tr>
<td>ve</td>
<td>Collector voltage</td>
</tr>
<tr>
<td>vci</td>
<td>Ci internal Collector voltage</td>
</tr>
<tr>
<td>vcx</td>
<td>Cx external Collector voltage</td>
</tr>
<tr>
<td>ve</td>
<td>Emitter voltage</td>
</tr>
<tr>
<td>vei</td>
<td>Ei internal Emitter voltage</td>
</tr>
<tr>
<td>vs</td>
<td>Substrate voltage</td>
</tr>
<tr>
<td>vsi</td>
<td>Si internal Substrate voltage</td>
</tr>
</tbody>
</table>
### Capacitor Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>creff</td>
<td>effective capacitance</td>
</tr>
<tr>
<td>curr</td>
<td>current</td>
</tr>
<tr>
<td>dq</td>
<td>$\dot{q}/\dot{v}$</td>
</tr>
<tr>
<td>q</td>
<td>charge</td>
</tr>
<tr>
<td>volt</td>
<td>voltage potential</td>
</tr>
</tbody>
</table>

### f element (CCCS) Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curr</td>
<td>source current</td>
</tr>
<tr>
<td>di or di1</td>
<td>derivative of source current w.r.t. first control</td>
</tr>
<tr>
<td>di2</td>
<td>derivative of source current w.r.t. second control</td>
</tr>
<tr>
<td>di3</td>
<td>derivative of source current w.r.t. third control</td>
</tr>
<tr>
<td>volt</td>
<td>voltage potential across the CCCS</td>
</tr>
</tbody>
</table>

### h element (CCVS) Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curr</td>
<td>source current</td>
</tr>
<tr>
<td>dv or dv1</td>
<td>derivative of source voltage w.r.t. first control</td>
</tr>
<tr>
<td>dv2</td>
<td>derivative of source voltage w.r.t. second control</td>
</tr>
<tr>
<td>dv3</td>
<td>derivative of source voltage w.r.t. third control</td>
</tr>
<tr>
<td>volt</td>
<td>voltage potential across the CCVS</td>
</tr>
</tbody>
</table>

### Diode Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>c</td>
<td>total diode capacitance</td>
</tr>
<tr>
<td>curr</td>
<td>current through the diode</td>
</tr>
<tr>
<td>di</td>
<td>$\dot{i}/\dot{v}$</td>
</tr>
<tr>
<td>dq</td>
<td>$\dot{q}/\dot{v}$</td>
</tr>
<tr>
<td>gd</td>
<td>conductance</td>
</tr>
<tr>
<td>id</td>
<td>current, excluding the series resistor</td>
</tr>
</tbody>
</table>
### Diode Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ir</td>
<td>current through the series resistor</td>
</tr>
<tr>
<td>qd</td>
<td>charge</td>
</tr>
<tr>
<td>rgn</td>
<td>operating region: -1=breakdown, 0=reverse, 1=forward</td>
</tr>
<tr>
<td>vd</td>
<td>voltage potential, excluding the series resistor</td>
</tr>
<tr>
<td>volt</td>
<td>voltage across the diode</td>
</tr>
<tr>
<td>vr</td>
<td>voltage across the series resistor</td>
</tr>
</tbody>
</table>

### Inductor Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curr</td>
<td>current through the inductor</td>
</tr>
<tr>
<td>ic</td>
<td>current through the component capacitor</td>
</tr>
<tr>
<td>ir</td>
<td>current through the component resistor</td>
</tr>
<tr>
<td>leff</td>
<td>effective inductance</td>
</tr>
<tr>
<td>vc</td>
<td>voltage across the component capacitor</td>
</tr>
<tr>
<td>volt</td>
<td>voltage across the inductor</td>
</tr>
<tr>
<td>vr</td>
<td>voltage across the component resistor</td>
</tr>
</tbody>
</table>

### MOSFET Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>betaeff</td>
<td>effective beta</td>
</tr>
<tr>
<td>cap_bd</td>
<td>bulk-drain capacitance</td>
</tr>
<tr>
<td>cap_bs</td>
<td>bulk-source capacitance</td>
</tr>
<tr>
<td>cbdbo</td>
<td>(\partial Q_b/\partial V_d)</td>
</tr>
<tr>
<td>cbdo</td>
<td>DC drain-bulk diode current</td>
</tr>
<tr>
<td>chgbo</td>
<td>(\partial Q_b/\partial V_g)</td>
</tr>
<tr>
<td>cbsbo</td>
<td>(\partial Q_b/\partial V_s)</td>
</tr>
<tr>
<td>cbso</td>
<td>DC source-bulk diode current</td>
</tr>
<tr>
<td>cddbo</td>
<td>(\partial Q_d/\partial V_d)</td>
</tr>
<tr>
<td>cdgbo</td>
<td>(\partial Q_d/\partial V_g)</td>
</tr>
<tr>
<td>cdo</td>
<td>DC drain current</td>
</tr>
<tr>
<td>cdsbo</td>
<td>(\partial Q_d/\partial V_s)</td>
</tr>
<tr>
<td>cgdbo</td>
<td>(\partial Q_g/\partial V_d)</td>
</tr>
<tr>
<td>state identifier</td>
<td>MOSFET Device State printout identifiers</td>
</tr>
<tr>
<td>------------------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>cggbo</td>
<td>$\partial Q_g/\partial V_g$</td>
</tr>
<tr>
<td>cgsbo</td>
<td>$\partial Q_g/\partial V_s$</td>
</tr>
<tr>
<td>cqb</td>
<td>current due to the intrinsic bulk charge</td>
</tr>
<tr>
<td>cqd</td>
<td>current due to the intrinsic drain charge</td>
</tr>
<tr>
<td>cqg</td>
<td>current due to the intrinsic gate charge</td>
</tr>
<tr>
<td>cqs</td>
<td>current due to the intrinsic source charge</td>
</tr>
<tr>
<td>deltal</td>
<td>channel length modulation</td>
</tr>
<tr>
<td>gammaeff</td>
<td>effective gamma</td>
</tr>
<tr>
<td>gbdo</td>
<td>Conductance of the drain diode</td>
</tr>
<tr>
<td>gbso</td>
<td>conductance of the source diode</td>
</tr>
<tr>
<td>gdso</td>
<td>DC drain-source transconductance</td>
</tr>
<tr>
<td>gmbso</td>
<td>DC substrate transconductance</td>
</tr>
<tr>
<td>gmo</td>
<td>DC gate transconductance</td>
</tr>
<tr>
<td>qbi</td>
<td>intrinsic bulk charge</td>
</tr>
<tr>
<td>qbd</td>
<td>bulk-drain diode charge</td>
</tr>
<tr>
<td>qbs</td>
<td>bulk-source diode charge</td>
</tr>
<tr>
<td>cqbd</td>
<td>current due to bulk-drain diode charge</td>
</tr>
<tr>
<td>cqbs</td>
<td>current due to bulk-source diode charge</td>
</tr>
<tr>
<td>qdi</td>
<td>intrinsic drain charge</td>
</tr>
<tr>
<td>qgi</td>
<td>intrinsic gate charge</td>
</tr>
<tr>
<td>qsi</td>
<td>intrinsic source charge</td>
</tr>
<tr>
<td>rggn</td>
<td>operating region: -1=subthreshold, 0=linear, 1=saturation</td>
</tr>
<tr>
<td>ueff</td>
<td>effective mobility</td>
</tr>
<tr>
<td>vbs</td>
<td>bulk-source voltage</td>
</tr>
<tr>
<td>vbsi</td>
<td>Rs offset internal bulk-source voltage</td>
</tr>
<tr>
<td>vds</td>
<td>drain-source voltage</td>
</tr>
<tr>
<td>vdsat</td>
<td>saturation voltage</td>
</tr>
<tr>
<td>vdsi</td>
<td>Rd and Rs offset internal drain-source voltage</td>
</tr>
<tr>
<td>vfbeff</td>
<td>effective Vfb</td>
</tr>
<tr>
<td>vgs</td>
<td>gate-source voltage</td>
</tr>
<tr>
<td>vgsi</td>
<td>Rs offset internal gate-source voltage</td>
</tr>
<tr>
<td>vth</td>
<td>Threshold voltage</td>
</tr>
</tbody>
</table>
### Resistor Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cap1</td>
<td>capacitance of the first capacitor</td>
</tr>
<tr>
<td>cap2</td>
<td>capacitance of the second capacitor</td>
</tr>
<tr>
<td>curr</td>
<td>current through the resistor</td>
</tr>
<tr>
<td>di</td>
<td>( \frac{\partial I}{\partial V} )</td>
</tr>
<tr>
<td>g</td>
<td>conductance</td>
</tr>
<tr>
<td>ic1</td>
<td>current through the first capacitor</td>
</tr>
<tr>
<td>ic2</td>
<td>current through the second capacitor</td>
</tr>
<tr>
<td>qc1</td>
<td>charge of the first capacitor</td>
</tr>
<tr>
<td>qc2</td>
<td>charge of the second capacitor</td>
</tr>
<tr>
<td>r</td>
<td>effective resistance</td>
</tr>
<tr>
<td>vc1</td>
<td>voltage across the first capacitor</td>
</tr>
<tr>
<td>vc2</td>
<td>voltage across the second capacitor</td>
</tr>
<tr>
<td>volt</td>
<td>voltage across the resistor</td>
</tr>
</tbody>
</table>

### g element (VCCS) Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curr</td>
<td>VCCS source current</td>
</tr>
<tr>
<td>di or di1</td>
<td>derivative of source current w.r.t.</td>
</tr>
<tr>
<td>di2</td>
<td>the first control</td>
</tr>
<tr>
<td>di3</td>
<td>the second control</td>
</tr>
<tr>
<td>dq or dq1</td>
<td>derivative of source charge w.r.t.</td>
</tr>
<tr>
<td>dq2</td>
<td>the first control</td>
</tr>
<tr>
<td>dq3</td>
<td>the second control</td>
</tr>
<tr>
<td>q</td>
<td>VCCS source charge</td>
</tr>
<tr>
<td>volt</td>
<td>voltage across the VCCS</td>
</tr>
</tbody>
</table>

### e element (VCVS) Device State printout identifiers

<table>
<thead>
<tr>
<th>state identifier</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>curr</td>
<td>VCVS source current</td>
</tr>
<tr>
<td>dv or dv1</td>
<td>derivative of source voltage w.r.t.</td>
</tr>
<tr>
<td>dv2</td>
<td>the first control</td>
</tr>
<tr>
<td></td>
<td>the second control</td>
</tr>
</tbody>
</table>
You can export a single MOSFET using .print. For example, the following exports a MOSFET mn1 vth vs. time:

```
.print tran mn1:vth
```

The same format may be used for these BSIM# v3.33 output parameters:

- beta - Beta
- capbd - Bulk-Drain capacitance
- capbs - Bulk-Source capacitance
- cbd - DC Bulk-Drain diode current
- cbs - DC Bulk-Source diode current
- cbtot - Cbb
- cdtot - Cdd
- ceqbd - Ibd
- ceqbs - Ibs
- cgbo - Gate-Bulk overlap capacitance
- cgd - Cgd
- cgdb - Cgdb
- cgdo - Gate-Drain overlap capacitance
- cgs - Cgs
- cgsh - Cgsh
- cgso - Gate-Source overlap capacitance
- cgtot - Cgg
- cqb - Current from Bulk charge
- cqd - Current from Drain charge
- cqg - Current from Gate charge
- cqs - Current from Source charge
- cstot - Css
- gbd - Gbd
- gbs - Gbs
- gds - Gds
- gm - Gm
- gmbs - Gmb
- ib - bulk current
- id - drain current
- ids - Ids
- ig - gate current
- is - source current
- power - Dissipated power
- qb - Bulk charge
- qbd - Bulk-Drain charge
- qbi - Intrinsic Bulk charge
- qbs - Bulk-Source charge
- qd - Drain charge
- qdi - Intrinsic Drain charge
- qg - Gate charge
- qgi - Intrinsic Gate charge
- qs - Source charge
- rd - Drain resistor
- rs - Source resistor
.probe

Stores simulation results to the solutions database. .probe acts exactly as .print, except that outputs are not automatically displayed in the active W-Edit chart. If arguments are given, the mode must be specified.

Note that the difference in automatically displaying traces in W-Edit is not due to using the .probe versus .print statement. What makes the difference is whether you explicitly list a specific plot item in either a .print or .probe statement.

.probe tran

For instance, the above command will not result in any automatically displayed traces, even though all node voltage traces would be generated (by default, dependent upon .option probev probei probeq etc.):

.probe tran v(in) v(out)
.proprint tran v(in) v(out)

However, the two above commands will behave exactly the same way; they will have two automatically displayed traces.

- See also “csv” (page 325) for output file options.

Syntax

.probe [mode] [arguments]

mode Analysis type (see below). If mode is omitted, the .probe command applies to all analysis types.

arguments Specifies plot variables to be included in the output file. If arguments is omitted, T-Spice includes all node voltages and voltage source currents in the output. The format and types of arguments are the same as for the “.print” (page 137) command.

mode is one of the following:

tran Print results from transient analysis.
dc Print results from DC transfer analysis and DC operating point analysis.
ac Print results from AC analysis.
noise Print results from noise analysis.

Examples

.probe

Saves all node voltages and voltage source currents for all analyses to a binary file.

.probe tran
Saves all node voltages and voltage source currents for transient analysis to a binary file.

```
.probe tran v(2)
```

Saves the transient voltage at node 2 to a binary file.

```
.probe tran Output<v> = 'v(2) - v(1)'
```

Saves the transient value of an expression subtracting voltage at node 1 from voltage at node 2 and prints it in a column with the heading Output.
.protect / .unprotect

The .protect and .unprotect commands are used for temporarily turning input file echoing off and on. Input file echoing is initially turned on with the command .option echo. Subsequent use of .protect will turn off the echoing, until the .unprotect command is encountered. In this manner sensitive data, such as model libraries, can be protected from distribution, or excessive amounts of netlist echoing can be trimmed down.

Syntax

.protect

...

.unprotect
.save

Saves bias point information to a file. All of the non-internal node voltage values will be saved to the file using either the .ic or the .nodeset T-Spice command syntax. Subsequent simulations may use the .load command to read the file and execute the .nodeset or the .ic commands.

Syntax

```
.save [file=filename] [type=ic | nodeset] [time=time]
```

- **filename**
  Name of the file to be written. If the `file` parameter is not entered, then the filename is derived from the simulation output filename, with a .ic file extension.

- **type**
  Denotes the type of node initialization command which will be written to the file for each non-internal circuit node. Either `ic`, to have .ic commands generated, or `nodeset`, for .nodeset commands. *(Default: nodeset)*

- **time**
  The transient analysis time at which the bias information should be saved. Ordinarily, the .save command saves the DC operating point bias information. If a `time` parameter has been specified, and the simulation performs a transient analysis, then the bias at the specified timepoint will be saved.

Examples

```
.save type=ic time=50n
```
.savebias

Saves bias point information to a file. Node voltage values will be saved to the file using either the .ic or the .nodeset T-Spice command syntax. Subsequent simulations may use the .load command to read the file and execute the .nodeset or the .ic commands.

Note:
The .save and the .savebias commands perform essentially the same task, but use a different syntax and have different options. .save is provided for HSPICE command compatibility, while .savebias provides PSPICE command compatibility. The saved file from either command can be loaded into a subsequent simulation using the .load command.

Syntax

.savebias filename [op | dc | tran] [alter = alternum] [dc = dcvalue] [step = stepvalue] [temp = temperature] [time = timevalue] [repeat] [ic | nodeset] [internal] [nosubckt]

filename Name of the file to be written.

op | dc | tran Indicates the type of analysis for which voltage values will be saved. (Default: op)

alternum Identifies the alter block index number for which data should be saved. (Default: 0)

dcvalue Identifies the DC sweep value for which data should be saved. (Default: all)

stepvalue Identifies the .step value for which data should be saved. (Default: all)

temperature Identifies the .temp temperature for which data should be saved. (Default: all)

timevalue Identifies the transient timepoint at which data should be saved. (Default: 0)

repeat For transient timepoint saving, indicates that the output file should repeatedly be overwritten for each timepoint which is an integral multiple of timevalue.

ic Indicates that the node initialization command which is written should be the .ic command.

nodeset Indicates that the node initialization command which is written should be the .nodeset command.

internal Indicates that all internal node values should be included in the output. Internal nodes are those nodes which were not in the input circuit, but were automatically generated internal to devices.

nosubckt Indicates that only the top level circuit nodes, excluding subcircuit nodes, should be written.
Examples

.savebias ring.ic tran ic time=100n repeat

.savebias dc alter=2 dc=2.5 temp=75 internal
.step

Performs a parametric sweep of a sweep variable, performing all analyses in the input file for all parameter values in the sweep.

The .step command produces a separate output section for each parameter value of the sweep. For example, an input file with .step and `.tran` (page 165) produces one transient analysis output section for each parameter value in the sweep. In addition, all `.macro .eom` (page 107) results are plotted as traces with the swept variable as the x-axis. The output format for this is similar to that of the `.dc` (page 89) command.

**Note:**
The .step command can be abbreviated to .st.

**Syntax**

```
.step sweep [[[sweep] sweep [[[sweep] sweep]]]
```

where **sweep** is in one of the following formats:

- `[lin] npoints start stop`
  or
- `dec|oct variable start stop npoints`
  or
- `variable lin|dec|oct npoints start stop`
  or
- `variable list value [value [...]]`
  or
- `list variable value [value [...]]`
  or
- `variable poi npoints [value [...]]`
  or
- `data=dataname`
  or
- `monte=mcruns [seed=seedval]`
  or
- `optimize=optname results=measname [measname [...]] model=optmodelname`
```
**variable** specifies the parameter whose value is to be swept. It is one of the following:

- **temp**
  
  Specifies a temperature sweep.

  *Note*: Use `.dc` to plot voltage/current vs. temperature, not `.temp` and `.step temp`.

- **param paramname**
  
  Sweeps a global parameter named `paramname` defined using the `.param` command.

- **source sourcename**
  
  Sweeps the DC value of a voltage or current source value named `sourcename`.

- `[modparam] parname(modelname)`
  
  Sweeps the value of model parameter `parname` for the device model `modelname`.

- **paramname**
  
  Sweeps a global parameter (as with `param`) or DC source value (as with `source`). T-Spice first looks for a matching `.param` parameter, and then for a source name.

Other parameters include the following:

- **start**
  
  Specifies the beginning of a linear or logarithmic sweep.

- **stop**
  
  Specifies the end of a linear or logarithmic sweep.

- **inc**
  
  Specifies the increment for a linear sweep.

- **npoints**
  
  Specifies the total number of points for a linear or `poi` sweep, or the number of points per decade or octave for a logarithmic sweep.

- **value**
  
  Specifies a single value which `variable` takes on for one step of the sweep.

- **mcruns**
  
  Specifies the number of runs to be performed for Monte Carlo analysis.

- **seedval**
  
  An integer specifying a seed for initializing the random number sequence for Monte Carlo analysis. If `seedval` is negative, T-Spice uses the system clock to generate a different seed each time the simulation is run. A `seedval` of zero is equivalent to not specifying a seed value at all.

- **dataname**
  
  Specifies the name of a “.connect” (page 86) statement to be used for the sweep. The column names in the `.data` statement must correspond to global “.param” (page 131) parameters. For each sweep step, those parameters are assigned the values found in one row of data produced by the `.data` statement.

- **sweep**
  
  Specifies that analysis be performed for all parameter values of the sweep and indicates the beginning of the next nested sweep variable specification. The `sweep` keyword can be omitted if the previous sweep is not of the `list` or `poi` type or if one of the keywords `lin`, `dec`, `oct`, `list`, `poi`, `temp`, `param`, `source`, or `modparam` follows immediately.

- **lin**
  
  Specifies a linear sweep.

- **dec**
  
  Specifies a logarithmic sweep by decades.
oct

Specifies a logarithmic sweep by octaves.

list

Specifies a sweep over a list of values (P-Spice compatible syntax).

poi

Specifies a sweep over a list of values (HSPICE compatible syntax).

data

Specifies a sweep defined using a .data statement.

monte

Specifies a Monte Carlo sweep. For each Monte Carlo run, random circuit parameter values are generated from probability distributions. A Monte Carlo sweep must be the outermost sweep if sweeps are nested.

optimize

Specifies an optimization sweep. During an optimization sweep, T-Spice runs many analyses in an attempt to optimize a circuit performance objective. The user may specify a set of parameters to be varied and a set of measurements to be included in the optimization goal. Optimization sweeps may not be nested within other optimization sweeps. For further information on setting up an optimization run, see “Optimization” on page 525.

opname

Selects a set of parameters to be varied in an optimization run. The parameters to be optimized are specified using “.param” (page 131) with a matching opname.

results=measname

Specifies circuit measurement results to be used for defining an optimization goal. Each measname refers to a “.macro /eom” (page 107) command of the same name and contributes to the optimization goal. The complete optimization goal is the RMS of all measurements listed. For further information on specifying circuit measurement results, see “Defining Optimization Goals” on page 526.

model=optmodelname

Specifies an optimization algorithm model name. It is matched with a “.model” (page 116) statement of type opt and name modelname. That .model statement specifies parameters for the optimization algorithm. For further information on specifying an optimization algorithm, see “Optimization” on page 525.

Examples

.step vin 0 5 0.1

sweeps the DC value of voltage source vin from 0 to 5V with 0.1V increments.

.step lin param ml 2 3 0.5 sweep vdd 3 5 0.1

performs a nested linear sweep of the parameter ml and the voltage source vdd.

.step list temp 0 27 100 150 -50

sweeps the circuit operating temperature over the five values listed.

.step optimize=opt1 results=bandwidth,delay model=optmod
.param pl=opt1(le=3,le=5) p2=opt1(150,100,200)
.model optmod opt level=1 itropt=40
.measure ac bandwidth trig vm(out) val=0.5 cross=1
+ tarrg vm(out) val=0.5 cross=2
+ goal=2kHz
.measure tran delay when v(1)=2.5 goal=10ns

invokes an optimization of parameters \( p_1 \) and \( p_2 \). T-Spice will attempt to find values for \( p_1 \) and \( p_2 \) which result in a bandwidth of 2 kHz and a delay of 10 ns. An AC and a transient analysis would be performed for each optimization function evaluation.
**.subckt**

Defines a hierarchical set of devices and nodes to be used repeatedly in a higher-level circuit.

- Subcircuits are replicated by means of the instance (\(x\)) statement.
- When invocations of the following commands appear within subcircuit definitions and refer to nodes inside the subcircuit, the commands are executed for each instance of the node: `.acmodel` (page 79), `.hdl` (page 98), `.macro .eom` (page 107), `.nodeset` (page 119), `.noise` (page 120), `.print` (page 137), and `.probe` (page 152), and `.tf` (page 164).
- Node and device names have local scope in subcircuits unless global node names (defined elsewhere with the `.global` (page 97) command) are used.
- Subcircuit blocks cannot be nested: after one `.subckt` command, the `.ends` (page 94) command must appear before another `.subckt` command can be used.

**Syntax**

```
.subckt name node1 [node2 ...] [parameter=X ...]
subcircuit
.ends
```

- `name` Name of subcircuit.
- `node1 node2` Nodes used as “external” connections to the subcircuit.
- `parameter` Parameter(s), with default value(s) assigned. \(X\) can be a number or an expression. Subcircuit parameters have local scope. Parameters can be written in any order in both definition and instances. Parameter values specified in the definition are used as defaults when not specified in instances. Within the definition, parameter values are referenced (in place of numbers) by enclosing their names in single quotes. Alternatively, the `.param` (page 131) command may be used within the definition, with the same results. Parameters created outside the definition with the `.param` (page 131) command may be used inside the definition, but an assignment made with the `.subckt` command to an externally defined parameter always overrides its external value.
- `subcircuit` Subcircuit definition (may be multiple lines).

**Examples**

```
.subckt inv in out Vdd length=1.25u nwidth=2u pwidth=3u
mt1 out in GND GND nmos l='length' w='nwidth'
mt2 out in Vdd Vdd pmos l='length' w='pwidth'
c2 out GND 800f
.ends inv
```

This subcircuit could be instantiated as follows:

```
ixinv1 a1 a2 Vdd inv nwidth=4u pwidth=6u
```
.temp

Specifies the temperatures at which the circuit is to be simulated.

- Changing the temperature affects the behavior of diode, resistor, BJT, JFET, MESFET, and MOSFET models. It may also affect the behavior of Verilog-A devices.
- The .temp command has no effect on external tables, which should be regenerated to reflect the new temperature.

Syntax

```
.temp temperature [temperature [temperature [temperature [...]]]]
```

`temperature` Temperature. (Unit: °C. Default: 25.)

Using .TEMP and .STEP displays voltage vs. voltage plots with different temperatures displaying as different traces. For example,

```
.DC vin 0 5 0.1
.TEMP 25 30 35 40 50
```

To plot voltage vs. temperature, use the following so that the first sweep variable in the DC analysis will become the X axis:

```
.DC temp 25 40 5 VIN 0 5 0.1
```
Computes and reports the value of the small-signal DC transfer function between the specified output and input, and the corresponding input and output resistances, at the DC operating point.

- The `.tf` command automatically performs (but does not report the results from) a DC operating point calculation.
- Results are reported under the heading SMALL-SIGNAL TRANSFER FUNCTION (to the specified output file or in the Simulation Window).
- The transfer function value corresponds to a voltage (V/V) or current (I/A) gain, a transconductance (I/V), or a transresistance (V/I).

**Syntax**

```
.tf arguments source
```

- **arguments** Any arguments appropriate for the “`.print`” (page 137) `dc` command.
- **source** Voltage or current input source.

**Examples**

```
.tf i(mb1,out1) ii1
```

Computes transfer function results between node `out1` of device `mb1` and current source `ii1`. 
.tran

Performs large-signal time-domain (transient) analysis of the circuit to determine its response to initial conditions and time-dependent stimuli.

- The time step is adaptively varied throughout the simulation to ensure accuracy.
- Results for nodes selected by the .print tran, .probe tran, and .measure tran commands will be output for every time step, unless otherwise specified by the .options prtdel command. For additional information on these commands, see "".print" (page 137), "".probe" (page 152), "".macro .eom" (page 107) and "".options" (page 125).

Syntax

```
.tran[/mode] S L [start=A] [UIC] [restart] [restarttime=restarttime] [restartfile=restartfile] [sweep sweep]
```

**mode**
Analysis mode (see below). This parameter must immediately follow the keyword .tran and be preceded by a slash (/).

**S**
Maximum time step allowed. By default, the time step is dynamically adapted to resolve the output values. (Unit: seconds.)

**L**
Total simulation time. (Unit: seconds.)

**A**
Output start time. Execution of the .print tran command will not start until this time. (Unit: seconds. Default: 0.)

**UIC**
(An abbreviation for "Use Initial Conditions.") Instructs T-Spice to skip the DC operating point analysis for determining the time=0 circuit state. Instead, only the initial conditions specified using .ic commands are used to set the time=0 voltages. Voltages which cannot be determined using .ic commands are set to zero.

You can also use the .load command to load initial conditions.

**Note:** Although the dialog does not prevent it, it does not make sense to combine the UIC setting with a transient /powerup option, because UIC instructs T-Spice to fix the .IC values at time 0 to enforce a voltage setting for the initial transient analysis timeframe and /powerup instructs T-Spice to ramp all voltage sources from 0 volts to their initial settings in a brief timeframe.

**restart**
Denotes that this is a checkpoint-restart restarted simulation.

**restarttime**
The transient timepoint at which a checkpoint-restart simulation is to be restarted. If the checkpointed data file does not contain this exact timepoint, then the closest available time will be used.

**restartfile**
The name of the checkpoint datafile that is to be used for a simulation restart.

**sweep**
For a description of the syntax for this field, see "".step" (page 158).
Chapter 5: Simulation Commands

`.tran` takes one of the following values:

- **mode**
  - Performs a DC operating point calculation before simulation to determine initial steady-state node voltages. The commands `".nodeset"` (page 119) or `".hdl"` (page 98) can be used to impose initial conditions.
  - Performs a “powerup” simulation. All nodes are at the same potential at time zero, and the voltage sources are ramped gradually to their final values.
  - Steps through the input signals without simulating the circuit. Input waveforms will be reported as specified by the `".print"` (page 137) command.

If **mode** is not specified, T-Spice first performs a DC operating point analysis, without printing the DC operating point analysis results.

**sweep** indicates the beginning of the next nested sweep variable specification. The **sweep** keyword can be omitted if the previous sweep is not of the **list** or **poi** type or if one of the keywords **lin**, **dec**, **oct**, **list**, **poi**, **temp**, **param**, **source**, or **modparam** follows immediately.

Using the **sweep** option with `.tran` or `.ac` (page 76) causes that analysis to be performed for all parameter values of the sweep. It is equivalent to `".step"` (page 158), except that it applies only to one analysis command, while `.step` applies to all analysis commands in the input file. If **sweep** is specified on an analysis command and `.step` is present, the **sweep** sweep is nested inside the `.step` sweep. The **sweep** parameter may be used to specify a parametric sweep, Monte Carlo analysis, or optimization.

For more information about the **restart**, **restarttime**, and **restartfile** options for restarting previously checkpointed simulations, see `".checkpoint"` (page 84).

**Examples**

`.tran 0.5n 100n`

Defines a transient simulation lasting 100 nanoseconds, using time steps of at most 0.5 nanosecond. By default, a DC operating point calculation will first be performed to define a starting condition.

`.tran/preview 4n 4000n`

The input waveforms are reported for 4000 nanoseconds; the rest of the circuit is ignored.

`.tran 1ns 100ns`

Specifies a maximum time step of 1 nanosecond and a total simulation time of 100 nanoseconds.

`.tran/powerup 1ns 100 ns`

Specifies a powerup simulation with no operating point computation.

`.tran 1ns 100 ns start=50ns`

Produces output starting at time 50 nanoseconds.

`.tran 1n 100n sweep temp list 0 27 100 150 -50`
Performs five transient analysis runs, one for each temperature listed. The keyword **temp** specifies the sweep variable, as defined in “**.step**” (page 158).

```
.tran 1n 400n sweep temp -50 150 50
```

This performs five transient analyses at temperatures -50, 0, 50, 100, and 150 degrees Celsius.

```
.tran 0.5u 100u sweep monte=20
```

This performs 20 transient simulations as part of a Monte Carlo analysis. The keyword **monte** defines one of the many sweep options described in “**.step**” (page 158)). For each of the 20 transient analyses, values are randomly chosen for circuit variables, which are assigned probability distributions according to the specified “**Monte Carlo Parameters**” (page 132).

For a demonstration of Monte Carlo analysis in T-Spice, see “Example 2: Monte Carlo Analysis” on page 522.

```
.tran 1n 200n sweep temp list 0 25 75 150
```

This example performs four transient analysis runs at temperatures 0, 25, 75, and 150 degrees Celsius.
.vector

Names a bus and specifies how many bits the bus will contain.

- The bus is connected to a vector-valued current or voltage source, defined by a \texttt{i} or \texttt{v} statement with the \texttt{bus} keyword.
- The input source generates signals composed of bit strings of the length specified in the \texttt{.vector} command.

Syntax

\texttt{.vector bus \{node1 [[,] node2 ...]\}}

- \texttt{bus} \quad Bus name.
- \texttt{node1 node2} \quad Input nodes. If there are \( n \) nodes in a bus, the rightmost \( n \) bits of the input waveform (a binary number) are assigned one by one to these nodes. The last-named \((n)\)th node is assigned the least-significant (rightmost) bit; the \((n-1)\)th node is assigned the next bit to the left; and so on. Extra bits are discarded. Extra nodes are set to zero.

Examples

\texttt{.vector bus1 \{b7 b6 b5 b4 b3 b2 b1 b0\}}

Defines a bus \texttt{bus1} and lists its eight input nodes. The input waveform to these nodes is specified as some number or numbers, convertible to a binary string with at least eight bits, in the accompanying voltage or current source statement.
.warn

The .warn command provides a mechanism for suppressing and customizing warning messages that are issued during T-Spice simulations.

Two basic types of warning message matching are supported: message pattern matching and element name matching. When the pattern keyword is used, then each T-Spice warning message is processed to see if it matches the specified regular expression, and then processed accordingly if it does match. Alternatively, when the type and name keywords are used, then only messages for the specified command, option, or parameter are handled.

The minimum keywords that must be specified are pattern and action, or name, type, and action. All other keywords are optional and are used to more narrowly define the message that will be matched and the subsequent action.

Syntax

```
.warn [pattern="pattern"] [type=type] [name=name] [device=device] [level=level] [value=value] [action=action]
```

- **pattern**: A regular expression
- **type**: The type of error message that is being matched. Supported types are: option, device_parameter | device, model_parameter | model.
- **name**: The name of an option, device parameter, or model parameter.
- **device**: When performing matching of device and model parameter messages, it is the type of device. Possible device values are: bjt | q, capacitor | c, diode | d, mosfet | m, or resistor | r.
- **level**: If it is a model parameter message, restrict the message handler to a specific model level.
- **value**: Match a message only when the parameter or option has a specified value.
- **action**: How the warning should be processed. Supported actions include: default, disable | ignore, once, error

Examples

```
.warn type=model_parameter name=tcbo action=ignore
```

Suppress warning messages for unsupported model parameter tcbo.

```
.warn type=option name=itrprt action=ignore
```

Suppress warning messages for option itrprt.

```
.warn pattern=".*"
```

A regular expression
Suppress all warning messages.

```
.warn pattern=".*Unknown .* parameter .*" action=ignore
```

Suppress all warning messages for unknown model and device parameters.
6 Device Statements

Introduction

This chapter documents the *device statements* of the T-Spice circuit description language.

The *device types* are listed in alphabetical order; each type is associated with a *key letter* (in parentheses). Many statements have “options,” which branch to different modes, and “arguments,” which indicate expressions, nodes, or devices to be used. In the input file, a device statement must begin with its key letter in the first column of the line containing it (no leading spaces). Options and arguments must be separated by spaces or new lines (with line continuation).

Syntax sections in this documentation follow these conventions:

- *Italics* indicate variables to be replaced by actual names, numbers, or expressions.
- Curly brackets `{}` indicate alternative values for the same option or argument.
- Square brackets `[ ]` enclose items that are *not required*.
- Vertical bars `|` separate alternative values for the same option or argument.
- Ellipses `...` indicate items that may be repeated as many times as needed.

These characters are not typed in the input file. All other characters are typed as shown.

For more information, see “Input Conventions” on page 66 and “Simulation Commands” on page 75.
BJT (q)

A transistor with up to four terminals: collector, base, emitter, and (optional) substrate. (BJT stands for bipolar junction transistor.)

Several types of bipolar models are supported in T-Spice:
- SPICE Gummel-Poon model (level 1)
- Vertical Bipolar Inter-Company (VBIC) (level 9)
- Philips MEXTRAM (levels 6, 503, and 504)
- Philips Modella (level 10 and 500)

The substrate is optional so that both discrete and IC BJTs may be modeled correctly.

Syntax

```
q name collector base emitter [substrate] model [[area=A] [areab=B] [areac=C] [M=M] [SCALE=S]]
```

The following device options are available for Gummel-Poon models.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>BJT name</td>
<td></td>
</tr>
<tr>
<td>collector</td>
<td></td>
<td>Collector terminal</td>
<td></td>
</tr>
<tr>
<td>base</td>
<td></td>
<td>Base terminal</td>
<td></td>
</tr>
<tr>
<td>emitter</td>
<td></td>
<td>Emitter terminal</td>
<td></td>
</tr>
<tr>
<td>substrate</td>
<td></td>
<td>Substrate terminal</td>
<td></td>
</tr>
<tr>
<td>model</td>
<td></td>
<td>BJT model name. The model is specified elsewhere in the input file in the form .model name npn/pnp [parameters].</td>
<td></td>
</tr>
</tbody>
</table>

VBIC device statements are different from the Gummel-Poon bipolars. The VBIC model does not contain any terms for explicitly defining geometry or junction areas.

```
q name collector base emitter [substrate] [tmode] model [M=M] [SCALE=S] [tnodeout]
```

`tmode` refers to the thermal node (dt), and `tnodeout` is the flag indicating that the last node in the list of device nodes is the thermal node; otherwise it when four nodes are specified it is unclear as to whether that node is a substrate node or thermal node. (With five nodes the fifth is always thermal.)
Instead, the device SCALE parameter is provided for linearly scaling the device currents and charges. For compatibility with Gummel-Poon devices, the VBIC device statement will also accept the \( M \) multiplicity factor as a synonym for the SCALE parameter.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>scale</td>
<td>S</td>
<td>Scale factor</td>
<td>1</td>
</tr>
</tbody>
</table>

**Examples**

```
qout1 r32 r23 gnd sub1 npnmod
qout2 r32 r23 gnd sub1 npnmod area=2
```

The `area` factor scales the generated current; thus, `qout2` generates twice as much current as `qout1`. 
Capacitor (c)

A two-terminal capacitor. A nonlinear capacitor can be created using the `g`-element with an expression and the `chg` keyword. See “Nonlinear Capacitor” on page 214.

Syntax

```
c name node1 node2 CapValue|C=CapValue|=[M=M] [IC=IC] [scale=scale] [tc1=T1] [tc2=T2] [dtemp=dtemp] .
```

```
oc name node1 node2 POLY c0 [c1 [...] [M=M] .
```

```
oc name node1 node2 modelname [c=C] [M=M] [IC=IC] [scale=scale] [tc1=T1] [tc2=T2] [dtemp=dtemp] [l=length] [w=width] .
```

```
oc name node1 node2 modelname C [T1 [T2 ]] [M=M] [IC=IC] [scale=scale] [dtemp=dtemp] [l=length] [w=width] .
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>Capacitor name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>node1</td>
<td></td>
<td>Positive terminal</td>
<td></td>
<td></td>
</tr>
<tr>
<td>node2</td>
<td></td>
<td>Negative terminal.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>C</td>
<td>Capacitance.</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>POLY</td>
<td></td>
<td>Keyword indicating that the capacitance is a polynomial function.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>c0, c1, ...</td>
<td>C0, C1, ...</td>
<td>Coefficients of the polynomial function for capacitance.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>M</td>
<td>Multiplicity - the number of devices to be placed in parallel.</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>IC</td>
<td></td>
<td>Capacitor voltage initial condition</td>
<td>-</td>
<td>V</td>
</tr>
<tr>
<td>scale</td>
<td>S</td>
<td>Element scale factor</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>T1</td>
<td>First temperature coefficient for capacitance.</td>
<td>0</td>
<td>1/deg</td>
</tr>
<tr>
<td>T2</td>
<td>T2</td>
<td>Second temperature coefficient for capacitance.</td>
<td>0</td>
<td>(1/deg^2)</td>
</tr>
<tr>
<td>dtemp</td>
<td>D_temp</td>
<td>Difference between the capacitor and the circuit temperatures.</td>
<td>0</td>
<td>deg</td>
</tr>
<tr>
<td>l</td>
<td>L</td>
<td>Length of capacitor.</td>
<td>m</td>
<td></td>
</tr>
<tr>
<td>w</td>
<td>W</td>
<td>Width of capacitor</td>
<td>m</td>
<td></td>
</tr>
</tbody>
</table>
Equations

If the first syntax is employed the capacitance is calculated as

\[ C = M \cdot S[1 + T_1(\Delta T) + T_2(\Delta T)^2] \cdot C_0 \]  \hspace{1cm} (6.5)

where

\[ \Delta T = T_{\text{circuit}} + D_{\text{temp}} - T_{\text{nom}} \]  \hspace{1cm} (6.6)

where \( T_{\text{circuit}} \) is set in .temp and \( T_{\text{nom}} \) in .options tnom.

If the second syntax is employed, capacitance is calculated as

\[ C = c_0 + c_1 V + c_2 V^2 + \ldots \]  \hspace{1cm} (6.7)

where \( V \) denotes the voltage between node1 and node2.

If the third or fourth syntax is employed, there must be a matching “.model” on page 116.

Note: When the calculated capacitance is greater than 0.1 F, T-Spice issues a warning message.

Examples

\texttt{cwire w1 gnd 82f}

The example defines a capacitor with a value of 82 femtofarads. A common error is to omit the metric abbreviation on the capacitance value, which can lead to unexpected results.

\texttt{cxx 1 0 poly 0.08 2.08 3.08}

This example defines a capacitor \texttt{cxx} connected between nodes 1 and 0. The capacitance of \texttt{cxx} is described as

\[ C = 0.08 + 1.08 V + 2.08 V^2 + 3.08 V^3 \]  \hspace{1cm} (6.8)

where \( V \) is the voltage between nodes 1 and 0.

A capacitor exhibiting polynomial dependence on its applied voltage can be modeled using the POLY keyword:

\texttt{cl n+ n- POLY c0 'c0*vcc'}

The waveform for this capacitor is illustrated in “Nonlinear Capacitor” on page 214.

\texttt{cl 10 20 capxx 0.02 1.5e-2 5.0e-4 dtemp=20}
This example illustrates the fourth syntax. The capacitor is named \textbf{c1}. Its terminals are connected to nodes 10 and 20. Its model name is \texttt{capxx}. It has two temperature coefficients, $t_{c1} = 1.5e^{-2}$ and $t_{c2} = 5.0e^{-4}$. Its $d_{temp} = 20$. As the model name is \texttt{capxx}, the corresponding \texttt{.model} statement must also contain the word \texttt{capxx}. 
Coupled Transmission Line (u)

A set of coupled transmission lines.

There is no limit (besides physical memory) on the number of transmission lines that can be coupled.

Syntax

\[
\text{u name in1 [in2 [...] in0 out1 [out2 [...] out0 model length=L [lumps=X] [lumptype=Y]}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>Coupled transmission line name.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in1 in2 ...</td>
<td></td>
<td>Input terminals (as many as needed).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out1 out2</td>
<td></td>
<td>Output terminals (as many as needed).</td>
<td></td>
<td></td>
</tr>
<tr>
<td>in0</td>
<td></td>
<td>Input reference terminal.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>out0</td>
<td></td>
<td>Output reference terminal.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>model</td>
<td></td>
<td>CPL coupling model name. The model is specified elsewhere in the input file</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>in the form .model name cpl level=l [[r]={matrix}] [[c]={matrix} [[l]={matrix} [[g]={matrix}]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>length</td>
<td>L</td>
<td>Physical length.</td>
<td>1</td>
<td>m</td>
</tr>
<tr>
<td>lumps</td>
<td>X</td>
<td>Number of lumps used for iterative ladder circuit (ILC) expansion.</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>lumptype</td>
<td></td>
<td>Type of lumps used for ILC expansion. Y is one of the following: 0 = “Gamma”</td>
<td>3</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td></td>
<td>type lumps 1 = “Tee” type symmetric lumps 2= “Pi” type symmetric lumps 3= Hybrid RGT lumps</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Examples

\[
\text{ufine in1 in2 in3 refin out1 out2 out3 refout}
+ \text{cplmod l=lm lumps=3 lumptype=1}
\]
Current Source (i)

A two-terminal ideal current supply.

Exponential, pulse, piecewise linear, frequency-modulated, sinusoidal, arbitrary and customizable (vectorized) waveforms can be specified.

To specify a current source with an equation, use the “Voltage-Controlled Current Source (g)” on page 210 with an expression that may involve the time() function.

Syntax

```
iname node1 node2 [[DC] I] [AC M [P]] [waveform]
```

- **name**: Current source name.
- **node1**: Positive terminal—or bus named by an associated .vector command.
- **node2**: Negative terminal.
- **I**: DC level. (Unit: A)
- **M**: AC Magnitude. (Unit: A)
- **P**: AC Phase (Unit: degrees. Default: 0.)
- **waveform**: Waveform identifier and parameters (see below).

DC, AC, and transient values can be specified independently and in any order.

**waveform** is one of the following:

**Exponential Waveform**

```
exp (Ii Ip [Dr [Tr [Df [Tf]]]])
```

- **Ii**: Initial current. (Unit: amperes.)
- **Ip**: Peak current. (Unit: amperes.)
- **Dr**: Rise time delay. (Unit: seconds. Default: 0.)
- **Tr**: Rise time constant. (Unit: seconds. Default: 0.)
- **Df**: Fall time delay. (Unit: seconds. Default: 0.)
- **Tf**: Fall time constant. (Unit: seconds. Default: 0.)
The formula used is:

\[ I(t) = \begin{cases} 
I_i & 0 \leq t \leq D_r \\
I_i + (I_p - I_i) \left( 1 - \exp \left( \frac{-t - D_r}{T_r} \right) \right) & D_r \leq t \leq D_f \\
I_i + (I_i - I_p) \left( 1 - \exp \left( \frac{-t - D_f}{T_f} \right) \right) & D_f \leq t 
\end{cases} \]  

(6.9)

**Pulse Waveform**

\[
\text{pulse} \ (I_i \ [D \ [Tr \ [Tf \ [Pw \ [Pp]]]]]) \ [\text{ROUND}=\text{RND}]
\]

- **li**: Initial current. (Unit: amperes.)
- **Ip**: Peak current. (Unit: amperes.)
- **D**: Initial delay. (Unit: seconds. Default: 0.)
- **Tr**: Rise time. (Unit: seconds. Default: time step from .tran.)
- **Tf**: Fall time. (Unit: seconds. Default: time step from .tran.)
- **Pw**: Pulse width. (Unit: seconds. Default: stop time from .tran.)
- **Pp**: Pulse period. (Unit: seconds. Default: stop time from .tran.)
- **RND**: Rounding half-interval. A corner at time \( T \) is replaced by a smoothly differentiable polynomial in the interval \( (T - RND, T + RND) \). The maximum \( RND \) is half the distance to the nearest neighboring corner. (Default: 0—no rounding.)

Note that beginning with version 11, T-Spice interprets rise time as the time to go from the initial voltage to the pulse voltage, regardless of which is larger.

**Piecewise Linear Waveform**

\[
\text{pwl} \ (T_1 \ I_1 \ [T_2 \ I_2 \ ...]) \ [\text{ROUND}=\text{RND}] \ [\text{REPEAT}[=\text{Tr}]] \ [\text{TD}=\text{DELAY}]
\]

- **T1 T2**: Time at corner 1, 2, and so on. (Unit: seconds.)
- **I1 I2**: Current at corner 1, 2, and so on. (Unit: amperes.)
- **ROUND**: Rounding half-interval. A corner at time \( T \) is replaced by a smoothly differentiable polynomial in the interval \( (T - RND, T + RND) \). The maximum \( RND \) is half the distance to the nearest neighboring corner. (Default: 0—no rounding.)
- **REPEAT**: Starting time within the specified waveform for an infinite number of repetitions of the subwaveform. If \( Tr \) is not specified, the entire waveform repeats indefinitely (i.e., \( Tr=0 \)). \( Tr \) must be less than or equal to the duration of the waveform. Waveforms can only repeat if the start and end points match. If they do not match, the repeat option is ignored. The REPEAT keyword can be abbreviated to \( R \).
TD

Time delay added to the beginning of the waveform. If you specify corners T1, T2, etc. and TD=DELAY, then the defined current values will actually be applied at effective corner times T1+DELAY, T2+DELAY, etc.

Piecewise Linear Waveform File

\[
\text{pwlfile filename [ROUND=RND] [REPEAT=[Tr]] [TD=DELAY]}
\]

filename

Input file which contains the piecewise linear waveform definition in a series of time, current pairs, one per line.

ROUND

Same meaning as with pwl waveforms

REPEAT

Same meaning as with pwl waveforms

TD

Same meaning as with pwl waveforms

Frequency-Modulated Waveform

\[
\text{sffm (Io Ip [Fc [Xm [Fs]])}}
\]

Io

Offset current. (Unit: amperes.)

Ip

Peak current. (Unit: amperes.)

Fc

Carrier frequency (Unit: Hertz. Default: 1/T, where T is the stop time from .tran.)

Xm

Modulation index. (Default: 0.)

Fs

Signal frequency. (Unit: Hertz. Default: 1/T, where T is the stop time from .tran.)

The formula used is:

\[
I(t) = I_o + I_p \cdot \sin[(2\pi \cdot F_c \cdot t) + Xm \cdot \sin(2\pi \cdot F_s \cdot t)]
\]

(6.10)

Sinusoidal Waveform

\[
\text{sin (Io Ip [Fr [De [Da [Ph]])}}
\]

Io

Offset current. (Unit: amperes.)

Ip

Peak current. (Unit: amperes.)

Fr

Frequency. (Unit: Hertz. Default: 1/T, where T is the stop time from .tran.)

De

Delay time. (Unit: seconds.)

Da

Damping factor. (Unit: 1/seconds.)
**Vectorized Waveform**

A bit pattern consists of a set of numbers (possibly associated with multiplier factors) whose binary representations sequentially specify the “on”/“off” structure of the waveform. The pattern takes the form \( a(b(x) c(y) \ldots) \), where \( a \), \( b \), and \( c \) are the optional multiplier factors and \( x \) and \( y \) are the numbers.

A bus pattern consists of a set of numbers (possibly associated with multiplier factors) whose binary representations—“bit strings”—are grouped together as a waveform bus and treated as a single input. The length of the bit strings is specified by the `.vector` command. If there are \( n \) nodes in a vector, then T-Spice assigns the first \( n \) bits of each bit string to those nodes. Extra bits are discarded. If there are not enough bits, the highest-order bits are set to zero. The leftmost node name in the `.vector` command takes the most significant bit.

Numbers are specified on the device statement in binary, hexadecimal (suffixed by `h`), octal (suffixed by `o`), or decimal (suffixed by `d`) notation. (For decimal representations the number of lower-order bits to be collected is also given.)
Examples

\[ i_1 \ a \ b \ 4.5u \ AC \ 1.0m \ 0.0 \]

\[ i_1 \] has a DC value of 4.5 microamps, an AC magnitude of 1 milliampere, and an AC phase shift of 0 degrees.

\[ i_2 \ n_2 \ GND \ pwl \ (0n \ 0 \ 100n \ 0 \ 101n \ 5 \ 300n \ 5 \ 301n \ 0 \ + \ 500n \ 0 \ 680n \ 5 \ 700n \ 0 \ 880n \ 5 \ 900n \ 0) \]

\[ i_2 \] generates a \[ pwl \] (piecewise linear) input: a single pulse followed by a pair of sawtooth cycles.

\[ i_3 \ n_3 \ GND \ bit \ ((01010 \ 11011) \ on=5.0u \ off=0.0 \ pw=50n \ rt=10n \ ft=30n) \]

\[ i_3 \] generates a \[ bit \] input. Enclosed in braces \{ \} are two binary-valued five-bit patterns specifying the waveform. The two patterns alternate in time. The \[ on \] current value is 5.0 microamps; the \[ off \] current value is zero. The pulse width (\[ pw \]), 50 nanoseconds, is the time the wave is either (ramping up and) on, or (dropping down and) off. The rise time (\[ rt \]), 10 nanoseconds, is the time given for the wave to ramp from off to on; and the fall time (\[ ft \]), 30 nanoseconds, the time given for the wave to drop from on to off.

\[ i_4 \ n_4 \ GND \ bit \ ((5(01010 \ 5(1))) \ pw=10n \ on=5.0u \ off=0.0) \]

\[ i_4 \] generates a repeating \[ bit \] input. Two distinct patterns are given again, but now \[ multiplier factors \] are included. The wave consists of two alternating patterns: the first pattern contains five bits, the second is a single bit. The five-bit pattern is followed by five successive repetitions of the single-bit pattern, and this combination is repeated five times. (The same pattern could be described by \{5(3(01) 4(1))\}.) The pulse width and on and off voltages are again specified, but the rise and fall times take default values.

\[ .vector \ bb \ {n_5 \ n_6 \ n_7 \ n_8} \]
\[ ib \ bb \ GND \ bus \ ((50(Ah) \ 30(7d4) \ 20(1000)) \ pw=5n \ on=5.0u \ off=0.0) \]

The \[ .vector \] command defines the bus waveform generated by current source \[ ib \]. The command assigns the bus a name (\[ bb \]) and specifies by name the number of bits the bus waveform will have (four: \[ n_5 \] through \[ n_8 \]). The current source statement, which contains the \[ bus \] keyword, specifies waveforms with one or more patterns, along with pulse width and level information.

- The first pattern is \[ Ah \] (hex) = 1010 (binary). Thus, using the names given on the \[ .vector \] command, \[ n_5=1, n_6=0, n_7=1, \] and \[ n_8=0 \]. The pattern is repeated 50 times (that is, maintained for a time period equal to the pulse width multiplied by 50).
- The next pattern is \[ 7d4 \]—that is, 7 (decimal) = 111 (binary), or, to four lower-order bits, 0111. So \[ n_5=0, n_6=1, n_7=1, \] and \[ n_8=1 \]. The pattern is repeated 30 times.
- The last pattern is \[ 1000 \] (binary), so \[ n_5=1, n_6=0, n_7=0, \] and \[ n_8=0 \]. The pattern is repeated 20 times.
Current-Controlled Current Source (f)

A two-terminal ideal DC current supply with a level that is a function of one or more control currents.

Syntax

Linear

\[ fname \ node1 \ node2 \ vname1 \ K \ [Options] \]

Polynomial

\[ fname \ node1 \ node2 \ POLY(N) \ vname1 \ [vname2 \ [vname3]] \ P_0 \ P_1 \ P_2 \ldots \ [Options] \]

\[ \text{fname} \]
Current-controlled current source name. Must begin with "f".

\[ \text{node1} \]
Positive terminal. Positive current flows into \text{node1}.

\[ \text{node2} \]
Negative terminal. Positive current flows out of \text{node2}.

\[ K \]
Current gain—the ratio of the output current to the control current.

\[ \text{POLY} \]
Keyword indicating that the output current is a polynomial function of the control currents.

\[ N \]
Number of control currents (valid values 1-6).

\[ \text{vname1} \ \text{vname2} \ldots \]
Name(s) of the voltage source(s) supplying the control current(s).

\[ P_0 \ P_1 \ P_2 \ldots \]
Coefficients of the control polynomial.

Options

\[ [\text{MAX}=\text{value}] \ [\text{MIN}=\text{value}] \ [\text{ABS} = [0 \ | \ 1]] \ [\text{TC1}=\text{value}] \ [\text{TC2}=\text{value}] \ [\text{SCALE}=\text{value}] \]

\[ \text{MAX} \]
Maximum output voltage value.

\[ \text{MIN} \]
Minimum output voltage value.

\[ \text{ABS} \]
Output is absolute value if ABS=1.

\[ \text{TC1}, \ \text{TC2}, \]
First- and second-order temperature coefficients.

\[ \text{SCALE} \]
Element value multiplier.

Current is reckoned positive if it enters a voltage source at its first terminal. A similar convention holds for the current-controlled current source.

The first statement creates a current source with a level equal to \( K \) multiplied by the current through voltage source \( \text{vname1} \).

The second statement creates a current source whose level is a nonlinear polynomial function of the currents through up to three voltage sources. Let:

- \( x = \) current through voltage source \( \text{vname1} \);
\[ y = \text{current through voltage source } vname2 \text{ (if } N \geq 2); \]
\[ z = \text{current through voltage source } vname3 \text{ (if } N \geq 3). \]

Then the controlled current source’s level is defined as follows:

If \( N = 1 \):
\[
P_0 + P_1 x + P_2 x^2 + P_3 x^3 + \ldots \tag{6.12}
\]

If \( N = 2 \):
\[
P_0 + P_1 x + P_2 y + P_3 x^2 + P_4 xy + P_5 y^2 + P_6 x^3 + P_7 xy^2 + P_8 y^3 + \ldots \tag{6.13}
\]

If \( N = 3 \):
\[
P_0 + P_1 x + P_2 y + P_3 z + P_4 x^2 + P_5 xy + P_6 y^2 + P_7 yz + P_8 x^3 + P_9 x^2 y + P_{10} x y^2 + P_{11} x z + P_{12} y^2 x + P_{13} z^2 + P_{14} x^2 y + P_{15} x y^2 + P_{16} z^2 + P_{17} y^2 x + P_{18} y^2 z + P_{19} z^2 + \ldots \tag{6.14}
\]

If \( N = 1 \) and only one polynomial coefficient is specified, it is assumed to be \( P_1 \), to facilitate the specification of linearly-controlled sources.

**Examples**

\texttt{ftest in gnd vin 1.0}

Current-controlled current source \texttt{ftest} has a gain of 1 and is controlled by the current through \texttt{vin}.

\texttt{f1 0 1 vcntrl 2.0}

This defines a current source with a level equal to \( 2 \times i(vcntrl) \), that is, twice the current through \texttt{vcntrl}.

\texttt{f2 0 1 POLY(1) vcntrl 1m 0 2}

This defines a current source with a level equal to \( 10^{-3} + (2 \times i(vcntrl)^2) \).

\texttt{f3 0 1 POLY(2) v1 v2 0 1 2 3}

This defines a current source with a level equal to \( i(v1) + (2 \times i(v2)) + (3 \times i(v1) \times i(v2)) \).

\texttt{f4 0 1 POLY(3) v1 v2 v3 0 1 0 3 0 4}

This defines a current source with a level equal to \( i(v1) + (3 \times i(v3)) + (4 \times i(v1) \times i(v2)) \).
Current-Controlled Voltage Source (h)

A two-terminal ideal DC voltage supply with a level that is a function of one or more controlling currents.

Syntax

Linear

```
h name node1 node2 vname1 K [Options]
```

Polynomial

```
h name node1 node2 POLY(N) vname1 [vname2 [vname3 ]] P0 P1 P2 ... [Options]
```

<table>
<thead>
<tr>
<th>hname</th>
<th>Current-controlled voltage source name. Must begin with &quot;h&quot;.</th>
</tr>
</thead>
<tbody>
<tr>
<td>node1</td>
<td>Positive terminal.</td>
</tr>
<tr>
<td>node2</td>
<td>Negative terminal.</td>
</tr>
<tr>
<td>K</td>
<td>Transresistance—the ratio of the output voltage to the control current.</td>
</tr>
<tr>
<td>M</td>
<td>Multiplicity—the number of devices to be placed in parallel. (Default: 1.) Can be a decimal or integer. Scales down through the subcircuit hierarchy.</td>
</tr>
<tr>
<td>POLY</td>
<td>Keyword indicating that the output voltage is a polynomial function of the control currents.</td>
</tr>
<tr>
<td>N</td>
<td>Number of control currents (valid values 1-6).</td>
</tr>
<tr>
<td>vname1 vname2</td>
<td>Name(s) of the voltage source(s) supplying the control current(s).</td>
</tr>
<tr>
<td>P0 P1 P2 ...</td>
<td>Coefficients of the control polynomial. Current is reckoned positive if it enters a voltage source at its first terminal.</td>
</tr>
</tbody>
</table>

Options

```
[MAX=value] [MIN=value] [ABS =[0 | 1]] [TC1=value] [TC2=value] [SCALE=value]
```

<table>
<thead>
<tr>
<th>MAX</th>
<th>Maximum output voltage value.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIN</td>
<td>Minimum output voltage value.</td>
</tr>
<tr>
<td>ABS</td>
<td>Output is absolute value if ABS=1.</td>
</tr>
<tr>
<td>TC1, TC2,</td>
<td>First- and second-order temperature coefficients.</td>
</tr>
<tr>
<td>SCALE</td>
<td>Element value multiplier.</td>
</tr>
</tbody>
</table>

The first statement creates a voltage source with a level equal to \( K \) multiplied by the current through voltage source \( vname1 \).

The second statement creates a voltage source whose level is a nonlinear polynomial function of the currents through up to three voltage sources. Let
- $x$ = current through voltage source $vname1$;
- $y$ = current through voltage source $vname2$ (if $N \geq 2$);
- $z$ = current through voltage source $vname3$ (if $N \geq 3$).

Then the controlled voltage source’s level is defined as follows:

If $N = 1$:

$$P_0 + P_1x + P_2x^2 + P_3x^3 + \ldots$$  \hspace{1cm} (6.15)

If $N = 2$:

$$P_0 + P_1x + P_2y + P_3x^2 + P_4xy + P_5y^2 + P_6x^3 + P_7xy^2 + P_8y^3 + \ldots$$  \hspace{1cm} (6.16)

If $N = 3$:

$$P_0 + P_1x + P_2y + P_3z + P_4x^2 + P_5xy + P_6yz + P_7z^2 + P_8x^3 + P_9x^2y + P_{10}xy^2 + P_{11}y^2 + P_{12}x^2z + P_{13}xyz + P_{14}xz^2 + P_{15}yz^2 + P_{16}xy^2 + P_{17}x^3 + \ldots$$  \hspace{1cm} (6.17)

If $N = 1$ and only one polynomial coefficient is specified, it is assumed to be $P1$, to facilitate the specification of linearly-controlled sources.

**Examples**

htest in gnd vin 1.23e4

Current-controlled voltage source htest has a transresistance of 12.3 kilohms and is controlled by the current through vin.

h1 0 1 vcntrl 2.0

This defines a voltage source with a level equal to $2 \times i(vcntrl)$, that is, twice the current through vcntrl.

h2 0 1 POLY(1) vcntrl 1m 0 2

This defines a voltage source with a level equal to $10^{-3} + (2 \times i(vcntrl))^2$.

h3 0 1 POLY(2) v1 v2 0 1 2 3

This defines a voltage source with a level equal to $i(v1) + (2 \times i(v2)) + (3 \times i(v1) \times i(v2))$.

h4 0 1 POLY(3) v1 v2 v3 0 1 0 3 0 4

This defines a voltage source with a level equal to $i(v1) + (3 \times i(v3)) + (4 \times i(v1) \times i(v2))$. 
Diode (d)

A two-terminal $p$-$n$ junction diode.

Syntax

```plaintext
d name node1 node2 model [[area=A] [M=M] [L=length] [W=width] [PJ=PJ] [LM=LM] [WM=WM] [LP=LP] [WP=WP]
```

- **name**: Diode name.
- **node1**: Positive terminal ($p$ side).
- **node2**: Negative terminal ($n$ side).
- **model**: Diode model name. This is specified elsewhere in the input file in the form `.model name d [parameters]`
  Schottky barrier diodes may be simulated using an appropriate model specification.
- **A**: Area of the diode. (Units: unitless for level 1, square meters for level 3. *Default*: 1.)
- **M**: Multiplicity—the number of devices to be placed in parallel. (*Default*: 1.)
- **L**: Length of the diode
- **W**: Width of the diode
- **PJ**: Junction periphery. Overrides the model PJ value. (Units: Unitless for level 1, meters for level 3.)
- **LM**: Length of metal capacitor. Overrides the model LM value. (Units: meters, for level 3 only)
- **WM**: Width of metal capacitor. Overrides the model WM value. (Units: meters, for level 3 only)
- **LP**: Length of polysilicon capacitor. Overrides the model LP value. (Units: meters, for level 3 only)
- **WP**: Width of polysilicon capacitor. Overrides the model WP value. (Units: meters, for level 3 only)

Examples

```plaintext
dpn2 n1 n2 dmodel
D3 n3 n4 dmodel 3
```

The **area** factor scales the diode current; thus, **D3** provides three times as much current as **dpn2**, given the same bias conditions.
Chapter 6: Device Statements

Inductor (l)

A two-terminal inductor.

Coupled (mutual) inductors can be defined with the \texttt{k} statement.

Syntax

\begin{verbatim}
\texttt{\textit{name node1 node2} [\textit{L=}] [\textit{M=M}] [\textit{IC=IC}] [\texttt{scale = scale}] [\textit{tc1 = tc1}] [\textit{tc2 = tc2}] [\textit{dtemp = dtemp}] [r=resistance]}
\end{verbatim}

or

\begin{verbatim}
\texttt{\textit{name node1 node2 \textit{L}} [\textit{tc1[tc2]}] [\textit{M=M}] [\textit{IC=IC}] [\texttt{scale = scale}] [\textit{dtemp = dtemp}] [\textit{r=resistance]}}
\end{verbatim}

or

\begin{verbatim}
\texttt{\textit{name node1 node2 POLY} \textit{C0 C1 ...} [\textit{M=M}] [\textit{IC=IC}]}
\end{verbatim}

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td>Inductor name.</td>
</tr>
<tr>
<td>node1</td>
<td></td>
<td>Positive terminal.</td>
</tr>
<tr>
<td>node2</td>
<td></td>
<td>Negative terminal.</td>
</tr>
<tr>
<td>L</td>
<td>(L_0)</td>
<td>Inductance. (Unit: henries. Default: 0.)</td>
</tr>
<tr>
<td>POLY</td>
<td></td>
<td>Keyword indicating that the inductance is a polynomial function.</td>
</tr>
<tr>
<td>C0 C1 ...</td>
<td>(c_0, c_1, c_2) ...</td>
<td>Coefficients of the polynomial. The inductance is (C_0 + (C_1 \times i) + (C_2 \times i^2)) ..., where (i) is the current through the inductor.</td>
</tr>
<tr>
<td>M</td>
<td>(M)</td>
<td>Multiplicity—the number of devices to be placed in parallel. (Default: 1.)</td>
</tr>
<tr>
<td>IC</td>
<td></td>
<td>Inductor voltage initial condition (Unit: volts Default: unspecified).</td>
</tr>
<tr>
<td>scale</td>
<td>(S)</td>
<td>Element scale factor. (Default: 1.)</td>
</tr>
<tr>
<td>Tc1</td>
<td>(T_{c1})</td>
<td>First temperature coefficient for inductance. (Unit: (1/deg C)^2). (Default: 0.)</td>
</tr>
<tr>
<td>Tc2</td>
<td>(T_{c2})</td>
<td>Second temperature coefficient for inductance. (Unit: (1/deg C)^2). (Default: 0.)</td>
</tr>
<tr>
<td>Dtemp</td>
<td>(D_{\text{temp}})</td>
<td>Difference between the inductor and the circuit temperatures. (Unit: Deg C). (Default: 0.)</td>
</tr>
<tr>
<td>R</td>
<td>(R_0)</td>
<td>Parasitic resistance of the inductor. (Unit: ohm). (Default: 0.)</td>
</tr>
</tbody>
</table>
The formula for inductance is:

\[ L = M S [1 + T_{c1}(\Delta T) + T_{c2}(\Delta T)^2]L_0 \]  \hspace{1cm} (6.18)

where

\[ \Delta T = T_{\text{circuit}} + D_{\text{temp}} - T_{\text{nom}} \]  \hspace{1cm} (6.19)

where \( T_{\text{circuit}} \) is set in \text{.temp} and \( T_{\text{nom}} \) in \text{.options} \text{tnom}.

The formula for parasitic resistance is:

\[ R_{\text{parasitic}} = \frac{R_0}{M}. \]  \hspace{1cm} (6.20)

**Note:**

When the calculated inductance is greater than or equal to 0.1 H, T-Spice issues a warning message.

**Examples**

L1 na nb 10u

The example specifies an inductor with a value of 10 microhenries.

L1 a c 25 m=10scale=20R=10 dtemp=20 tcl=1.5e-2 tc2=5e-4
Chapter 6: Device Statements

Instance (x)

An instantiation of a subcircuit definition or a Verilog-A device.

Subcircuit must be defined elsewhere in the input file with a .subckt/ends block. Verilog-A module must be loaded with the “.hdl” on page 98 command.

Nodes several levels deep within a subcircuit hierarchy are named using hierarchical notation in the form xinstance.xinstance.node.

Syntax

\[\text{xname node1 [node2 ...]} \text{ subcircuit | modulename [parameter=X ...] [M=M]}\]

- **name**: Subcircuit instance name.
- **node1 node2 ...**: Specific instance nodes. The order of nodes named corresponds to the order specified by the .subckt command.
- **parameter=X**: Parameter(s) from the subcircuit definition or external model whose default value(s) are to be overridden by the assignment(s) made here. X can be a number or an expression. Subcircuit parameters have local scope. Parameters can be written in any order in both definition and instances. Parameters not specified here take their default values.
- **subcircuit**: Original subcircuit definition name.
- **modulename**: A Verilog-A module name.
- **M**: Multiplicity—the number of representations of parallel instances of the subcircuit. For subcircuits, m is passed to each device in the subcircuit and the device supports its own algorithm for M parallel devices. Sometimes that means scaling currents, sometimes it's the model evaluation code that works in its own way (like Verilog-A $mfactor), and for resistors it does the correct M parallel resistor calculations. M can be any positive integer or decimal. (Default: 1.)

Examples

```
.subckt inv in out Vdd length=1.25u nwidth=2u pwidth=3u
mt1 out in GND GND nmos l='length' w='nwidth'
mt2 out in Vdd Vdd pmos l='length' w='pwidth'
c2 out GND 800f
.ends inv
...xinv1 a1 a2 Vdd inv nwidth=2.5u
```

The .subckt/ends block creates a three-terminal subcircuit (an inverter) and names it inv. The subcircuit consists of two MOSFETs (one n-type and one p-type) and an 800-femtofarad capacitor. The instance statement defines an instance, named inv1, of the inverter subcircuit inv. Following the instance name are the three terminals of the instance (in order corresponding to that of the original subcircuit definition); the name of the subcircuit to which it refers; and a new assignment for parameter nwidth, which overrides the default value assigned in the definition.
Chapter 6: Device Statements

JFET (j)

A transistor with drain, gate, and source terminals and an optional fourth terminal. (JFET stands for junction field effect transistor.)

Syntax

\[ j\text{name\ drain\ gate\ source\ model\ [\{area\=\}\A\]}\ \{M\=\}\M \]

- **name**: JFET name.
- **drain**: Drain terminal.
- **gate**: Gate terminal.
- **source**: Source terminal.
- **model**: JFET model name. This is specified elsewhere in the input file in the form
  \[ .\text{model\ name\ njf|pjf\ [parameters]} \]
- **A**: Area scale factor. (Default: 1.)
- **M**: Multiplicity—the number of devices to be placed in parallel. (Default: 1.)

Examples

\[ j\text{out\ 4\ 8\ 6\ jfet2} \]
\[ j\text{l\ vdd\ in\ out\ jfet2\ 3} \]

The **area** factor scales the generated currents; thus, the currents at the terminals of **j1** are three times those at the terminals of **jout**.
MESFET (z)

A transistor with three or four terminals: drain, gate, and source. (MESFET stands for metal semiconductor field effect transistor.)

Syntax

\[ \text{z} \text{name} \text{ drain gate source} \text{ [bulk]} \text{ model } \text{[[area=]A]} \text{ [l=L]} \text{ [w=W]} \text{ [M=M]} \]

For HSPICE compatibility, you can create a MESFET device in T-Spice using a device name \text{jname} instead of \text{zname}. The syntax for these MESFET device statement is the same.

<table>
<thead>
<tr>
<th>name</th>
<th>MESFET name.</th>
</tr>
</thead>
<tbody>
<tr>
<td>drain</td>
<td>Drain terminal.</td>
</tr>
<tr>
<td>gate</td>
<td>Gate terminal.</td>
</tr>
<tr>
<td>source</td>
<td>Source terminal.</td>
</tr>
<tr>
<td>bulk</td>
<td>Bulk terminal.</td>
</tr>
<tr>
<td>model</td>
<td>MESFET model name. This is specified elsewhere in the input file in the form [ .\text{model name nmf</td>
</tr>
<tr>
<td>A</td>
<td>Area scale factor. (Default: 1.)</td>
</tr>
<tr>
<td>L</td>
<td>Device length. (Unit: meters.)</td>
</tr>
<tr>
<td>W</td>
<td>Device width. (Unit: meters.)</td>
</tr>
<tr>
<td>M</td>
<td>Multiplicity—the number of devices to be placed in parallel. (Default: 1.)</td>
</tr>
</tbody>
</table>

Examples

\text{zout 4 8 6 mfet2} \\
\text{z1 vdd in out mfet2 3} \\
\text{ztest drain gate source vbg nmes1 w=20u l=2u}

The \text{area} factor scales the generated current; thus, the currents at the terminals of \text{z1} are three times those at the terminals of \text{zout}. The third example shows specification of the bulk terminal and of width and length. The area is fixed by the given width and length; any \text{area} specification is overridden by this computed area.
MOSFET (m)

A transistor with four terminals: drain, gate, source, and bulk. (MOSFET stands for metal oxide semiconductor field effect transistor.)

Refer to Additional Model Documentation for complete documentation of the model parameter variations for each MOSFET level.

Unique T-Spice device model parameters can be found in “MOSFET Levels 8, 49 and 53 (BSIM3 Revision 3.3)” on page 453, “Variables for which equations are not given here are as follows.” on page 461, “MOSFET Levels 44 and 55 (EKV Revision 2.6)” on page 482, and “MOSFET Level 57 / 70 (BSIM3SOI and BSIM4SOI)” on page 488.

Syntax

`m name drain gate source bulk model [l=L] [w=W] [ad=Ad] [pd=Pd] [as=As] [ps=Ps] [nrd=Nrd] [nrs=Nrs] [rdc=Rdc] [rsc=Rsc] [rsh=Rsh] [geo=Geo] [M=M] [delvto=delvto] [dtemp=dtemp]

`name`  MOSFET name.

`drain`  Drain terminal.

`gate`  Gate terminal.

`source`  Source terminal.

`bulk`  Bulk terminal.

`model`  MOSFET model name. The model is declared elsewhere in the input file in the form:

```
.model name nmos|pmos
level=1|2|3|4|5|9|13|20|28|30|31|40|47|49|52|100...
[parameters]
```

`L`  Channel length. (Unit: meters. Default: set by the .options defl command.)

`W`  Channel width. (Unit: meters. Default: set by the .options defw command.)

`Ad`  Drain area. (Unit: square meters. Default: see “Drain area” on page 195.)

`Pd`  Drain perimeter. (Unit: meters. Default: see “Drain perimeter” on page 196.)

`As`  Source area. (Unit: square meters. Default: see “Source area” on page 195.)

`Ps`  Source perimeter. (Unit: meters. Default: see “Source perimeter” on page 196.)

`Nrd`  Number of squares of diffusion—drain. (Default: set by the .options defnrd command.)

This parameter is only valid for MOS level 2, BSIM3, BSIM4, and HiSIM MOSFET models.
Chapter 6: Device Statements

**MOSFET** (m)

Default values for `Ad`, `Pd`, `As`, and `Ps` depend on the `acm` model parameter.

The parasitic diode characteristics are determined by the MOSFET device parameters `as`, `ad`, `pd`, `ps`, and `geo`, as well as the MOSFET model parameters `acm`, `cj`, `cjsw`, `cjgate`, `js`, `jsw`, `is`, `n`, `nds`, `vnds`, and `hdif`. The quantity `weff` also plays a role in determining default values for source and drain areas and perimeters for some values of `acm`.

The parasitic diode equations have been modified from the standard diode equations, in an effort to improve compatibility with other SPICE simulators and to improve simulator convergence. The diode charge/capacitance equations are unchanged; they are the same as for regular diodes. The DC current equations for MOSFET parasitic diodes are as follows.

If the MOSFET bulk-source voltage `vbs` is positive (the bulk-source diode is forward-biased), then the bulk-source diode’s DC current is given by

\[ i_{bs} = isatbs \times (\exp(vbs/(n \times vt)) - 1) \]  

(6.21)

where \( vt \) = \( kT/q \) (the thermal voltage), and the diode’s saturation current `isatbs` is

\[ isatbs = (js \times aseff) + (jsw \times pseff) \]  

(6.22)

if that value is positive, or is zero otherwise. The effective source area `aseff` and perimeter `pseff` are computed as described below, depending on the value of the `acm` parameter.

Similarly, if the MOSFET bulk-drain voltage `vds` is positive (the bulk-drain diode is forward-biased), the bulk-drain diode’s DC current is

\[ i_{bd} = isatbd \times (\exp(vbd/(n \times vt)) - 1) \]  

(6.23)

where the diode saturation current `isatbd` is

\[ isatbd = (js \times adeff) + (jsw \times pdeff) \]  

(6.24)
if that value is positive, or is otherwise. The effective drain area \( a_{\text{def}} \) and perimeter \( p_{\text{def}} \) are computed as described below.

The exponential function in both diodes is replaced by a linear extension when the current is larger than the value of the \( \text{expli} \) model parameter. The linear extension is chosen such that the diode current function is continuously differentiable at the transition point where the diode current equals \( \text{expli} \).

The \( n \) parameter is now supported for MOSFET parasitic diodes.

When a MOSFET parasitic diode with saturation current \( \text{isat} \) is reverse-biased with a negative voltage \( v_{\text{di}} \), then its current \( i_{\text{di}} \) behaves as follows.

If \( 0 > v_{\text{di}} > v_{\text{nds}} \):

\[
i_{\text{di}} = \text{isat} \times v_{\text{di}}
\]

(6.25)

If \( v_{\text{di}} < v_{\text{nds}} \):

\[
i_{\text{di}} = \text{isat} \times (v_{\text{nds}} + (v_{\text{di}} - v_{\text{nds}})/\text{nds})
\]

(6.26)

The effective source and drain areas and perimeters are computed as in the chart below, depending on the value of the \( \text{acm} \) parameter. If \( \text{acm} = 3 \), the \( \text{geo} \) device parameter affects these calculations. The \( \text{geo} \) parameter is used to handle stacked MOSFET devices properly, and it can have the following values:

- \( \text{geo} = 0 \) (default): the drain and the source are not shared by other devices.
- \( \text{geo} = 1 \): the drain is shared with another device.
- \( \text{geo} = 2 \): the source is shared with another device.
- \( \text{geo} = 3 \): the drain and the source are shared with other devices.

The \( \text{geo} \) parameter may be specified on the MOSFET device statement, at any point after the model name.

Each parasitic diode inherits its multiplicity factor \( m \) from its “parent” MOSFET. The values of \( \text{defas} \), \( \text{defad} \), and \( \text{moscap} \) are specified using \( \text{.options} \).

<table>
<thead>
<tr>
<th>( \text{acm} )</th>
<th>( \text{Source area} )</th>
<th>( \text{Drain area} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \text{as} \times \text{wmlt}^2 )</td>
<td>( \text{ad} \times \text{wmlt}^2 )</td>
</tr>
<tr>
<td>1</td>
<td>( \text{weff} \times \text{wmlt} )</td>
<td>( \text{weff} \times \text{wmlt} )</td>
</tr>
<tr>
<td>2</td>
<td>( \text{as} \times \text{wmlt}^2 )</td>
<td>( \text{weff} \times \text{wmlt} )</td>
</tr>
<tr>
<td>3</td>
<td>( \text{as} \times \text{wmlt}^2 )</td>
<td>( \text{ad} \times \text{wmlt}^2 )</td>
</tr>
</tbody>
</table>

\( \text{as} \times \text{wmlt}^2 \) (otherwise)
The parasitic drain/source diodes’ sidewall capacitance now makes use of the new \( \text{cjgate} \) parameter, which describes the sidewall capacitance per unit length along the gate edge. If \( \text{cjgate} \) is specified and the MOSFET’s effective width \( \text{weff} \) is not greater than the diode’s perimeter, then the total sidewall capacitance is given by:

\[
\text{csw} = \text{cjsw} \times (p - \text{weff}) + (\text{cjgate} \times \text{weff})
\]  

(6.27)

where \( p \) is \( \text{ps} \) for a source diode or \( \text{pd} \) for a drain diode. Otherwise, if \( \text{cjgate} \) is not specified or \( \text{weff} \geq p \), the total sidewall capacitance is:

\[
\text{csw} = \text{cjsw} \times p
\]  

(6.28)

### Examples

```
m12 n1 n2 GND GND ndep l=10u w=5u ad=100p as=100p pd=40u ps=40u
```
Mutual Inductor (k)

A coupled pair of inductors.

Syntax

\[ k \text{name} \text{inductor1 inductor2 } K \]

- \text{name} \quad Mutual inductor name.
- \text{inductor1} \quad First inductor.
- \text{inductor2} \quad Second inductor.
- \text{K} \quad Coefficient of coupling (0 < K \leq 1).

Examples

\[ k1 \text{La Lb 10u} \]

The example illustrates coupling between two inductors \text{La} and \text{Lb}, defined elsewhere in the input file:

\[ \text{La node1a node2a 10m} \]
\[ \text{Lb node1b node2b 20m} \]

The order of node naming on the inductor statements determines the relative directions of current flow in the mutual inductor. The current flow from \text{node1a} to \text{node2a} (inductor La) is in the same direction as from \text{node1b} to \text{node2b} (inductor Lb). To reverse the current flow in either inductor, reverse the node order on the appropriate inductor statement.
Resistor (r)

A two-terminal resistor.

The resistance $R$ is influenced by the temperature as follows:

$$R = N (1 + AT + BT^2)$$

$$T = Ta - Tn$$

where $N, A, B$ are device parameters described below; $Ta$ (the “ambient” temperature) is set by the .temp command; and $Tn$ (the “nominal” temperature) is set by the .options tnom command.

Resistors can be specified using geometric and physical parameters such as $c, l, w$, and $rsh$. For a description of how resistance is calculated using these parameters, refer to the device model “Resistor” on page 498.

Optional capacitors may be included between the terminals and a bulk node (usually ground) to obtain a simple transmission line model.

Syntax

```plaintext
r name node1 node2 r=r [resistor_parameters]
```

or

```plaintext
r name node1 node2 r [tc1(tc2)] [resistor_parameters]
```

or

```plaintext
r name node1 node2 modelname + [r=r] [resistor_parameters]
```

or

```plaintext
r name node1 node2 modelname r [tc1(tc2)] [resistor_parameters]
```

**name**

Resistor name.

**node1**

Positive terminal.

**node2**

Negative terminal.

**modelname**

Name of resistor model. Must match .model name when type is r. For additional information, see “.model” on page 116.

**r=resistance**

Nominal resistance. (*Unit: ohms.*)

In the first syntax and the third syntax, the **resistor_parameters** field is of the form:

```plaintext
[tcl=tcl] [tc2(tc2)] [noise=noise] [m=mult] [scale=devscale] [ac=acres]
[dtemp=dtemp] [l=l] [w=w] [c=c]
```

In the second syntax and the fourth syntax, the **resistor_parameters** field is of the form:
Chapter 6: Device Statements

T-Spice User Guide

Resistor (r)

\[
\text{noise} = \text{noise} \\
\text{m} = \text{mult} \\
\text{scale} = \text{scale} \\
\text{ac} = \text{acres} \\
\text{dtemp} = \text{dtemp} \\
\text{l} = \text{l} \\
\text{w} = \text{w} \\
\text{c} = \text{c}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(tc1)</td>
<td>First-order temperature coefficient. (Default: resistor model parameter (tc1r); 0 if no model is specified.)</td>
</tr>
<tr>
<td>(tc2)</td>
<td>Second-order temperature coefficient. (Default: resistor model parameter (tc2r); 0 if no model is specified.)</td>
</tr>
<tr>
<td>(noise)</td>
<td>Noise source multiplier. (noise=0) eliminates resistor noise. (Default: resistor model parameter (noise); 1 if no model is specified.)</td>
</tr>
<tr>
<td>(mult)</td>
<td>Multiplicity—the number of devices to be placed in parallel. (Default: 1.)</td>
</tr>
<tr>
<td>(devscale)</td>
<td>Multiplies resistance and capacitance of device. (Default: 1.)</td>
</tr>
<tr>
<td>(acres)</td>
<td>Specifies device resistance during AC analysis. (Default: resistor model parameter (rac); if no model is specified, default is DC resistance.)</td>
</tr>
<tr>
<td>(dtemp)</td>
<td>Specifies the difference between the device temperature and the general circuit operating temperature. (Default: 0.)</td>
</tr>
<tr>
<td>(l)</td>
<td>Resistor length. Scaled length is obtained by multiplying (l) by \text{.options scale} (not the element parameter \text{devscale}, above) or the resistor model parameter \text{shrink}. (Default: resistor model parameter (l).)</td>
</tr>
<tr>
<td>(w)</td>
<td>Resistor width. Scaled width is obtained by multiplying (w) by \text{.options scale} (not the element parameter \text{devscale}, above) or the resistor model parameter \text{shrink}. (Default: resistor model parameter (w).)</td>
</tr>
<tr>
<td>(c)</td>
<td>Capacitance between node2 and a bulk node specified as a model parameter. Multiplied by \text{.options scale}. (Default: resistor model parameter \text{cap}.)</td>
</tr>
</tbody>
</table>

**Note:**
If T-Spice Pro calculates the effective resistance, \(R_{\text{eff}}\), to be less than \(10^{-5} \Omega\), then a warning message is issued and the effective resistance is automatically assigned a value of \(10^{-5} \Omega\). See the model description “Resistor” on page 498 for calculation of \(R_{\text{eff}}\).

**Examples**

\[r1 \ 2 \ \ 1 \ \ 30K \ \ \text{TC}=1e-2,1e-4\]

This produces a resistor of resistance 30 kilohms at the nominal temperature \(t_{\text{nom}}\). If the temperature \(T\) is different from \(t_{\text{nom}}\), the resistance is \(30,000*(1+0.01*(T-t_{\text{nom}})+0.0001*(T-t_{\text{nom}})*(T-t_{\text{nom}}))\). For example, if the circuit temperature is 127 degrees and \(t_{\text{nom}}\) is 27 degrees, the resistance is \(30,000*(1+0.01*100+0.0001*100*100) = 90,000 \text{ Ohms}\).

\[r1 \ n1 \ n2 \ \text{rmod} \ l=5u \ w=10u\]
\[.model \ \text{rmod} \ r \ \text{rsh}=1k \ \text{cap}=10pf \ \text{cratio}=0.5\]
This example creates a resistor between nodes \texttt{n1} and \texttt{n2} of size 500 Ohms \((r = \frac{l^*rsh}{w})\) as well as two capacitors, 5 picofarads each, one between \texttt{n1} and \texttt{Gnd}, the other between \texttt{n2} and \texttt{Gnd}.

or

\texttt{C1 N1 N2 C=5f}
Voltage- or Current-Controlled Switch (s)

A switch is implemented as a resistor between node1 and node2, whose resistance is controlled by the controlling voltage or current. (See also the device model “Switch” on page 502.)

Syntax

The general syntax for T-Spice's voltage-controlled switch element is:

\[ \text{sname node1 node2 control1 control2 modelname} \]

The syntax for a current-controlled switch is:

\[ \text{sname node1 node2 vsourcename modelname} \]

- **name**: Switch name.
- **node1**: Positive terminal.
- **node2**: Negative terminal.
- **control1**: Name of the voltage source supplying the control voltage.
- **control2**: Name of the voltage source supplying the control voltage.
- **vsourcename**: Controlling current for a current-controlled switch.
- **modelname**: Name of resistor model. Must match `.model modelname` when type is `sw` or `csw`. For additional information, see “.model” on page 116.

Resistance is `roff` when the switch is off and `ron` when the switch is on. The switch is on when the control voltage or current for the switch is greater than its threshold voltage or current.

The `ron`, `roff`, and threshold values for the switch are specified in a `.model` statement whose model name matches the `modelname` on the device statement.

T-Spice switch elements can display hysteresis, so that the threshold value is different when the control voltage/current is increasing than when it is decreasing. For a voltage-controlled switch, the threshold voltage is `vt` when `v(control1, control2)` is increasing, and `vt-vh` when `v(control1, control2)` is decreasing. For a current-controlled switch the threshold current is `it` when `i(vsourcename)` is increasing, and `it-ih` when `i(vsourcename)` is decreasing. The switch is on when the control voltage or current is greater than the threshold value.

The `dv` and `di` parameters define a small interval around the threshold in which a smooth transition between `ron` and `roff` is made.

Examples

**Voltage-Controlled Switch**

The following two examples would both produce voltage-controlled switches:

\[ \text{s1 n1 n2 c1 c2 swmod} \]
\[ \text{.model swmod sw ron=1 roff=1e12 vt=2.5} \]
and

```plaintext
s1 out 0 0 in swmod
.model swmod sw vt=0 dv=0.2
```

The waveform for this switch is illustrated in “Voltage-Controlled Resistor” on page 213.

**Current-Controlled Switch**

The following example demonstrates the modeling of a current-controlled switch:

```plaintext
s1 out 0 vin swmod
.model swmod csw it=0.7 di=0.1
```

This switch would produce the following waveform:
Transmission Line (t)

A mechanism for “lossless” or “lossy” signal propagation.

The “lossless” transmission line is described by characteristic impedance and delay; the “lossy” transmission line is described by RLCG parameters.

Syntax

“Lossless” Transmission Line

```
t name node1 node2 node3 node4 z0=Z [td=D] [f=F [nl=N] ]
```

- **name**: Transmission line name.
- **node1 ... node4**: Terminals. `node1` (+) and `node2` (−) are at one end of the transmission line, `node3` (+) and `node4` (−) at the opposite end.
- **Z**: Impedance. (*Unit: ohms.*)
- **D**: Transmission delay. The delay may instead be specified indirectly from `F` and `N`. (*Unit: seconds.*)
- **F**: Line frequency. (*Unit: Hertz. Default: 1 x 10⁹.*)
- **N**: Normalized number of wavelengths. The transmission delay is the ratio of the wavelength number `N` to the line frequency `F`. (*Default: 0.25.*)

“Lossy” Transmission Line

```
t name node1 node2 node3 node4 r=R l=L c=C g=G length=W [lumps=X] [lumptype=Y]
```

- **name**: Transmission line name.
- **node1 ... node4**: Terminals. `node1` (+) and `node2` (−) are at one end of the transmission line, `node3` (+) and `node4` (−) at the opposite end.
- **R**: Distributed resistance. (*Unit: ohms/meter.*)
- **L**: Distributed inductance. (*Unit: henries/meter.*)
- **C**: Distributed capacitance. (*Unit: farads/meter.*)
- **G**: Distributed conductance. (*Unit: siemens/meter.*)
- **W**: Physical length. (*Unit: meters.*)
- **X**: Number of lumps used for iterative ladder circuit (ILC) expansion. (*Default: 1.*)
- **Y**: Type of lumps used for ILC expansion (see below). (*Default: 3.*)
\( Y \) is one of the following:

- **0**  “Gamma” type lumps.
- **1**  “Tee” type symmetric lumps.
- **2**  “Pi” type symmetric lumps.
- **3**  Hybrid RGT lumps (default).

**Examples**

```
tline2 pad2 GND pin2 GND z0=100 td=10ns
tline3 drive GND out GND z0=300 f=100meg nl=.1
```
Voltage Source (v)

A two-terminal ideal voltage supply.

Exponential, pulse, piecewise linear, frequency-modulated, sinusoidal, and customizable (vectorized) waveforms are available.

Voltage sources whose waveform is described using an expression can be created using the \( e \)-element with an expression and the \( \text{time() \ function} \) function.

Syntax

\[
\text{v} \ name \ \text{node1} \ \text{node2} \ [[\text{DC}] \ \text{V}] \ [\text{AC} \ \text{M} \ [\text{P}]] \ [\text{waveform}]
\]

- \text{name} \quad Voltage source name.
- \text{node1} \quad Positive terminal—or bus named by an associated \text{.vector} command.
- \text{node2} \quad Negative terminal.
- \text{V} \quad DC level between \text{node1} and \text{node2}. (Unit: volts. Default: 0.)
- \text{waveform} \quad Waveform identifier and parameters (see below).
- \text{M} \quad AC magnitude. (Unit: volts.)
- \text{P} \quad AC phase. (Unit: degrees. Default: 0.)

DC, AC, and transient values can be specified independently and in any order.

\text{waveform} \quad \text{is one of the following:}

**Exponential Waveform**

\[
\text{exp} \ (\text{Vi} \ \text{Vp} \ [\text{Dr} \ [\text{Tr} \ [\text{Df} \ [\text{Tf}]])]])
\]

- \( \text{Vi} \) \quad Initial voltage. (Unit: volts.)
- \( \text{Vp} \) \quad Peak voltage. (Unit: volts.)
- \( \text{Dr} \) \quad Rise time delay. (Unit: seconds. Default: 0.)
- \( \text{Tr} \) \quad Rise time constant. (Unit: seconds. Default: 0.)
- \( \text{Df} \) \quad Fall time delay. (Unit: seconds. Default: 0.)
- \( \text{Tf} \) \quad Fall time constant. (Unit: seconds. Default: 0.)

**Pulse Waveform**

\[
\text{pulse} \ (\text{Vi} \ \text{Vp} \ [\text{D} \ [\text{Tr} \ [\text{Tf} \ [\text{Pw} \ [\text{Pp}]])]]) \ [\text{ROUND=}\text{RND}]
\]

- \( \text{Vi} \) \quad Initial voltage. (Unit: volts.)
Chapter 6: Device Statements Voltage Source (v)

Note that rise time is not necessarily a “rise” time, but is the time to go from the initial voltage to the pulse voltage, regardless of whether it's smaller or larger.

**Piecewise Linear Waveform**

```
pwl (T1 V1 [T2 V2 ...]) [ROUND=RND] [REPEAT[=Tr]] [TD=DELAY]
```

- **T1 T2**: Time at corner 1, 2, and so on. (Unit: seconds.)
- **V1 V2**: Voltage at corner 1, 2, and so on. (Unit: volts.)
- **ROUND**: Rounding half-interval. A corner at time $T$ is replaced by a smoothly differentiable polynomial in the interval $(T-RND, T+RND)$. The maximum $RND$ is half the distance to the nearest neighboring corner. (Default: 0—no rounding.)
- **REPEAT**: Starting time within the specified waveform for an infinite number of repetitions of the subwaveform. If $Tr$ is not specified, the entire waveform repeats indefinitely (i.e., $Tr=0$). $Tr$ must be less than or equal to the duration of the waveform. Waveforms can only repeat if the start and end points match. If they do not match, the repeat option is ignored. The **REPEAT** keyword can be abbreviated to **R**.

For example, a 100 ns PWL wave and **REPEAT=60ns** repeats the 60ns - 100ns portion of the waveform.

- **TD**: Time delay added to the beginning of the waveform. If you specify corners $T1$, $T2$, etc. and **TD=DELAY**, then the defined voltage values will actually be applied at effective corner times $T1+DELAY$, $T2+DELAY$, etc.

**Piecewise Linear Waveform File**

```
pwlfile filename [ROUND=RND] [REPEAT[=Tr]] [TD=DELAY]
```

- **filename**: Input file which contains the piecewise linear waveform definition in a series of **time**, **voltage** pairs, one per line.
- **ROUND**: Same meaning as with **pwl** waveforms
**Frequency-Modulated Waveform**

\[ \text{sffm} \ (Vo \ Vp \ [Fc \ [Xm \ [Fs]]]) \]

- **Vo**: Offset voltage. (*Unit*: volts.)
- **Vp**: Peak voltage. (*Unit*: volts.)
- **Fc**: Carrier frequency (*Unit*: Hertz. *Default*: \(1/T\), where \(T\) is the stop time from `.tran`.)
- **Xm**: Modulation index. (*Default*: 0.)
- **Fs**: Signal frequency. (*Unit*: Hertz. *Default*: \(1/T\), where \(T\) is the stop time from `.tran`.)

**Sinusoidal Waveform**

\[ \text{sin} \ (Vo \ Vp \ [Fr \ [De \ [Da \ [Ph]]]]) \]

- **Vo**: Offset voltage. (*Unit*: volts.)
- **Vp**: Peak voltage. (*Unit*: volts.)
- **Fr**: Frequency. (*Unit*: Hertz. *Default*: \(1/T\), where \(T\) is the stop time from `.tran`.)
- **De**: Delay time. (*Unit*: seconds.)
- **Da**: Damping factor. (*Unit*: 1/seconds.)
- **Ph**: Phase advance. (*Unit*: degrees.)

**Vectorized Waveform**

\[ \text{bit} \ |
\text{bus} \ ((\text{pattern}) \ [on=A] \ [off=Z] \ [delay=D] \ [pw=P] \ [rt=R] \ [ft=F] \ [lt=L] \ [ht=H]) \ [ROUND=RND] \]

- **pattern**: An expression consisting of one or more string or string-multiplier combinations (see below).
- **A**: On voltage (*Unit*: volts. *Default*: 0.001.)
- **Z**: Off voltage (*Unit*: volts. *Default*: 0.)
- **D**: Delay time. (*Unit*: seconds. *Default*: 0.)
- **P**: Pulse width: \(P = R + T_A = F + T_Z\), where \(T_A\) is the time during a pulse where the voltage is “on” \((V = A)\) and \(T_Z\) is the time during a pulse where the voltage is “off” \((V = Z)\). (*Unit*: seconds. *Default*: \(10 \times 10^{-9}\).)
A bit pattern consists of a set of numbers (possibly associated with multiplier factors) whose binary representations sequentially specify the “on”/“off” structure of the waveform. The pattern takes the form \(a(b(x) c(y) \ldots)\), where \(a\), \(b\), and \(c\) are the optional multiplier factors and \(x\) and \(y\) are the numbers.

A bus pattern consists of a set of numbers (possibly associated with multiplier factors) whose binary representations—“bit strings”—are grouped together as a waveform bus and treated as a single input. The length of the bit strings is specified by the .vector command. If there are \(n\) nodes in a vector, then T-Spice assigns the first \(n\) bits of each bit string to those nodes. Extra bits are discarded. If there are not enough bits, the highest-order bits are set to zero. The leftmost node name in the .vector command takes the most significant bit.

Numbers are specified on the device statement in binary, hexadecimal (suffixed by \(h\)), octal (suffixed by \(o\)), or decimal (suffixed by \(d\)) notation. (For decimal representations the number of lower-order bits to be collected is also given.)

Examples

\[v1\ n1\ \text{GND} \ \text{sin} \ (2.5\ 2.5\ 30\text{MEG} \ 100\text{n})\]

\(v1\) generates a \textit{sin} (sinusoidal) input. It has an amplitude of 2.5 volts, a frequency of 30 MHz, an offset of 2.5 volts from system ground, and a time delay of 100 nanoseconds after the start of the simulation before the wave begins.

\[v2\ n2\ \text{GND} \ \text{bit} \ ([01010\ 11011])\ \text{on}=5.0\ \text{off}=0.0\ \text{pw}=50\text{n} \ \text{rt}=10\text{n} \ \text{ft}=30\text{n}\]

\(v2\) generates a \textit{bit} input. Enclosed in braces \{\} are two binary-valued five-bit patterns specifying the waveform. The two patterns alternate in time. The \textit{on} voltage value is 5.0 volts; the \textit{off} voltage value is zero. The pulse width (\(\text{pw}\)), 50 nanoseconds, is the time the wave is either (ramping up and) on, or (dropping down and) off. The rise time (\(\text{rt}\)), 10 nanoseconds, is the time given for the wave to ramp from off to on; and the fall time (\(\text{ft}\)), 30 nanoseconds, the time given for the wave to drop from on to off.

\[v3\ n3\ \text{GND} \ \text{bit} \ ([5(01010\ 5(1))])\ \text{pw}=10\text{n} \ \text{on}=5.0\ \text{off}=0.0\]

\(v3\) generates a repeating \textit{bit} input. Two distinct patterns are given again, but now \textit{multiplier factors} are included. The wave consists of two alternating patterns: the first pattern contains five bits, the second is a single bit. The five-bit pattern is followed by five successive repetitions of the single-bit pattern, and this combination is repeated five times. (The same pattern could be described by \(5(3(01) 4(1)))\). The pulse width and on and off voltages are again specified, but the rise and fall times take default values.

\[.\text{vector} \ bb\ \{n7\ n6\ n5\ n4\}
\]

\(vb\ bb\ \text{GND} \ \text{bus} \ ([5(0Ah)\ 30(7d4)\ 20(1000)])\ \text{pw}=5\text{n} \ \text{on}=5.0\ \text{off}=0.0\]
The `.vector` command defines the bus waveform generated by voltage source `vb`. The command assigns the bus a name (`bb`) and specifies by name the number of bits the bus waveform will have (four: `n7` through `n4`). The voltage source statement, which contains the `bus` keyword, specifies waveforms with one or more patterns, along with pulse width and level information.

- The first pattern is `Ah` (hex) = 1010 (binary). Thus, using the names given on the `.vector` command, `n7`=1, `n6`=0, `n5`=1, and `n4`=0. The pattern is repeated 50 times (that is, maintained for a time period equal to the pulse width multiplied by 50).

- The next pattern is `7d4`—that is, 7 (decimal) = 111 (binary), or, to four lower-order bits, 0111. So `n7`=0, `n6`=1, `n5`=1, and `n4`=1. The pattern is repeated 30 times.

- The last pattern is `1000` (binary), so `n7`=1, `n6`=0, `n5`=0, and `n4`=0. The pattern is repeated 20 times.
Voltage-Controlled Current Source (g)

A two-terminal ideal DC current supply with an output current level that is a function of one or more controlling voltages. The device charge storage can also be defined as a function of controlling voltages.

This device can be utilized to model a wide variety of elements, including voltage-controlled resistors, nonlinear capacitors, voltage-controlled capacitors, switch-level MOSFETs, and diodes. See “Examples” on page 212.

Syntax

**Linear**

```
g name node1 node2 na1 nb1 K [Options]
```

**Piecewise Linear**

```
g name node1 node2 PWL na1 nb2 v1 i1 v2 i2 ... vn in [Options]
```

**Polynomial**

```
g name node1 node2 POLY(N) na1 nb2 [na2 mb2 ...] P0 P1 P2 ... [Options]
```

**LaPlace Transform**

```
g name node1 node2 LAPLACE na1 nb1 a1 a2 ... am [/ b1 b2 ... bm] [Options]
```

**Nonlinear | Behavioral**

```
g name node1 node2 [cur='expression'] [chg='expression'] [Options]
```

- **gname** Voltage-controlled current source name. Must begin with "g".
- **node1** Positive terminal. Positive current flows into `node1`.
- **node2** Negative terminal. Positive current flows out of `node2`.
- **K** Transconductance—the ratio of the output current to the control voltage.
- **PWL** Keyword indicating that the output current is a piecewise linear function of the control voltages.
- **POLY** Keyword indicating that the output current is a polynomial function of the control voltages.
- **N** Number of control voltages (valid values 1-6).
- **LAPLACE** Keyword indicating that the output current is described via a Laplace transform function
- **cur** Keyword indicating an expression `expression` that defines current flowing through the device.
- **chg** Keyword indicating an expression `expression` that defines terminal charges of the device. Used to define the capacitance of a nonlinear capacitor.
- **expression** Expression involving any node voltages and source currents.
Chapter 6: Device Statements

Voltage-Controlled Current Source (g)

Options

[M=value] [MAX=value] [MIN=value] [ABS=[0 | 1]] [TC1=value] [TC2=value]
[SCALE=value]

M
- Multiplicity—the number of devices to be placed in parallel. (Default: 1.) Can be a decimal or integer. Scales down through the subcircuit hierarchy.

MAX
- Maximum output voltage value.

MIN
- Minimum output voltage value.

ABS
- Output is absolute value if ABS=1.

TC1, TC2,
- First- and second-order temperature coefficients.

SCALE
- Element value multiplier.

Current is reckoned positive if it enters the g element at its first terminal.

Linear Functions

The first form of voltage-controlled current sources creates a current source with a level equal to \( K \) multiplied by the voltage across the node pair \( na1 \text{ } nb1 \).

Piecwise Linear Functions

The next form of voltage-controlled current sources creates a current source with a level that varies according to a voltage versus current PWL function table. The PWL table relates the controlling voltage across the node pair \( na1 \text{ } nb1 \) to the output current across the node1 node2 branch.

The \( x \) values of the PWL function must be in increasing order.

Polynomial Functions

The third form creates a current source whose level is a nonlinear polynomial function of the voltages across one or more node pairs. Let

- \( x = \) voltage across node pair \( na1 \text{ } nb1 \);  
- \( y = \) voltage across node pair \( na2 \text{ } nb2 \) (if \( N \geq 2 \));  
- \( z = \) voltage across node pair \( na3 \text{ } nb3 \) (if \( N \geq 3 \)).
Then the controlled voltage source’s level is defined as follows:

If \( N = 1 \):

\[
P_0 + P_1 x + P_2 x^2 + P_3 x^3 + ...
\]

(6.29)

If \( N = 2 \):

\[
P_0 + P_1 x + P_2 y + P_3 x^2 + P_4 x y + P_5 y^2 + P_6 x^3 y + P_7 y^3 + ...
\]

(6.30)

If \( N = 3 \):

\[
P_0 + P_1 x + P_2 y + P_3 x^2 + P_4 x y + P_5 y^2 + P_6 x^3 y + P_7 y^3 + P_8 x^2 y + P_9 x y^2 + P_{10} x^3 y + P_{11} y^3 x + P_{12} x^2 y^2 + P_{13} x y^3 + P_{14} y^3 + ...
\]

(6.31)

If \( N = 1 \) and only one polynomial coefficient is specified, it is assumed to be \( P1 \), to facilitate the specification of linearly-controlled sources.

**Laplace Functions**

With the Laplace keyword, the current source is implemented as a Laplace transfer function.

(6.32)

\[
H(s) = \frac{a_0 + a_1 s + a_2 s^2 + ... + a_m s^m}{b_0 + b_1 s + b_2 s^2 + ... + b_n s^n}
\]

**Expression-Controlled Functions**

The final form of voltage controlled current sources uses mathematical expressions ("Expressions" on page 70) to define the output current and charge functions. At least one of the keywords `cur` or `chg` must be specified.

**Examples**

```
Gtest in out n10 n17 -2.314
```

`Gtest` is a `g` element. Its terminals are connected to nodes `in` and `out`. The voltage across the node pair `n10` and `n17` control the level of `gtest`. The level of `gtest` equals \((-2.314)\{v(n10) - v(n17)\}\). If this is a positive number, current flows in this direction: node `in`-`gtest`-node `out`.

```
G1 0 1 poly(1)
```

`G1` is a `g` element. It terminals are connected to nodes `0` and `1`. The level of `G1` is a polynomial in one variable. The one variable is the voltage across the node pair `n10` and `n17`. The level of `G1` is computed as

\[
g1 = 10^{-3} + 2\{v(n10) - v(n17)\}^2
\]

(6.33)
If this is a positive number, current flows in this direction: node 0–g1–node 1.

\[ G3 \quad 0 \quad 1 \quad \text{poly}(3) \]
\[ + \quad \text{nkingnkong} \]
\[ + \quad \text{npingnpong} \]
\[ + \quad \text{nsingnsong} \]
\[ + \quad 01 \quad 03 \quad 04 \]

\( g3 \) is a \( g \) element. Its terminals are connected to nodes 0 and 1. The level of \( g3 \) is a polynomial in three variables. The three variables are the voltages across the three node pairs \( \text{nking} \) and \( \text{nkong} \), \( \text{nping} \) and \( \text{npong} \), \( \text{nsing} \) and \( \text{nsong} \). The level of \( g3 \) is computed as

\[
g3 = \{ v(nking) - v(nkong) \} + 3 \{ v(nsing) - v(nsong) \} \\
+ 4 \{ v(nking) - v(nkong) \} \{ v(nping) - v(npong) \}
\]  

(6.34)

If this is a positive number, current flows in this direction: node 0–g3–node 3.

**Laplace Transforms**

For examples of Laplace transforms, please refer to the analogous examples of Laplace transforms in the voltage-controlled voltage source section (see “Voltage-Controlled Resistor (g VCR)” on page 217).

**Voltage-Controlled Resistor**

\( g\text{switch out 0 cur='v(out)*table(v(in), -1,1e-12, -0.1,1e-12, 0.1,1, 1,1)'} \)

The switch's resistance between nodes \( \text{out} \) and \( \text{ground} \) is controlled by the voltage at node \( \text{in} \). The switch is off when \( v(\text{in}) \) is less than -0.1 V, and on when \( v(\text{in}) \) is greater than 0.1 V. The interval \( -0.1V<v(\text{in})<0.1V \) serves as a smooth transition between the on and off states. Note the use of the table function to describe the conductance characteristics of the switch: the conductance is \( 10^{-12} \) (corresponding to a resistance of \( 10^{12} \)) when the switch is off, while the conductance is 1 when the switch is on. The chart below shows the switch's output current as a function of input voltage (holding the voltage at node out fixed at 1V).
Nonlinear Capacitor

The following example models a nonlinear capacitor whose capacitance is a function of applied voltage. A capacitor is described by specifying a charge function that depends on the voltage across the device. The device's capacitance is the derivative of the charge function with respect to the voltage. For example, a CMOS capacitor might be modeled as follows:

```
.param c0=10p vcc=1
gcmoscap n+ n- chg='c0*v(n+,n-) * (1 + 0.5*vcc*v(n+,n-))'
```

The capacitance of the device `gcmoscap` is then given by:

\[
C = c0 * (1 + vcc * v(n+,n-))
\]  

(6.35)

where \(c0\) represents the capacitance at zero applied voltage, and \(vcc\) measures the sensitivity of the capacitance with respect to input voltage.

The capacitor's charge depends on applied voltage, as shown in the chart below.

Voltage-Controlled Capacitor

A voltage-controlled capacitor is a two-terminal device whose capacitance is a function of node voltages elsewhere in the circuit. Such an element can be modeled in T-Spice with the expression-controlled g-element, using the `chg` parameter.

For example, the capacitance of a vertically moving parallel plate capacitor, a device used in the design of microelectromechanical systems (MEMS) might be modeled as follows:

```
gvccap n+ n- chg='v(n+,n-)*k/v(gapdistance)'
```

where `gapdistance` refers to a state variable which represents the distance between the capacitor's plates and \(k\) is a proportionality constant defined using `".param"` on page 131.
Switch-Level MOSFET

The T-Spice \texttt{stp()} and \texttt{table()} functions can be used to create a switch-level model for a MOSFET. An example of such a model for an N-type MOSFET is as follows:

\begin{verbatim}
gmos d s cur='v(d,s) * table(v(g,s)*stp(v(d,s)) + v(g,d)*(1-stp(v(d,s))), + 0,1e-12, 0.4,1e-12, 1,1e-7, 2,2e-5, + 3, 1e-4, 5, 4e-4)'
\end{verbatim}

Note that the use of the \texttt{stp()} function allows for a model that is symmetric with respect to source and drain.

Diode

The T-Spice \texttt{g}-element can be used to model any device for which analytic equations are available. When the equations have different forms for different regions of operation, the T-Spice \texttt{stp()} function can be useful. The following example models the current and capacitance of a diode. Note that the capacitance equation has different forms for the forward and reverse bias regions, but the \texttt{stp()} function allows us to describe the entire model using a single charge expression.

\begin{verbatim}
.param vt=0.02586 is=1e-14 tt=30n cjo=1e-12 vj=1 m=0.5
gdiode 1 2 cur='is*(exp(v(1,2)/vt)-1)'
+ chg='tt*is*(exp(v(1,2)/vt)-1)'
+ cjo*(2*vj*(1-sqrt(1-v(1,2)/vj))*stp(v(1,2)))
\end{verbatim}

This example would produce the following waveform for current:
and the following waveform for charge:

![Diode Charge Graph](image_url)
Voltage-Controlled Resistor (g VCR)

Voltage-controlled resistors can be modeled using a g element with the VCR keyword. Resistance value may be a linear, polynomial, or piece-wise linear function.

Syntax

Linear

\[ g \text{name} \text{node1 node2 VCR na1 nb1 K} \text{ [Options]} \]

Piecewise Linear

\[ g \text{name} \text{node1 node2 VCR PWL na1 nb1 v1 r1 v2 r2 ... vn rn} \text{ [Options]} \]

Polynomial

\[ g \text{name} \text{node1 node2 VCR POLY(N) na1 nb2 [na2 mb2 ...]} P_0 P_1 P_2 \ldots \text{ [Options]} \]

gname Voltage-controlled resistor name. Must begin with "g".
node1 Positive terminal. Positive current flows into node1.
node2 Negative terminal. Positive current flows out of node2.
K Transfactor — the ratio of the resistance to the control voltage.
PWL Keyword indicating that the VCR resistance is a piecewise linear function of the control voltages.
POLY Keyword indicating that the VCR resistance is a polynomial function of the control voltages.
N Number of control voltages (valid values 1-6).
v1 v2 Controlling voltage at corner 1, 2, and so on. (Unit: volts)
i1 i2 Output resistance at corner 1, 2, and so on. (Unit: ohms)
na1 nb2... Node pairs whose voltages control the level of the g element.
P0 P1 P2... Coefficients of the control polynomial.

Options

[M=value] [MAX=value] [MIN=value] [ABS =0 | 1] [TC1=value] [TC2=value] [SCALE=value]

M Multiplicity—the number of devices to be placed in parallel. (Default: 1) Can be a decimal or integer. Scales down through the subcircuit hierarchy.
MAX Maximum output voltage value.
MIN Minimum output voltage value.
ABS
Output is absolute value if ABS=1.

TC1, TC2,
First- and second-order temperature coefficients.

SCALE
Element value multiplier.

Linear Functions

The first form of VCR creates a linear resistor with a resistance that is equal to \( K \) multiplied by the voltage across the node pair \( na1 \ na2 \).

Pcwise Linear Functions

The next form of VCR creates a resistor that varies according to a voltage versus resistance PWL function table. The PWL table relates the controlling voltage across the node pair \( na1 \ na2 \) to the resistance of the \( node1 \ node2 \) branch.

The \( x \) values of the PWL function must be in increasing order.

Polynomial Functions

The third form creates a VCR whose resistance is a polynomial function of the voltages across one or more node pairs. Let

- \( x = \) voltage across node pair \( na1 \ na2 \);
- \( y = \) voltage across node pair \( na2 \ na3 \) (if \( N \geq 2 \));
- \( z = \) voltage across node pair \( na3 \ na4 \) (if \( N \geq 3 \)).

Then the resistance is defined as follows:

If \( N = 1 \):\n
\[
P_0 + P_1x + P_2x^2 + P_3x^3 + ... \quad (6.36)
\]

If \( N = 2 \):\n
\[
P_0 + P_1x + P_2y + P_3x^2 + P_4xy + P_5y^2 + P_6x^3 + P_7x^2y + P_8xy^2 + P_9y^3 + ... \quad (6.37)
\]

If \( N = 3 \):\n
\[
P_0 + P_1x + P_2y + P_3z + P_4x^2 + P_5xy + P_6xz + P_7y^2 + P_8yz + P_9z^2 + P_{10}x^3 + P_{11}x^2y + P_{12}xy^2 + P_{13}xz^2 + P_{14}xy^2 + P_{15}x^2z + P_{16}xyz + P_{17}x^3 + P_{18}x^2y + P_{19}xy^2 + P_{20}x^2z + P_{21}xyz + P_{22}x^3 + ... \quad (6.38)
\]

Examples

G1 in out VCR n10 n17 3.5

G1 is a resistor whose terminals are connected to nodes \textbf{in} and \textbf{out}. The voltage across the node pair \textbf{n10} and \textbf{n17} control the resistance of g1 according to \( 3.5(v(n10) - v(n17)) \).
gsw1 n1 n2 vcr pwl(1) nc1 nc2 0,1e10 3,1m

Gsw1 is effectively a switch element with a resistance which rapidly varies from 1e10 ohms to .001 ohms when the control voltage across the node pair nc1 and nc2 changes from 0 volts to 3 volts.
Voltage-Controlled Voltage Source (e)

A two-terminal ideal DC voltage supply with a level that is a function of one or more controlling voltages.

Syntax

Linear

\[
\text{ename node1 node2 na1 nb1 K [Options]}
\]

Polynomial

\[
\text{ename node1 node2 POLY(N) na1 nb1 [na2 nb2...] P_0 P_1 P_2 ... [Options]}
\]

LaPlace Transform

\[
\text{ename node1 node2 LAPLACE na1 nb1 a_1 a_2 ... a_m /[ / b_1 b_2 ... b_n] [Options]}
\]

Nonlinear | Behavioral

\[
\text{ename node1 node2 vol='expression' [Options]}
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>ename</em></td>
<td>Voltage-controlled voltage source name. Must begin with &quot;e&quot;.</td>
</tr>
<tr>
<td><em>node1</em></td>
<td>Positive terminal.</td>
</tr>
<tr>
<td><em>node2</em></td>
<td>Negative terminal.</td>
</tr>
<tr>
<td><em>K</em></td>
<td>Voltage gain—the ratio of the output voltage to the control voltage.</td>
</tr>
<tr>
<td><em>POLY</em></td>
<td>Keyword indicating that the output voltage is a polynomial function of the control voltages.</td>
</tr>
<tr>
<td><em>N</em></td>
<td>Number of control voltages (valid values 1-6).</td>
</tr>
<tr>
<td><em>LAPLACE</em></td>
<td>Keyword indicating that the output current is described via a Laplace transform function</td>
</tr>
<tr>
<td><em>vol</em></td>
<td>Keyword indicating that the source voltage is specified by an expression.</td>
</tr>
<tr>
<td><em>expression</em></td>
<td>Expression specified by the <em>vol</em> keyword.</td>
</tr>
<tr>
<td><em>na1 nb2...</em></td>
<td>Node pairs whose voltages control the level of the <em>e</em> element.</td>
</tr>
<tr>
<td><em>P_0 P_1 P_2...</em></td>
<td>Coefficients of the control polynomial.</td>
</tr>
<tr>
<td><em>a_0 a_1 a_2 ... a_m</em></td>
<td>Numerator of Laplace transfer function.</td>
</tr>
<tr>
<td><em>b_0 b_1 b_2 ... a_n</em></td>
<td>Denominator of Laplace transfer function.</td>
</tr>
</tbody>
</table>
Options

[MAX=value] [MIN=value] [ABS = [0 | 1]] [TC1=value] [TC2=value] [SCALE=value]

**MAX**
Maximum output voltage value.

**MIN**
Minimum output voltage value.

**ABS**
Output is absolute value if ABS=1.

**TC1, TC2,**
First- and second-order temperature coefficients.

**SCALE**
Element value multiplier.

**Linear Functions**

The first statement creates a voltage source with a level equal to $K$ multiplied by the voltage across node pair $na1 nb1$.

**Polynomial Functions**

The second statement creates a voltage source whose level is a nonlinear polynomial function of the voltages supplied by up to three voltage sources. Let

- $x = $ voltage across node pair $na1 nb1$;
- $y = $ voltage across node pair $na2 nb2$ (if $N \geq 2$);
- $z = $ voltage across node pair $na3 nb3$ (if $N \geq 3$).

Then the controlled voltage source’s level is defined as follows:

If $N = 1$:

$$P_0 + P_1x + P_2x^2 + P_3x^3 + ... \quad (6.39)$$

If $N = 2$:

$$P_0 + P_1x + P_2y + P_3x^2 + P_4xy + P_5y^2 + P_6x^3 + P_7x^2y + P_8xy^2 + P_9y^3 + ... \quad (6.40)$$

If $N = 3$:

$$P_0 + P_1x + P_2y + P_3z + P_4x^2 + P_5xy + P_6xz + P_7y^2 + P_8y^2z + P_9z^2 + P_{10}x^3 + P_{11}x^2y + P_{12}xy^2 + P_{13}xz^2 + P_{14}x^2y + P_{15}xyz + P_{16}x^3 + P_{17}y^3 + P_{18}y^2z + P_{19}z^3 + ... \quad (6.41)$$

If $N = 1$ and only one polynomial coefficient is specified, it is assumed to be $P1$, to facilitate the specification of linearly-controlled sources.

**Laplace Functions**

With the Laplace keyword, the current source is implemented as a Laplace transfer function. The three transform equations supported by the Laplace transform source types in T-Spice are shown in the...
equation below and several of the examples ("Integrator element" on page 223, "Differentiator element" on page 223, "Single Pole / Residue" on page 224).

\[
H(s) = \frac{a_0 + a_1 s + a_2 s^2 + \ldots + a_m s^m}{b_0 + b_1 s + b_2 s^2 + \ldots + b_n s^n}
\]

Expression-Controlled Functions

The fourth form of voltage controlled current sources uses mathematical expressions ("Expressions" on page 70) to define the output current and charge functions. At least one of the keywords cur or chg must be specified.

Examples

Etest in out n10 n17 -2.314

Etest is an e element. Its nodes are connected to nodes in and out. The voltage across the node pair n10 and n17 control the level of etest. The level of etest equals \((-2.314)\{v(n10) – v(n17)\} \)\n
E1 0 1 poly(1)
+ n10 n17
+ 1m 0 2

e1 is an e element. It terminals are connected to nodes 0 and 1. The level of e1 is a polynomial in one variable. The one variable is the voltage across the node pair n10 and n17. The level of e1 is computed as

\[
e1 = 10^{-3} + 2\{v(n10) – v(n17)\}^2
\] (6.43)

E3 0 1 poly(3)
+ nk1 nkong
+ nping npong
+ nsing nsong
+ 0 1 0 304

e3 is an e element. Its terminals are connected to nodes 0 and 1. The level of e3 is a polynomial in three variables. The three variables are the voltages across the three node pairs nk1 and nkong, nping and npong, nsing and nsong. The level of e3 is computed as

\[
e3 = \{v(nk1) – v(nkong)\} + 3\{v(nsing) – v(nsong)\} + 4\{v(nk1) – v(nkong)\}\{v(nping) – v(nping)\}
\] (6.44)

Ideal OpAmp

The expression-controlled e-element (voltage-controlled voltage source) can be used with the table function to model an ideal voltage amplifier. The following circuit element implements a voltage amplifier with a gain of 5, and minimum and maximum output voltages of -5V and 5V, respectively.

eamp out 0 vol='table(v(in), -1, -5, 1, 5)'
This example would produce the following waveform:

```
Integrator element

You can model an integrator using the e element with the Laplace transformation function.

Consider the behavior of an integrator. In the frequency domain, the integrator is modeled as:

\[ V_{out} = \frac{k}{s} V_{in} \]  (6.45)

And, in the time domain the integrator is modeled as:

\[ V_{out} = k \int V_{in} \, dt \]  (6.46)

The transfer function for the voltage gain is:

\[ H(s) = \frac{V_{out}}{V_{in}} = \frac{k}{s} \]  (6.47)

This is equivalent to the Laplace transfer function with the coefficient assignments:
\[ a_0 = k, a_1=0, b_0=0, b_1=1 \]

The voltage-controlled voltage source element which implements this function is:

```
einteg out in laplace cpos cneg k 0.0 / 0.0 1.0
```

**Differentiator element**

You can also use the e element with the Laplace transformation function to model a voltage differentiator.
Consider the behavior of a differentiator. In the frequency domain, the differentiator is defined by:

\[ V_{out} = ksV_{in} \quad (6.48) \]

And, in the time domain the differentiator is defined by:

\[ V_{out} = \frac{dV_{in}}{dt} \quad (6.49) \]

The transfer function for the voltage gain is:

\[ H(s) = \frac{V_{out}}{V_{in}} = ks \quad (6.50) \]

This is equivalent to the Laplace transfer function with the coefficient assignments:

\[ a_0 = 0, \ a_1=k, \ b_0=1 \]

The voltage-controlled voltage source element to implement this function is:

\texttt{ediff out in laplace cpos cneg 0.0 k / 1.0}

**Single Pole / Residue**

A single pole/residue element is modeled using a transfer function which is:

\[ H(s) = \frac{a_0}{b_0 + b_1s} \quad (6.51) \]

A representative single pole element is:

\texttt{epole out in laplace cpos cneg 1.0 / 0.5 3.0}

**Zero-Delay Inverter Gate**

The same technique used to produce the ideal amplifier, above, could be used to create a simple model of a zero-delay inverter:

\texttt{einvert out 0 vol='table(v(in), 0.6, 5, 1, 4.9, 1.5, 0.1, 1.9, 0)'}
This example would produce the following waveform:

![Inverter Waveform](image)

**Zero-Delay AND Gate**

The following example implements a simple model for a zero-delay digital AND gate:

```
eand out 0 vol='table( min(v(a),v(b)), 0, 0, 1, 0.5, 4, 4.5, 5, 5)'
```

This example would produce the following waveform:

![AND Gate Waveform](image)

Other logic gates can also be modeled using the same technique:

- **NAND gate:**
  ```
enand nand 0 vol='table( min(v(a),v(b)), 0, 5, 1, 4.5, 4, 0.5, 5, 0)'
```

- **OR gate:**
  ```
eor or 0 vol='table( max(v(a),v(b)), 0, 0, 1, 0.5, 4, 4.5, 5, 5)'
```

- **NOR gate:**
Voltage-Controlled Oscillator (VCO)

A voltage-controlled oscillator model can be built using a combination of expression-controlled sources. The following example implements a VCO whose output voltage is a sine wave of frequency $f_0+k_0(v_{control} - v_c)$, where $v_{control}$ is the control voltage, and $f_0$, $k_0$, and $v_c$ are fixed parameters. The amplitude and offset of the VCO's output sine wave are controlled by the parameters $amp$ and $offset$, respectively.

```
.param offset=2.5 amp=2.5 pi=3.141592654
.param f0=10k k0=3k vc=2.5
evco out 0 vol='offset+amp*sin(2*pi*(f0*time()+k0*v(theta)))'
gtheta theta 0 cur='vc-v(control)' chg='v(theta)'
.ic v(theta)=0
```

Note that a state variable called $theta$ has been introduced to track the phase of the VCO. Without this additional state variable, the VCO might have been modeled using:

```
evco2 out 0 vol='offset+amp*sin(2*pi*(k0*(v(control)-vc)+f0)*time()))'
```

but this would result in phase discontinuities of the VCO's output sine wave. The initial condition for $theta$ is necessary to define a DC value for the phase, which would otherwise be arbitrary. The $gtheta$ element implements the equation

$$\frac{d\theta}{dt} = v(control) - v_c$$  \hspace{1cm} (6.52)

The chart below shows the behavior of a VCO modeled as in the five SPICE lines above. The VCO's frequency ranges from 2.5kHz when the control voltage is zero to 17.5kHz when the control voltage is 5V.
7 Simulation Options

This section provides a reference to the T-Spice simulation options that can be set with the ".options" (page 125) command.

Options are grouped in the following categories:

- “Accuracy and Convergence Options” (page 228)
- “Timestep and Integration Options” (page 262)
- “Model Evaluation Options” (page 282)
- “Linear Solver Options” (page 303)
- “General Options” (page 307)
- “Output Options” (page 320)
- “Probing Options” (page 348)
Accuracy and Convergence Options

“absi | abstol” (page 229)
“accurate” (page 231)
“bypass” (page 233)
“cshunt” (page 235)
“dcmethod” (page 238)
“extraiter[ations] | newtol” (page 240)
“gmin” (page 242)
“gramp” (page 244)
“kcitest” (page 246)
“maxdcbfailures” (page 248)
“minsrcstep” (page 250)
“numndset” (page 252)
“numnx | itl2” (page 254)
“precise” (page 256)
“reli | reltol” (page 259)
“vmax” (page 261)

“absv | vntol” (page 230)
“autoconverge” (page 232)
“bytol” (page 234)
“dchomotopy” (page 236)
“dctest” (page 239)
“fast” (page 241)
“gmindc” (page 243)
“gshunt” (page 245)
“kvtest” (page 247)
“mindratio” (page 249)
“numnd | itl1” (page 251)
“numns | itl6” (page 253)
“numnxramp” (page 255)
“recycleOP” (page 258)
“relv” (page 260)
absi | abstol

```plaintext
.options absi = absi
```

where \( \text{absi} > 0 \)

**Default Value**

\[ 1 \times 10^{-10} \text{ A} \]

**Description**

Specifies a convergence criterion that limits the total RMS (root mean square) of residual branch currents at all nodes in the circuit. The current test for convergence is applied when `kcltest = true`.

When the current tolerance test is applied, the system of equations is determined to be converged if:

\[
\text{reli} > \sqrt{\sum_{i=0}^{n} (\Delta I')^2} \quad \text{and} \quad \text{absi} > \sqrt{\sum_{i=0}^{n} I^2}
\]

where \( I \) is the residual branch current at each node. The quantity \( \Delta I' \) represents the change in relative residual branch current between two consecutive iterations:

\[
\Delta I' = \left( \frac{I_j}{i_{\text{max}j}} \right) - \left( \frac{I_{j-1}}{i_{\text{max}j-1}} \right)
\]

where \( i_{\text{max}j} \) is the largest branch current (in absolute value) flowing into the node in question in the \( j \)th Newton iteration.

**See Also**

“reli | reltol” (page 259), “kcltest” (page 246)
absv | vntol

.options absv = absv

where absv > 0

Default Value

$1 \times 10^{-6}$ V

Description

Specifies a convergence criterion limiting the absolute change in node voltage between two Newton iterations. The voltage test for convergence is applied when \texttt{kvltest} = \texttt{true}.

The voltage tolerance at each node is calculated as follows:

$voltage \ tolerance = \max(\texttt{absv, relv} \times V),$

where $V$ is the node voltage value. If the voltage variation at each node is less than the calculated voltage tolerance for that node, then the iteration is considered to be converged.

See Also

“\texttt{relv}” (page 260), “\texttt{kvltest}” (page 247)
accurate

.options accurate = { true | false }

Default Value

false

Description

Triggers changes to other option settings to maximize simulation accuracy:

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Accurate</th>
</tr>
</thead>
<tbody>
<tr>
<td>“lvltim” (page 267)</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>“numnd</td>
<td>itl1” (page 251)</td>
<td>250</td>
</tr>
<tr>
<td>“numndset” (page 252)</td>
<td>numnd / 10 (25)</td>
<td>numnd / 5 (100)</td>
</tr>
<tr>
<td>“numns</td>
<td>itl6” (page 253)</td>
<td>50</td>
</tr>
<tr>
<td>“numnx</td>
<td>itl2” (page 254)</td>
<td>100</td>
</tr>
<tr>
<td>“numnxramp” (page 255)</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>“reldv</td>
<td>relvar” (page 277)</td>
<td>0.35</td>
</tr>
<tr>
<td>“reli</td>
<td>reltol” (page 259)</td>
<td>5 × 10^{-4}</td>
</tr>
<tr>
<td>“rmax” (page 279)</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Specifying any of the above fields individually will override the value set by accurate.

See Also

“fast” (page 241), “precise” (page 256)
**autoconverge**

```plaintext
.options autoconverge = { true | false }
```

**Default Value**

`true`

**Description**

The autoconverge option enables a number of simulator and model evaluation features that can improve DC convergence.

The autoconverge methods for improving convergence include:

- Selectively applying a conductance across terminals of capacitors that are bounded by floating nodes (see `dcstep` (page 239)).
- After the Newton solver has achieved convergence, it will continue to iterate as long as the residuals are being reduced, thus increasing the accuracy of the solution.
- The `Imelt` test for high device currents will be enabled with a limit of 1 amp for those devices that support it.
- Resistors will be limited to a minimum of 1 milliOhm (see `minresistance | resmin` (page 294)).
bypass

.options bypass = { true | false }

Default Value

true

Description

Enables or disables the diode and transistor evaluation bypass algorithm. If the terminal voltage values for a device have a relative change that is less than or equal to bytol since the previous evaluation, then this evaluation will be skipped, and the previous solutions used.

See Also

“bytol" (page 234)
bytol

 opciones bytol = bytol  \text{ where } \textit{bytol} \geq 0

Default Value

1.0e-9

Description

Sets the relative tolerance for the bypass algorithm terminal voltage values.

See Also

“\textit{bypass}” (page 233)
cshunt

$options cshunt = cshunt$  where $cshunt \geq 0.0$ F

Default Value

0.0 F

Description

Adds a capacitor with the specified capacitance from each node to ground. A small $cshunt$ value will sometimes resolve transient analysis “timestep too small” values that are caused by small, high-frequency oscillations within the circuit.

See Also

“gmin” (page 242), “gshunt” (page 245)
dchomotopy

.options dchomotopy = { none | source | gmin | pseudo | all }

Default Value

all

Description

Specifies the algorithm used to correct DC operating point non-convergences. When a non-convergence occurs during DC operating point analysis, the selected form of homotopy will be used to try to obtain a valid, converged solution.

Options include:

- **none**: Do not attempt any homotopy methods.
- **source**: Source stepping. All voltage and current sources are ramped up from zero to their final values. The smallest source step that T-Spice will take is controlled by "minsrcstep" (page 250).
- **gmin**: Gmin stepping. T-Spice uses the $g_{min}$ stepping algorithm to find the minimum conductance value that yields a convergent solution. The options $gmindc$ and $gramp$ specify a search range for the minimum required conductance, $g_{min}$:

  \[ \text{gmindc} \leq g_{min} \leq \text{gmindc} \times 10^{gramp} \]

- **multirate**: Multirate source-stepping. Multirate is similar to the regular source-stepping, except that each source is ramped up independently, one by one, rather than all together uniformly.
- **pseudo**: Pseudotransient solution. In the pseudotransient solution method, T-Spice uses homotopy methods to approximate a solution, then removes the homotopies for the final solution. T-Spice obtains a pseudotransient solution as follows:
  - T-Spice first enables pseudotransient simulation values for $gmindc$ and $cshunt$, which are determined internally. The source values are then ramped up to their final values.
  - T-Spice then performs a time-stepping simulation. When this simulation converges, T-Spice removes the homotopy devices ($gmindc$ and $cshunt$), one at a time, until the final solution is reached.
If a non-convergence occurs, T-Spice attempts homotopy methods in the following order:

- standard damped Newton solver
- gmin-stepping
- source-stepping
- multirate source-stepping
- pseudo-transient solution

As soon as a converged solution is achieved, the simulation completes without attempting the next homotopy algorithm.

**Note:**

If you know that a specific homotopy is required to reach a convergent solution, you can reduce simulation time by setting this as the default method with `dcmethod`. Using `dcmethod` automatically skips the standard solution (no homotopy) and only attempts to calculate the solution using the method specified.

**See Also**

- “`dcmethod`” (page 238), “`minsrcstep`” (page 250), “`gmindc`” (page 243), “`gramp`” (page 244)
Chapter 7: Simulation Options

Accuracy and Convergence Options

**dcmethod**

```
.options dcmethod = {standard | source | gmin | pseudo }
```

**Default Value**

*standard*

**Description**

Specifies the default method for solving a DC operating point problem. This option is useful when you have prior knowledge that the circuit can only reach a convergent solution when a particular homotopy is required. In this case, you can reduce simulation time by setting the `dcmethod` option to the appropriate method, thus skipping attempts to solve the problem using either the standard method or the other homotopy methods. The settings for `dcmethod` are:

- **standard**: Default setting. If a non-convergence is reached using the standard method, T-Spice will then apply the homotopy methods specified by `dchomotopy` to try to reach a convergent solution.
- **source**: Source stepping. All voltage and current sources are ramped up from zero to their final values. The smallest source step that T-Spice will take is controlled by the `minsrcstep` option.
- **gmin**: Gmin stepping. T-Spice finds the minimum conductance value that yields a convergent solution. The options `gmindc` and `gramp` specify a search range for the minimum required conductance, $g_{min}$:

  $$ gmindc \leq g_{min} \leq (gmindc \times 10^{gramp}) $$

- **pseudo**: Pseudotransient solution. In the pseudotransient solution method, T-Spice uses homotopy methods to approximate a solution, then removes the homotopies for the final solution. T-Spice obtains a pseudotransient solution as follows:
  - T-Spice first enables pseudotransient simulation values for `gmindc` and `cshunt`, which are determined internally. The source values are then ramped up to their final values.
  - T-Spice then performs a time-stepping simulation. When this simulation converges, T-Spice removes the homotopy devices (`gmindc` and `cshunt`), one at a time, until the final solution is reached.

**Note:**

If you do not know the best solution method for your circuit, then do not set `dcmethod`. Instead, set `dchomotopy` to the default value of `all`. In this case, T-Spice will automatically cycle through the homotopies as necessary to achieve convergence.

**See Also**

“`dcstep`” (page 239), “`minsrcstep`” (page 250), “`gmindc`” (page 243), “`gramp`” (page 244)
**dcstep**

```
.options dcstep = dcstep
```

where \( dcstep \geq 0 \)

**Default Value**

0.0

**Description**

Adds a conductance across the terminals of each capacitor during DC operating point computation. If a non-zero value is specified, T-Spice computes the additional conductance for each capacitor as:

\[
g = \frac{c}{dcstep}
\]

where \( g \) is the applied conductance and \( c \) is the device capacitance.

**See Also**

“cshunt” (page 235), “gmindc” (page 243)
extraiter[ations] | newtol

```
.options extraiter = extraiter
```

where `extraiter` is a non-negative integer

Default Value

0

Description

Instructs T-Spice to compute the specified number of Newton solver iterative steps after convergence criteria have been met. This option is used to improve the accuracy of the solution, and is applicable to DC operating point, DC sweep, and AC simulations. For transient analysis, use the `trnewtol` option.

When `precise = true`, the default value of `extraiter` is 10.

See Also

“precise” (page 256), “trextraiter[ations] | trnewtol” (page 280)
fast

.options fast = { true | false }

Default Value

false

Description

Triggers changes to other options settings to maximum simulation speed:

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Fast</th>
</tr>
</thead>
<tbody>
<tr>
<td>“absi</td>
<td>abstol” (page 229)</td>
<td>$1 \times 10^{-10}$ A</td>
</tr>
<tr>
<td>“bytol” (page 234)</td>
<td>0</td>
<td>$1 \times 10^{-13}$</td>
</tr>
<tr>
<td>“reldv</td>
<td>relvar” (page 277)</td>
<td>0.35</td>
</tr>
<tr>
<td>“reli</td>
<td>reltol” (page 259)</td>
<td>$5 \times 10^{-4}$</td>
</tr>
<tr>
<td>“relq</td>
<td>relchgtol” (page 278)</td>
<td>reli ($5 \times 10^{-4}$)</td>
</tr>
</tbody>
</table>

Specifying any of the above fields individually will override the value set by fast.

See Also

“accurate” (page 231), “precise” (page 256)
gmin

.options gmin = gmin  where gmin ≥ 0

Default Value

$1 \times 10^{-12} \Omega^{-1}$

Description

Specifies a conductance added in parallel with all $pn$ junctions during transient analysis. When $gmin > 0$, T-Spice generates a $gmin$ conductance across two terminals by adding a resistor with resistance $R$

$$R = \frac{1}{gmin}$$

T-Spice applies the $gmin$ conductance to various elements as follows:

- diode—conductance is added across the positive/negative terminals.
- —conductance is added across the base/emitter and the base/collector terminals.
- MOSFET—conductance is added across the source/bulk, drain/bulk, and the source/drain terminals.
- MESFET—conductance is added across the source/gate, drain/gate, and source/drain terminals.
- JFET—conductance is added across the source/gate, drain/gate, and source/drain terminals.

See Also

“gmindc” (page 243), “gshunt” (page 245)
gmindc

```
.options gmindc = gmindc

where \( gmindc \geq 0 \)
```

**Default Value**

\[ 1 \times 10^{-12} \Omega^{-1} \]

**Description**

Specifies a conductance that is added in parallel with all \( pn \) junctions during DC operating point analysis. When \( gmindc > 0 \), T-Spice generates a \( gmindc \) conductance across two terminals by adding a resistor with resistance \( R \)

\[ R = \frac{1}{gmindc} \]

T-Spice applies the \( gmindc \) conductance to various elements as follows:

- diode—conductance is added across the positive/negative terminals.
- —conductance is added across the base/Emitter and the base/collector terminals.
- MOSFET—conductance is added across the source/bulk, drain/bulk, and the source/drain terminals.
- MESFET—conductance is added across the source/gate, drain/gate, and source/drain terminals.
- JFET—conductance is added across the source/gate, drain/gate, and source/drain terminals.

**Note:** When a DC operating point non-convergence occurs, T-Spice can begin a \( g_{min} \) stepping algorithm to find the minimum conductance that yields a convergent solution. The \( g_{min} \) stepping algorithm is triggered when a non-convergence occurs and the `dchomotopy` option is set to `all` or `gmin`.

**See Also**

“gramp” (page 244), “dchomotopy” (page 236)
gramp

```
.options gramp = gramp
```

where \( 0 < gramp < (-\log_{10} (100 \times gmin\!\!c)) \)

Default Value

4

Description

Specifies the range over which the \( gmin\!\!c \) value will be swept in \( gmin \) stepping for DC analysis. The \( gmin \) stepping algorithm is triggered when a non-convergence occurs and dchomotopy is set to all or \( gmin \). Together, the options \( gmin\!\!c \) and \( gramp \) specify a search range for the minimum required conductance, \( gmin \):

\[
gmin\!\!c \leq gmin \leq gmin\!\!c \times 10^{gramp}
\]

T-Spice’s \( gmin \) stepping algorithm searches the specified conductance range in two steps. First, T-Spice performs a binary search between \( gmin\!\!c \) and \( gmin\!\!c \times 10^{gramp} \). T-Spice searches for the smallest value of \( gmin \) that results in a converged solution. T-Spice automatically ends the binary search when it reaches a \( \Delta gmin \) that is less than or equal to a factor of 10.

Starting with binary search results, T-Spice then begins reducing the value of \( gmin \) by a factor of 10 in each iteration. Once a non-convergence occurs, the previous convergent iteration provides the final solution.

See Also

“\( gmin\!\!c \)” (page 243)
gshunt

.options gshunt = gshunt where gshunt ≥ 0

Default Value

0.0 Ω⁻¹

Description

Specifies a conductance to be added between every node and ground. When gshunt > 0, T-Spice generates a gshunt conductance from each node to ground by adding a resistor with resistance \( R \)

\[
R = \frac{1}{gshunt}
\]

See Also

“cshunt” (page 235), “gmin” (page 242)
kcltest

.options kcltest = {true | false }

Default Value
true

Description
Enables the current tolerance test for convergence. When the current tolerance test is applied, the system of equations is determined to be converged if:

\[
reli > \sqrt{\sum_{i=0}^{n} (\Delta I')^2} \quad \text{and} \quad \text{absi} > \sqrt{\sum_{i=0}^{n} I^2}
\]

where \(I\) is the residual branch current at each node. The quantity \(\Delta I'\) represents the change in relative residual branch current between two consecutive iterations:

\[
\Delta I' = \left( \frac{I_j}{imax_j} \right) - \left( \frac{I_{j-1}}{imax_{j-1}} \right)
\]

where \(imax_j\) is the largest branch current (in absolute value) flowing into the node in question in the \(j^{th}\) Newton iteration.

See Also
“absi | abstol” (page 229), “reli | reltol” (page 259), “kvtest” (page 247)
**kvltest**

```
.options kvltest = { true | false }
```

**Default Value**

`false`

**Description**

Enables the voltage tolerance test for convergence during transient analysis. The voltage tolerance test is always performed during DC, DC sweep, Transfer, and AC analysis. The `kvltest` option is used for enabling this test during transient analysis also.

The voltage tolerance is calculated as follows:

\[
\text{voltage tolerance} = \max(\text{absv}, \text{relv} \times V),
\]

where \( V \) is the node voltage value. If the voltage variation at each node is less than the calculated voltage tolerance for that node, then the iteration is considered to be converged.

**See Also**

“\text{absv \mid vntol}” (page 230), “\text{relv}” (page 260), “\text{kcltest}” (page 246)
maxdcfailures

.options maxdcfailures = n

where \( n \) is a non-negative integer

Default Value

4

Description

Maximum number of non-convergence failures allowed in a DC sweep simulation before T-Spice ends processing with a “too many nonconvergences” error.

See Also

“mindcratio” (page 249), “numnx | itl2” (page 254)
mindcratio

```
.options mindcratio = mindcratio
```

where \( 0 < \text{mindcratio} < 1 \)

Default Value

\( 1 \times 10^{-4} \)

Description

Minimum fractional step size allowed in source stepping for DC sweep analysis:

\[
\Delta d_{\text{cmin}} = \text{mindcratio} \times \Delta d
\]

where \( \Delta d \) is the step size specified in the netlist `sweep` statement. If the step size falls below \( \Delta d_{\text{cmin}} \), T-Spice will declare a non-convergence error.

T-Spice applies source stepping when a fixed source step fails to converge. In source stepping, the source variable is gradually ramped up from the previous sweep value to the next sweep value.

See Also

“\text{maxdcfailures}” (page 248), “\text{numnx | itl2}” (page 254)
Chapter 7: Simulation Options

Accuracy and Convergence Options

minsrcstep

.options minsrcstep = minsrcstep where minsrcstep > 0

Default Value

$1 \times 10^{-8}$

Description

Minimum fractional step size for source stepping:

$min \text{ step size} = minsrcstep \times (Source \text{ value})$

In source stepping, all voltage and current sources are ramped up from zero to their final values. This allows T-Spice to find the DC operating points of difficult-to-converge circuits. Source stepping is used only in non-converging cases of initial DC operating point computations when dchomotopy is set to source or all.

See Also

“numns | itl6” (page 253), “dchomotopy” (page 236)
numnd | itl1

.options numnd = numnd  

where numnd is a positive integer

Default Value

1000

Description

Newton iteration limit for DC operating point computation. If a solution does not converge within numnd iterations, T-Spice applies the homotopy method specified by dchomotopy to attempt to reach a convergent solution. If dchomotopy = none, T-Spice declares a non-convergence error.

See Also

“dchomotopy” (page 236)
numndset

```
.options numndset = numndset
```

where `numndset` is a positive integer

**Default Value**

```
numnd / 10
```

**Description**

This option is used during DC operating point computations when the user has specified node voltage guesses using the `.nodeset` command. `Numndset` is the number of Newton iterations during which the `.nodeset` nodes will be held at their user-specified voltage values. After `numndset` iterations, or when the circuit convergence criteria have been met, these node voltages are allowed to vary for the remainder of the computation.

**See Also**

```
".nodeset" (page 119)
```
numns | itl6

.options numns = numns

where numns is a positive integer

Default Value

50

Description

Newton iteration limit for each source stepping attempt in DC operating point analysis. In source stepping, all voltage and current sources are ramped up from zero to their final values. This allows T-Spice to find the DC operating points of difficult-to-converge circuits. Source stepping is used only in non-converging cases of initial DC operating point computations when dchomotopy is set to source or all.

See Also

“numnd | itl1” (page 251), “dchomotopy” (page 236)
numnx | itl2

.. code-block::

   .options numnx = numnx

where numnx is a positive integer

Default Value

100

Description

Newton iteration limit for DC sweep computation. If a convergent solution is not reached within numnx iterations, T-Spice begins source stepping (or "ramping"). In source stepping, the source variable is gradually ramped up from the previous sweep value to the next sweep value. The Newton iteration limit for source ramping is specified by numnxramp.

See Also

“numnxramp” (page 255)
numnxramp

.options numnxramp = numnxramp where numnxramp is a positive integer

Default Value

50

Description

Newton iteration limit for DC sweep computation during source ramping (or "stepping"). T-Spice applies source ramping when a fixed source step fails to converge within numnx iterations. In source ramping, the source variable is gradually ramped up from the previous sweep value to the next sweep value.

See Also

“numnx | itl2” (page 254), “minsrcstep” (page 250)
precise

.options precise = {true | false}

Default Value

false

Description

Triggers changes to other options for extreme simulation precision. This option should only be used for single transistor characterizations, or for very simple circuits.

The following option settings will be used:

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>Precise</th>
</tr>
</thead>
<tbody>
<tr>
<td>“absdv</td>
<td>absvar” (page 263)</td>
<td>0.5 V</td>
</tr>
<tr>
<td>“absi</td>
<td>abstol” (page 229)</td>
<td>$1 \times 10^{-10}$ A</td>
</tr>
<tr>
<td>“absq</td>
<td>chgtol</td>
<td>chargetol” (page 264)</td>
</tr>
<tr>
<td>“absv</td>
<td>vntol” (page 230)</td>
<td>$1 \times 10^{-6}$ V</td>
</tr>
<tr>
<td>“extrater[ations]</td>
<td>newtol” (page 240)</td>
<td>0</td>
</tr>
<tr>
<td>“gmin” (page 242)</td>
<td>$1 \times 10^{-12}$ Ω$^{-1}$</td>
<td>$1 \times 10^{-14}$ Ω$^{-1}$</td>
</tr>
<tr>
<td>“gmin dc” (page 243)</td>
<td>$1 \times 10^{-12}$ Ω$^{-1}$</td>
<td>$1 \times 10^{-14}$ Ω$^{-1}$</td>
</tr>
<tr>
<td>“kvltest” (page 247)</td>
<td>false</td>
<td>true</td>
</tr>
<tr>
<td>“lvltim” (page 267)</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>“extrater[ations]</td>
<td>newtol” (page 240)</td>
<td>0</td>
</tr>
<tr>
<td>“numnd</td>
<td>itl1” (page 251)</td>
<td>250</td>
</tr>
<tr>
<td>“numndset” (page 252)</td>
<td>$numnd / 10$ (25)</td>
<td>$numnd / 5$ (100)</td>
</tr>
<tr>
<td>“numns</td>
<td>itl6” (page 253)</td>
<td>50</td>
</tr>
<tr>
<td>“numnx</td>
<td>itl2” (page 254)</td>
<td>100</td>
</tr>
<tr>
<td>“numnxramp” (page 255)</td>
<td>50</td>
<td>100</td>
</tr>
<tr>
<td>“pivtol” (page 305)</td>
<td>$1 \times 10^{-14}$</td>
<td>$1 \times 10^{-16}$</td>
</tr>
<tr>
<td>“reldv</td>
<td>relvar” (page 277)</td>
<td>0.35</td>
</tr>
<tr>
<td>“reli</td>
<td>reltol” (page 259)</td>
<td>$5 \times 10^{-4}$</td>
</tr>
<tr>
<td>“relq</td>
<td>relchgtol” (page 278)</td>
<td>$reli (5 \times 10^{-4})$</td>
</tr>
<tr>
<td>“relv” (page 260)</td>
<td>$1 \times 10^{-3}$</td>
<td>$1 \times 10^{-4}$</td>
</tr>
<tr>
<td>“rmax” (page 279)</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Specifying any of the above fields individually will override the value set by `precise`.

**See Also**

- “accurate” (page 231), “fast” (page 241)
recycleOP

\texttt{.options recycleop = \{true | false\}}

**Default Value**

\texttt{true}

**Description**

Controls whether previously computed DC Operating Point node voltages are used as the initial guess for subsequent DC OP calculations. Alternatively, each DC OP calculation will start from scratch, and may be computationally time consuming.

The default behavior is generally preferred because it results in much faster OP calculations. The only reason to disable recycleOP would be to ensure that all Operating Point calculations consistently yield the same result regardless of simulation sweep order.
reli | reltol

\[ \text{Default Value} \]
\[ 5 \times 10^{-4} \]

\[ \text{Description} \]
Specifies a convergence criterion limiting the relative change in total RMS branch current for all nodes in the circuit between consecutive iterations. The current test for convergence is applied when \text{kcltest = true}.

When the current tolerance test is applied, the system of equations is determined to be converged if:

\[ \text{reli} > \frac{\sum_{i=0}^{n} (\Delta I')^2}{\sum_{i=0}^{n} I^2} \quad \text{and} \quad \text{absi} > \frac{\sum_{i=0}^{n} I^2}{\sum_{i=0}^{n} \Delta I'} \]

where \( I \) is the residual branch current at each node. The quantity \( \Delta I' \) represents the change in relative residual branch current between two consecutive iterations:

\[ \Delta I' = \left( \frac{I_j}{\text{imax}_j} \right) - \left( \frac{I_{j-1}}{\text{imax}_{j-1}} \right) \]

where \( \text{imax}_j \) is the largest branch current (in absolute value) flowing into the node in question in the \( j \)th Newton iteration.

\[ \text{See Also} \]
“absi | abstol” (page 229), “kcltest” (page 246)
**relv**

```
.options relv = relv
```

where \( relv > 0 \)

**Default Value**

\( 1 \times 10^{-3} \)

**Description**

Specifies a convergence criterion limiting the relative change in node voltage at any node in the circuit between consecutive iterations. The voltage test for convergence is applied when \( \text{kvltest} = \text{true} \).

The voltage tolerance test uses both the \( \text{absv} \) and \( \text{relv} \) options to calculate tolerance:

\[
\text{voltage tolerance} = \max (\text{absv}, \text{relv} \times V),
\]

where \( V \) is node voltage. If the voltage change at each node is less than the voltage tolerance for that node, then the iteration is considered to be converged.

**See Also**

“\text{absv | vntol}” (page 230), “\text{kvltest}” (page 247)
**vmax**

```plaintext
.options vmax = vmax

where vmax >= 0
```

**Default Value**

The greater of 1000 Volts or the maximum voltage source value squared.

**Description**

The maximum voltage absolute value permitted during DC operating point computations. If any voltages exceed this value during Newton iterations, then a convergence error results.

Set VMAX to 0 in order to disable this voltage limit test.
Timestep and Integration Options

“absdv | absvar” (page 263)
“autopowerup” (page 265)
“lvltim” (page 267)
“method” (page 271)
“mu | xmu” (page 273)
“numntreduce | itl3” (page 275)
“reldv | relvar” (page 277)
“rmax” (page 279)
“trtol” (page 281)

“absq | chgtol | chargeol” (page 264)
“ft” (page 266)
“maxord” (page 270)
“mintimeratio | rmin” (page 272)
“numnt | itl4 | imax” (page 274)
“poweruplen” (page 276)
“relq | relchgtol” (page 278)
“trextraiterations | trnewtol” (page 280)
absdv | absvar

.options absdv = absdv

Default Value

0.5 V

Description

For transient analysis, absdv specifies the threshold absolute voltage change between two consecutive time steps. This quantity is used with reldv to calculate the voltage variance error measurement:

\[
\text{variance} = \left| \frac{V_{n+1} - V_n}{\text{reldv} \cdot \max(\text{absdv}, V_n)} \right|
\]

Voltage variance is used to scale time step sizes when “lvltim” (page 267) is equal to 1, 3, or 4.

See Also

“lvltim” (page 267)
absq | chgtol | chargetol

 opciones \texttt{absq = absq} \quad \text{where \ absq > 0}

Default Value

\[ 1 \times 10^{-14} \text{ C} \]

Description

Minimum capacitor charge or inductor flux used to predict a timestep in the Local Truncation Error (LTE) algorithm (\texttt{lvltim} = 2 or 4). The \texttt{absq} option sets a floor on charge values to prevent the timestep size from becoming too small.

The value of \texttt{absq} is used to calculate Local Truncation Error (LTE) as follows:

\[ LTE = \frac{Q - Q_{\text{predicted}}}{\text{trtol} \times \max(\text{absq}, \ (\text{relq} \times Q))} \]

When \texttt{lvltim}=2 or 4, and \(LTE > 1\), T-Spice recalculates the solution at a smaller timestep. See “\texttt{lvltim}” (page 267) for a description of the LTE algorithm.

See Also

“\texttt{lvltim}” (page 267), “\texttt{relq} | \texttt{relchgtol}” (page 278)
**autopowerup**

```plaintext
.options autopowerup = { true | false }
```

**Default Value**

```
true
```

**Description**

When there is a non-convergence failure for a transient analysis DC operating point, automatically perform a powerup simulation. See `.tran` (page 165) / `powerup` (page 166).
\textbf{ft}

\texttt{.options ft = ft} \hspace{1cm} \text{where } 0 < \text{ft} < 1

\textbf{Default Value}

0.25

\textbf{Description}

Fraction by which the internal timestep is decreased if a transient analysis solution does not converge within \texttt{numnt} iterations. T-Spice recalculates the solution for a smaller timestep:

\[ \Delta t_n = \Delta t_n' (1-\text{ft}) \]

where \( \Delta t_n' \) is the size of the failed or non-converged \( n \)th timestep.

The \texttt{ft} option also determines the fraction by which the next timestep (\( \Delta t_{n+1} \)) is decreased if the \( n \)th timestep solution requires more than \texttt{numntreduce} iterations to converge:

\[ \Delta t_{n+1} = \Delta t_n (1-\text{ft}) \]

For more information about timestep reduction algorithms, see \texttt{“lvltim”} (page 267).

\textbf{See Also}

\texttt{“numnt | itl4 | imax”} (page 274), \texttt{“numntreduce | itl3”} (page 275), \texttt{“lvltim”} (page 267)
lvltim

.options lvltim = {1 | 2 | 3 | 4}

Default Value

1

Description

Specifies the algorithm used to control timestep sizes in transient analysis simulation:

<table>
<thead>
<tr>
<th>lvltim</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Iteration count algorithm with voltage variance test.</td>
</tr>
<tr>
<td>2</td>
<td>Local Truncation Error timestep control algorithm.</td>
</tr>
<tr>
<td>3</td>
<td>Modified iteration count with voltage variance test and timestep reversal.</td>
</tr>
<tr>
<td>4</td>
<td>Combines lvltim=1 voltage variance test, prior to the timestep, with lvltim=2 local truncation error control of timestep reversal.</td>
</tr>
</tbody>
</table>

Iteration Count with Voltage Variance Test (lvltim = 1, 3, or 4)

All three algorithms require that the solution at each time step converge within numnt iterations. If a convergent solution is not found within numnt iterations, T-Spice recalculates the solution at a smaller timestep:

$$
\Delta t_n' = \Delta t_n \times (1 - ft)
$$

where $\Delta t_n$ is the original size of the $n^{th}$ timestep. When lvltim = 1, no further conditions are placed on the current timestep solution.

When lvltim=3, the timestep solution must also have an error value less than or equal to 1, where error is calculated as the voltage variance between time steps

$$
error = \left| \frac{V_{n+1} - V_n}{\max(absv, (reldv \cdot V_n))} \right|
$$

If lvltim=3 and the error is greater than 1, T-Spice recalculates the solution for a smaller timestep. The smaller timestep is obtained by scaling the current timestep by the error (voltage variance):

$$
\Delta t_n' = \Delta t_n \times \left( \frac{0.9}{error} \right)
$$

T-Spice continues the timestep reversal algorithm until it reaches a convergent solution within numnt iterations, such that $error < 1$. 
**Local Truncation Error Algorithm (lvltim = 2 or 4)**

The Local Truncation Error (LTE) algorithm adjusts the timestep size according to the discretization error generated by integration. The amount of truncation error introduced by integration increases with the rate-of-change in the circuit. Local truncation error is calculated as the ratio between the error in predicted charge and the charge tolerances `relq` and `absq`. The value of “trtol” (page 281) is included as a corrective factor in the LTE calculation:

$$\text{error} = \frac{Q - Q_{\text{predicted}}}{\text{trtol} \cdot \max((\text{relq} \cdot Q), \text{absq})}$$

If the calculated local truncation error (LTE) is greater than 1, then T-Spice recalculates the solution for a smaller time step:

$$\Delta t' = \Delta t \cdot \left(\frac{0.9}{\text{error}}\right)$$

**Note:** The LTE algorithm is error-prone in high-current devices, because the rapidly changing charge values will generate large local truncation error values. This can cause T-Spice to use extremely small timesteps, leading to slow simulations and/or “timestep too small” errors. In high-current circuits, a voltage-based timestep algorithm (lvltim = 1 or 3) is often the preferred choice.

**Determining the Next Timestep (lvltim = 1, 2, 3, or 4)**

All three algorithms use both iteration count and an error measurement to determine the size of the next timestep. After a convergent solution is found at the current timestep, T-Spice applies the following rules to determine the next timestep size:

- If the solution required more than numntreduce iterations, T-Spice reduces the next timestep by the fraction ft:
  $$\Delta t_{n+1} = \Delta t_n \cdot (1 - \text{ft})$$

- If the solution converged in fewer than numntreduce iterations, T-Spice uses the appropriate error measurement to scale the next timestep. (Error is equal to voltage variance when lvltim = 1 or 3, and LTE when lvltim = 2 or 4.)
  $$\Delta t_{n+1} = \Delta t_n \cdot \left(\frac{0.9}{\text{error}}\right)$$
The following chart summarizes the three timestep control algorithms:

1. **time\_n = time\_n-1 + \Delta t\_n**
   - \(lvltim=1, 3, \text{ or } 4\)
   - \(lvltim=2\)

   Calculate the predicted voltage at time\_n, then calculate the predicted voltage variance (error) from time\_n-1.

   - **predicted error > 1**
     - Decrease \(\Delta t\)
       - \(\Delta t\_n = 0.9(\Delta t\'_n/\text{error})\)
   
   - **predicted error < 1**
     - Newton-Raphson Solver
       - Converged in numnt iterations?
         - **no**
           - timestep reversal
             - \(\Delta t\_n = \Delta t\'_n(1-ft)\)
         - **yes**
           - \(lvltim=2, 3, \text{ or } 4\)
             - **error > 1**
               - timestep reversal
                 - \(\Delta t\_n = 0.9(\Delta t\'_n/\text{error})\)
             - **error < 1**
               - TIMESTEP CONVERGED
                 - \(\Delta t\_n+1 = \Delta t\_n(1-ft)\)
               
   - iterations > numntreduce
     - \(\Delta t\_n+1 = \Delta t\_n(1-ft)\)
   
   - iterations < numntreduce
     - \(\Delta t\_n+1 = 0.9(\Delta t\'_n/\text{error})\)

---

**See Also**

maxord

.options maxord = {1 | 2 | 3 | 4 }

Default Value

2

Description

Maximum time integration order for variable-order Gear’s BDF calculation in transient analysis. Gear’s BDF integration is used when method = gear.

See Also

“method” (page 271)
method

.options method = { gear | trap }

Default Value

trap

Description

Method of numerical integration for estimating the time derivative of the system’s charge components during a .tran simulation. Possible settings are:

**gear**

Variable order Gear’s backward differential formula (see “Gear’s BDF Method” on page 35). The order of integration is controlled by the option maxord.

**trap**

Trapezoidal integration. This method is faster than gear but may introduce non-physical oscillations in nodal responses. (see “Trapezoidal Integration Method” on page 35).

See Also

“maxord” (page 270)
mintimeratio | rmin

.options mintimeratio = mintimeratio  where mintimeratio > 0

Default Value

$1 \times 10^{-9}$

Description

Relative minimum timestep size for transient simulations. The minimum timestep for transient simulation is equal to $\text{mintimeratio} \times tstep$, where $tstep$ is the timestep size listed on the .tran statement.

If the timestep size falls below $\text{mintimeratio} \times tstep$, T-Spice ends processing with a "timestep too small" error.
mu | xmu

.options mu = mu

where $0 \leq \mu \leq 0.5$

Default Value

0.5

Description

Coefficient for varying the integration between the backward Euler formula and the Trapezoidal formula. Valid values of $\mu$ are between 0 and 0.5, with 0 yielding a backward Euler integration, 0.5 yielding Trapezoidal integration, and intermediate values producing a hybrid integration of the specified proportional weighting. Trapezoidal integration is used when $\text{method} = \text{trap}$.

See Also

“method” (page 271)
numnt | itl4 | imax

.options numnt = numnt

where numnt is a positive integer

Default Value

10

Description

Newton iteration limit for transient analysis solutions.

If a solution does not converge within numnt iterations, T-Spice recalculates the solution for a smaller time step. The fraction by which the time step is decreased after a non-convergence is specified by “ft” (page 266).

See Also

“ft” (page 266)
numntreduce | itl3

.options numntreduce = numntreduce

where numntreduce is a positive integer

Default Value

3

Description

For transient analysis, the threshold number of Newton iterations that controls the next time step size.

If more than numntreduce iterations are needed to reach a convergent solution at the nth time step, then the next time step size is reduced by the fraction ft:

\[ \Delta t_{n+1} = \Delta t_n (1 - \textit{ft}) \]

If the nth time step converges with fewer than numntreduce iterations, then T-Spice increases or decreases the next time step by scaling with a calculated error value:

\[ \Delta t_{n+1} = \Delta t_n \cdot \left( \frac{0.9}{\text{ERROR}} \right) \]

The error calculation is dependent on the algorithm specified by “lvltim” (page 267).

See Also

“ft” (page 266), “lvltim” (page 267)
poweruplen

```
.options poweruplen = poweruplen
```

**Default Value**

0.1% of the total transient simulation time.

**Description**

Length of the powerup ramp (seconds) during powerup transient analysis.

**See Also**

“`.tran`” (page 165)
**reldv | relvar**

```
.options reldv = reldv
```

where \( reldv > 1 \)

**Default Value**

0.35

**Description**

For transient analysis, \( reldv \) specifies the maximum relative voltage change between two consecutive time steps. This quantity is used with \( absdv \) to calculate the voltage variance error measurement:

\[
\text{variance} = \left| \frac{V_{n+1} - V_n}{reldv \cdot \max(absdv, V_n)} \right|
\]

Voltage variance is used to scale time step sizes when “lvltim” (page 267) is equal to 1, 3, or 4.

**See Also**

“absdv | absvar” (page 263), “lvltim” (page 267)
relq | relchgtol

```
.options relq = relq
```

where \( \text{relq} > 0 \)

**Default Value**

\[ 5 \times 10^{-4} \]

**Description**

Maximum relative error in predicted capacitor charge or inductor flux. The error between predicted and actual charges is used to adjust timestep sizes in the Local Truncation Error timestep algorithm (\( \text{lvltim} = 2 \) or 4).

The value of \( \text{relq} \) is used to calculate Local Truncation Error (LTE) as follows:

\[
LTE = \frac{Q - Q_{\text{predicted}}}{\text{trtol} \times \max(\text{absq}, (\text{relq} \times Q))}
\]

When \( \text{lvltim} = 2 \) or 4, and \( LTE > 1 \), T-Spice recalculates the solution at a smaller timestep. See “\( \text{lvltim} \)” (page 267) for a description of the LTE algorithm.

**See Also**

“\( \text{lvltim} \)” (page 267), “\( \text{absq} \) | \( \text{chgtol} \) | \( \text{chargetol} \)” (page 264)
rmax

.options rmax = rmax  

where $rmax \geq 1$

Default Value

2

Description

Defines the maximum allowed timestep in transient simulation, specified as a ratio. The maximum timestep is defined as:

$$\Delta t_{max} = rmax \times \Delta t_{tran}$$

where $\Delta t_{tran}$ is the timestep specified in the netlist .tran statement.

See Also

“fast” (page 241), “accurate” (page 231), “precise” (page 256)
trextraiter[ations] | trnewtol

.options trextraiter = trextraiter \quad \text{where } trextraiter \geq 0

Default Value

0

Description

Instructs T-Spice to compute the specified number of Newton solver iterative steps after convergence criteria have been met. This option is used to improve the accuracy of the solution, and is applicable to transient analysis only. For increasing the iterations of non-transient simulations use the newtol option.

When \textit{precise} = \text{true}, the default value of \textit{trextraiter} is 1.

See Also

“extraiter[ations] | newtol” (page 240), “precise” (page 256)
trtol

.options trtol = trtol

Default Value

10

Description

Corrective factor for estimation of the local truncation error in the LTE algorithm:

\[
LTE = \frac{Q - Q_{\text{predicted}}}{\text{trtol} \cdot \max((\text{relq} \cdot Q), \text{absq})}
\]

See “lvlim” (page 267) for a description of the LTE algorithm.

Note: The default value of trtol has been calculated to minimize error in the estimation of LTE. Changing the value of trtol is generally not recommended.

See Also

“absq | chgtol | chargetol” (page 264), “relq | relchgtol” (page 278), “trtol” (page 281)
Model Evaluation Options

“dcap” (page 283)  
“defad” (page 285)  
“defl” (page 287)  
“defnrs” (page 289)  
“defps” (page 291)  
“deriv” (page 293)  
“modmonte” (page 295)  
“mout” (page 297)  
“scale” (page 299)  
“tnom” (page 301)  

“dccap” (page 284)  
“defas” (page 286)  
“defnrd” (page 288)  
“defpd” (page 290)  
“defw” (page 292)  
“minresistance | resmin” (page 294)  
“moscap” (page 296)  
“paramchk” (page 298)  
“scalm” (page 300)  
“wl” (page 302)


**dcap**

```
.options dcap = {1 | 2 }
```

**Default Value**

2

**Description**

Selects the model used to calculate depletion capacitance for BJTs, diodes, and MOS parasitic diodes. Possible settings are:

- `dcap = 1` Berkeley SPICE diode equations.
- `dcap = 2` Revised equations. See the device model documentation for “BJT Level 1 (Gummel-Poon)” (page 373) and “Diode” (page 403).

**See Also**

“BJT Level 1 (Gummel-Poon)” (page 373), “Diode” (page 403)
dccap

.options dccap = {true | false}

Default Value

false

Description

Controls the computation of charges and capacitances in DC simulations. Charge and capacitance values are not normally computed for DC operating point, DC sweep, and DC transfer analyses. Enabling this flag will permit the user to print out charge values for capacitors, diode, and transistors, without affecting the DC solutions.
defad

\[ .\text{options defad = defad} \quad \text{where defad} \geq 0.0 \]

**Default Value**

0.0 m²

**Description**

Default MOSFET drain diode area.

**See Also**

“MOSFET (m)” (page 193)
defas

.options defas = defas

where \( \text{defas} \geq 0.0 \)

Default Value

0.0 m^2

Description

Default MOSFET source diode area.

See Also

“MOSFET (m)” (page 193)
defl

.options defl = defl where defl ≥ 0.0

Default Value

1e-4 m

Description

Default MOSFET channel length.

See Also

“MOSFET (m)” (page 193)
defnrd

```
.options defnrd = defnrd

where defnrd \geq 0.0
```

Default Value

0.0

Description

Default number of diffusion squares for a MOSFET drain resistor.

See Also

“MOSFET (m)” (page 193)
defnrs

\[ \text{.options defnrs = defnrs} \quad \text{where defnrs} \geq 0.0 \]

**Default Value**

0.0

**Description**

Default number of diffusion squares for a MOSFET source resistor.

**See Also**

“MOSFET (m)” (page 193)
**defpd**

```
.options defpd = defpd
```

where `defpd` ≥ 0.0

**Default Value**

0.0 m

**Description**

Default MOSFET drain diode perimeter.

**See Also**

“MOSFET (m)” (page 193)
defps

.options defps = defps

where defps ≥ 0.0

Default Value

0.0 m

Description

Default MOSFET source diode perimeter.

See Also

“MOSFET (m)” (page 193)
**defw**

```
.options defw = defw
where defw × 0.0
```

**Default Value**

1e-4 m

**Description**

Default MOSFET channel width.

**See Also**

“MOSFET (m)” (page 193)
deriv

.options deriv = { 0 | 1 }

Default Value

0

Description

Selects the default method for computing all device $dq/dv$ and $di/dv$ derivatives:

0  Computes analytical derivatives where possible. (Available for most MOSFETs, diodes, and BJTs.)
1  Only uses numerical, finite-difference derivatives.
minresistance | resmin

```
.options minresistance = minresistance
```

Default Value

1e-5 Ω

Description

Minimum resistance value for all resistors, including parasitics and inductor values, as well as resistors defined in the netlist. Any resistance that is specified or computed to be less than `minresistance` is reset to `minresistance`. 
modmonte

\[ \text{.options modmonte} = \{0 \mid 1\} \]

Default Value

0

Description

Controls how model parameters are evaluated when they involve Monte Carlo variables.

If \text{modmonte} is 0, then lot variations of model parameters are performed, and each device that references the model will have the same model definition.

If \text{modmonte} is 1, then local (mismatch) variations of model parameters are performed, and each device that references the model will have a unique set of model parameters derived from a re-evaluation of the Monte Carlo random variable(s). A new model is created for each device, and the model name will be the original model name with \#number appended (e.g. nch\#1, nch\#2, ...).

Example

\textbf{Example 1:}
In this example of mismatch analysis, each Mosfet mn1-mn3 will have a unique threshold voltage value due to the dVthN Monte Carlo parameter variations

\begin{verbatim}
.option modmonte=1
.param dVthN = agauss(0.0, 0.07, 3)
.model NCH nmos level=49 VTH0='0.3703728+dVthN'
MN1 1 11 21 0 NCH W=2.5u L=.25u
MN2 2 12 22 0 NCH W=2.5u L=.25u
MN3 3 13 32 0 NCH W=2.5u L=.25u
\end{verbatim}

\textbf{Example 2:}
In this example of process variation analysis, the 3 transistors mn1-mn3 will share the one NCH Mosfet model definition and will have equivalent performance for each Monte Carlo variation.

\begin{verbatim}
.option modmonte=0
.param dVthN = agauss(0.0, 0.07, 3)
.model NCH nmos level=49 VTH0='0.3703728+dVthN'
MN1 1 11 21 0 NCH W=2.5u L=.25u
MN2 2 12 22 0 NCH W=2.5u L=.25u
MN3 3 13 32 0 NCH W=2.5u L=.25u
\end{verbatim}

See Also

“Monte Carlo Parameters” (page 132)
moscap

.options moscap = (true | false)

Default Value

false

Description

Enables automatic source/drain area/perimeter estimation for MOSFETs.

When the ACM model parameter is set to 0 (default) or 10, then:
   The default AD and AS values will be \( l \times w \)
   The default PD and PS values will be \( 2 \times (l+w) \)

See Also

“MOSFET (m)” (page 193)
mout

.options mout = { 0 | 1 }

Default Value

0

Description

Selects whether output values should be scaled for all M parallel devices or should be single device values. This option affects AC small-signal parameter listings and terminal current, charge, and device detail plot outputs. Actual M scaling is unaffected but listings and plots will display as either one device (mout=1) or the sum of all M parallel devices (mout=0).

Note that this option is not implemented for the Philips SiMKit models and devices.

Example

mnmos1 d g s b nmosmodel l=1u w=.15u M=4
.options mout=0

In this example, the AC small-signal listings and plots will be scaled to include the multiplier effects of all four mnmos1 parallel devices.
paramchk

.options paramchk = { 0 | 1 }

Default Value

0

Description

Selects the default value of the paramchk model parameter that is used by all BSIM models. Paramchk determines whether model parameter error checking and correcting is active.
scale

.spec scale = scale

where \( scale \geq 0.0 \)

Default Value

1.0

Description

Scales the physical dimensions of capacitors, MESFETs, MOSFETs, and resistors. Lengths and widths are multiplied by \( scale \), and areas are multiplied by \( scale^2 \).

See Also

scalm

.options scalm = scalm

where scalm ≥ 0.0

Default Value

1.0

Description

Default scaling factor for resistors and capacitors. The scalm option is overridden by setting the model parameter scale. The scaling factor affects lengths, widths, and drawn lengths and widths.

See Also

“scale” (page 299), “Capacitor” (page 400), “Resistor” (page 498)
**tnom**

`.options tnom = tnom`

**Default Value**

25 °C

**Description**

Nominal temperature in °C. This is the temperature at which all device and model parameters are assumed to be measured.

**Note:** Differences in the nominal and the simulation temperatures may account for some variations in solutions when using different SPICE simulators. Many Berkeley SPICE derivatives use 27 °C as the baseline temperature.
wl

.options wl = { true | false }

Default Value

false

Description

Reverses MOSFET length and width specifications. If \texttt{wl} = \texttt{true}, then length specifications apply to width, and width specifications apply to length.

See Also

“\texttt{defl}” (page 287), “\texttt{defw}” (page 292)
Linear Solver Options

“linearSolver” (page 304)    “pivtol” (page 305)
“zpivtol” (page 306)
linear solver

.options linear solver = { best | klu | sparse }

Default Value

best

Description

Specifies how linear equations are solved. Possible settings are:

- **best**: Allows T-Spice to select a solver based on the number of independent nodes in the system (sparse for less than 100, klu otherwise).
- **klu**: A sparse solver for circuit simulators, developed by Tim Davis and the University of Florida. This is the default solver for systems with more than 100 independent nodes (equation unknowns).
- **sparse**: The original Berkeley SPICE direct solver for a system with fewer than 100 independent nodes (equation unknowns).
pivtol

.options pivtol = pivtol where pivtol > 0.0

Default Value

$1 \times 10^{-14}$

Description

Minimum pivoting tolerance for real matrices.
zpivtol

.options zpivtol = zpivtol

where zpivtol > 0.0

Default Value

1 × 10^{-6}

Description

Minimum pivoting tolerance for complex matrices.

“tnom” (page 301)
Chapter 7: Simulation Options

General Options

“autostop” (page 308)
“compatibility | spicemode” (page 310)
“macmod” (page 312)
“persist” (page 315)
“spice” (page 317)
“threads” (page 319)

“casesensitive” (page 309)
“conncheck” (page 311)
“parhier” (page 313)
“search” (page 316)
“str” (page 318)
autostop

.options autostop = { true | false }

Default Value

false

Description

Instructs T-Spice to terminate transient analysis after all “.macro .eom” (page 107) results have been found. The autostop option does not affect transient analyses run in preview mode.

See Also

“.macro .eom” (page 107), “.tran” (page 165)
casesensitive

.options casesensitive = { true | false }

Default Value

false

Description

Controls case sensitivity for names of models, subcircuits, library sections, parameters, and nodes.

Note: This option only controls case sensitivity for names that appear after the .options casesensitive command in the input file.
compatibility | spicemod

.options compatibility = { spice | hspice | pspice }

Default Value

hspice

Description

Selects a compatibility mode for the input netlist syntax and for simulation settings.

T-Spice can natively read about 99% of Berkeley SPICE, HSPICE, and PSpICE input statements. There are, however, some incompatibilities and some irreconcilable syntax distinctions across the various simulators. By assigning a simulator compatibility mode, these input syntax ambiguities can be resolved and appropriately parsed.

In addition to providing syntax parsing support, the compatibility command sets certain simulator settings in order to mimic the behavior of the target simulator. The most notable of these settings is the default temperature, which is 25 °C in HSPICE, and 27 °C in other simulators. Other differences, such as the default MOSFET nrd and nrs values, and output format settings, may sometimes be the source of subtle or surprising differences in simulator solutions.

HSPICE compatibility mode settings:

- HSPICE compatibility mode is the default setting. Therefore, the default T-Spice settings are used.

SPICE compatibility mode settings:

- The command for assigning Berkeley SPICE compatibility, .options compat=spice, is equivalent to the .options spice command, described in “spice” (page 317).

PSPICE compatibility mode settings:

- The default temperature and tnom setting are 27 °C
- The “acout” (page 322) option is set to 0
- The pwr() function is evaluated as a signed power function in expressions
**conncheck**

```plaintext
.options conncheck = { true | false }
```

**Default Value**

true

**Description**

Toggles connectivity checking, which tests for common connectivity problems such as "No dc path to ground from node X" and "Node X is attached to only one device."
macmod

.options macmod = { 0 | 1 | 2 | 3 }

Default Value

0 (disabled)

Description

Controls the automatic replacement of subcircuit instances for missing MOSFET model instances, and conversely the replacement of MOSFET model instances for missing subcircuit definition instances.

When MACMOD is 1, if any M* MOSFET instance refers to a model name that is not found, a subcircuit will be instanced if the subcircuit definition name matches the requested MOSFET model name and the number of pins is the same.

When MACMOD is 2, if any X* subcircuit instance refers to a subcircuit definition name that is not found, a MOSFET will be instanced if the model definition name matches the requested subcircuit definition name and the number of pins is the same.

When MACMOD is 3, both MACMOD=1 and MACMOD=2 types of substitutions are performed. Missing MOSFET model references will get equivalent subcircuit definition instances. Missing subcircuit references will get equivalent MOSFET model instances.

When MACMOD is 0, no substitutions are performed.
parhier

.options parhier={local | global}

Default value

global

Description

Establishes the scoping algorithm for selection of parameter values in a hierarchical design. Parameters can obtain their values in a number of different methods, and the varying methods must have an established precedence for resolving which value assignment to select when a parameter value is assigned in multiple ways or at multiple levels of hierarchy. The ways in which parameters can be set are:

- .param statements
  example:.param cap=600f

- subckt parameter declaration, which establish default values
  example:.subckt inverter in out v cap=800f

- subckt instances (calls) with parameter assignment
  example:xinverter n1 n2 vdd inverter cap=200f

- .step and DC sweep simulations
  example:.step cap 200f 1000f 200f

The parhier option controls whether global .param statements will override the local (subcircuit) parameter assignments. With global scoping the highest level (outermost) parameter assignment will be selected. With local scoping the lowest level (innermost) parameter assignment will be selected. The rules of scoping vary according to the parhier setting, as depicted in the following chart:

<table>
<thead>
<tr>
<th>parhier=local</th>
<th>parhier=global</th>
</tr>
</thead>
<tbody>
<tr>
<td>sweep assignments</td>
<td>sweep assignments</td>
</tr>
<tr>
<td>subckt call assignment</td>
<td>.param assignment</td>
</tr>
<tr>
<td>subckt declaration assignment</td>
<td>subckt call assignment</td>
</tr>
<tr>
<td>.param assignment</td>
<td>subckt declaration assignment</td>
</tr>
</tbody>
</table>

The following example demonstrates how the parhier setting can effect the final parameter value.

.param cap=600f
.subckt inverter in out v cap=800f
...
.cap1 n3 gnd C=cap
.ends
xinverter1 1 5 vdd cap=400f

If parhier is set to local, then xinverter.c1 will have a capacitance of 400f. If parhier is set to global, then xinverter.c1 will have a capacitance of 600f.
**Note:** To get a listing of all parameter names, values, and the type of assignment, use the `xref` option:

```
.options xref=true
```
**persist**

```
.options persist = { 0 | 1 | 2 | 3 }
```

**Default Value**

2

**Description**

Instructs T-Spice to continue simulation when the specified levels of warnings or errors are generated. In the default mode, only fatal errors encountered during execution cause the simulation to exit with a terminal error message.

0 Stop simulation when a warning or error occurs.
1 Ignore warning messages.
2 Ignore error messages.

**Note:** Fatal error messages cannot be ignored; these always cause the simulation to exit.
search

.options search = pathname

where *pathname* specifies a valid directory path. If the path contains spaces, *pathname* must be enclosed in quotes.

Default Value

*None*

Description

Sets the search path for libraries and include files. T-Spice uses the search path as follows:

- If T-Spice encounters an undefined subcircuit, it automatically searches *pathname* for a file named *subckt.inc*, where *subckt* is the name of the subcircuit.
- If *include* or *lib* statements reference files that are not in the current directory, T-Spice automatically looks for these files in the *pathname* directory.

See Also

“.if ... / .elseif ... / .else / .endif” (page 101), “.lib” (page 104)
spice

.options spice = { true | false }

Default Value

false

Description

Changes other option settings to be compatible with Berkeley SPICE:

<table>
<thead>
<tr>
<th>Option</th>
<th>Default</th>
<th>spice</th>
</tr>
</thead>
<tbody>
<tr>
<td>“acout” (page 322)</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>“dcap” (page 283)</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>“defnrd” (page 288)</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>“defnrs” (page 289)</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>“ingold” (page 329)</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>“tnom” (page 301)</td>
<td>25 °C</td>
<td>27 °C</td>
</tr>
</tbody>
</table>

Specifying any of the above fields individually will override the value set by spice.
str

.options str = { true | false }

Default Value

false

Description

When true, input string parameters are required to be presented as str("...") rather than a simple quoted string "...".

Note that expressions should always be enclosed in single quotes, e.g. '2.4*del1*lvt'

Example

X1 a b VerilogModule filename=str("test.dat")
threads

.options threads = { 0 | 1 }

Default Value

0 on single processor computers
1 on multiprocessor and multicore computers.

Description

Controls parallel processing of model evaluation in T-Spice simulation. When the value is set to 1, T-Spice will automatically decompose the workload into small tasks, and dynamically distribute these tasks to multiple threads for processing. The size of the tasks is controlled via internal variables, and is outside of user control.

0 Disables threading. This is the default on single processor computers.
1 Enables threading using a number of threads equal to the number of processor cores. This is the default on multiprocessor and multicore computers.

n Enable threading using \( n \) threads, where \( n \) is a number greater than 1. It is not recommended to use a value of \( n \) which is greater than the number of processor cores available on the computer.

See Also

“Multi-Threaded Processing” on page 40
Output Options

“acct” (page 321)
“brief” (page 323)
“csv” (page 325)
“echo” (page 327)
“ingold” (page 329)
“maxmsg” (page 331)
“measinfo” (page 333)
“nomod” (page 336)
“outputall” (page 339)
“prtdel” (page 343)
“probelvl” (page 341)
“prtinterp” (page 344)
“verbose” (page 346)
“acout” (page 322)
“captab” (page 324)
“dnout” (page 326)
“expert” (page 328)
“list” (page 330)
“measform” (page 332)
“node” (page 335)
“numdgt” (page 337)
“opts” (page 338)
“pathnum” (page 340)
“probetop” (page 342)
“statdelay” (page 345)
“xref” (page 347)
acct

.options acct = { true | false }

Default Value

false

Description

Tracks and reports simulation iteration counts and other accounting statistics. This information is written to the Simulation Window, and is also recorded in the output file.
acout

.options acout = { 0 | 1 }

Default Value

1

Description

Calculation method for AC magnitude or phase differences requested in .print and .probe statements (e.g., vm(x,y)).

- If acout = 0, T-Spice performs subtraction first, then calculates the magnitude or phase of the difference, e.g., vm(x,y) = vm(x-y).
- If acout = 1, T-Spice first calculates the magnitudes or phases and then takes the difference, e.g., vm(x,y) = vm(x) - vm(y).

Note: Use acout = 0 for compatibility with Berkeley SPICE, and acout = 1 for compatibility with H-Spice.

See Also

“.print” (page 137), “.probe” (page 152)
brief

.options brief = { true | false }

Default Value

false

Description

Turns off most of the printout which is sent to the Simulation Status window.

The brief option is equivalent to the verbose=0 setting.
captab

\[ \text{options captab} = \{ \text{true} | \text{false} \} \]

Default Value

false

Description

Lists the capacitances for each node in the netlist, and identifies the node with the greatest capacitance.
CSV

.options csv = { 0 | 1 | 2}

Default Value

0

Description

Causes T-Spice to generate an additional CSV (Comma-Separated Values) output file for importing the T-Spice solutions into third-party tools, such as spreadsheets.

- If csv = 0, then the CSV file will not be written.
- If csv = 1, then all specific plot items will be written to the CSV file. e.g. ".print v(out)" will cause one trace, v(out), to be written to the .csv file.
- If csv = 2, then all .print and .probe outputs will be written to the CSV file.

Note that no .csv file is created if there are no traces to print.
dnout

.options dnout = { 0 | 1 }

Default Value

0

Description

Selects the units that T-Spice uses to measure all input and output noise spectral density magnitudes.

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>T-Spice reports noise spectral density contributions in units of Volts/sqrt(Hz).</td>
</tr>
<tr>
<td>1</td>
<td>T-Spice reports noise spectral density contributions in units of Volts²/Hz.</td>
</tr>
</tbody>
</table>

Note: Use dnout = 1 for compatibility with H-Spice computations.
**echo**

`.options echo = { true | false }`

**Default Value**

false

**Description**

Causes T-Spice to print each line of input to the error log as it is read. The error log is the T-Spice GUI output window, or the specified file when the `-e filename` commandline option is used.

When `echo = true`, T-Spice lists each input next to the line number on which it occurs. For example:

```plaintext
Initializing parser with command line options
line 00001: .options verbose=2
End-of-input
line 00003: .param pres=100
line 00004: .param ptc1=0.1
line 00005: .param ptc2=0.4
line 00006:
line 00007: .options precise
line 00008:
line 00009: v1 1 0 0.01 sin(0.01 12 1e8)
...```
expert

.options expert = { true | false }

Default Value

false

Description

Activates the printout of detailed information about non-convergent nodes and devices when a simulation, or a stage of a simulation, fails.
**ingold**

```
.options ingold = { 0 | 1 | 2 }
```

**Default Value**

0

**Description**

Controls the format of all real numeric data which is written to the AC Small-Signal output, see “`.acmodel` (page 79), to `.measure` command output, see “`.measure` (page 109), and to device listings.

<table>
<thead>
<tr>
<th><code>ingold</code> value</th>
<th>Format</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>engineering format</td>
<td>-2.875u</td>
</tr>
<tr>
<td>1</td>
<td>g format - combined fixed and exponential format (scientific notation)</td>
<td>6.234</td>
</tr>
<tr>
<td>2</td>
<td>e format - constant width exponential format</td>
<td>-1.7428e-005</td>
</tr>
</tbody>
</table>

For the engineering format, exponents between the value of 1e-18 and 1e15 are expressed as a single character appended to the end of the real data. Numbers which are smaller then 1e-18 or greater than 1e15 are printed in the exponential format. The characters used to express the exponential value are:

<table>
<thead>
<tr>
<th>Character suffix</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>$10^{12}$</td>
</tr>
<tr>
<td>G</td>
<td>$10^{9}$</td>
</tr>
<tr>
<td>X</td>
<td>$10^{6}$</td>
</tr>
<tr>
<td>K</td>
<td>$10^{3}$</td>
</tr>
<tr>
<td>m</td>
<td>$10^{-3}$</td>
</tr>
<tr>
<td>u</td>
<td>$10^{-6}$</td>
</tr>
<tr>
<td>n</td>
<td>$10^{-9}$</td>
</tr>
<tr>
<td>p</td>
<td>$10^{-12}$</td>
</tr>
<tr>
<td>f</td>
<td>$10^{-15}$</td>
</tr>
<tr>
<td>a</td>
<td>$10^{-18}$</td>
</tr>
</tbody>
</table>
**list**

.options list = { true | false }

**Default Value**

false

**Description**

Directs T-Spice to printout detailed information about every element in the netlist. The information will include nodal connectivity, the device values (resistance, capacitance, etc.), and other pertinent device parameter and device geometry settings.
maxmsg

.options maxmsg = maxmsg  

where maxmsg is a non-negative integer

Default Value

4

Description

Sets the maximum number of times that a duplicate warning message will be printed. A value of 0 specifies that all warning messages should be printed an unlimited number of times.
measform

.options measform = { 0 | 1 }

Default Value

0

Description

Controls the format of the .measure results file.

measform=0 (default) is for historic file format with tabular listings of sweep results.

measform=1 generates one block of output per inner sweep.
**measinfo**

```
.options measinfo = { 0 | 1 }
```

**Default Value**

0

**Description**

Controls what outputs are generated for the `.measure` command. 0 is for minimal output (measured value only). 1 is for added outputs: at, when, from, to, etc. intermediate values.

**Example**

Measurement result summary using option 0:

```
vta_mb0 = 2.5650e+000
vta_mb8 = 2.7810e+000
vta_mb15 = 2.9700e+000
bias_mb0_ub0 = 4.3388e+000
bias_mb0_ub1 = 4.3353e+000
bias_mb0_ub2 = 4.3319e+000
```

Measurement result summary using option 1:

```
vta_mb0 = 2.5649 @ 20.0000m
vta_mb8 = 2.7811 @ 38.0000m
vta_mb15 = 2.9701 @ 56.0000m
bias_mb0_ub0 = 4.3387 @ 20.0000m
bias_mb0_ub1 = 4.3353 @ 22.0000m
bias_mb0_ub2 = 4.3320 @ 24.0000m
```


**monteinfo**

\[ \text{.options monteinfo = \{ 0 | 1 | 2\}} \]

**Default Value**

1

**Description**

Controls what Monte Carlo results are written to the results files and thus are available to plot in W-Edit.

- 0: Print final summary information.
- 1: Print final summary information, measurements and random variable values for each step.
- 2: Print final summary information and print or probe results for each step.
node

.options node = { true | false }

Default Value

false

Description

Prints a node cross-reference table listing each node and all the elements connected to it. Each element is listed as **element:term**, where *term* identifies the element terminal as follows.

<table>
<thead>
<tr>
<th>Device Type</th>
<th>Terminal identifiers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diode</td>
<td>+ = anode</td>
</tr>
<tr>
<td></td>
<td>− = diode</td>
</tr>
<tr>
<td></td>
<td>B = base</td>
</tr>
<tr>
<td></td>
<td>C = collector</td>
</tr>
<tr>
<td></td>
<td>E = emitter</td>
</tr>
<tr>
<td></td>
<td>S = substrate</td>
</tr>
<tr>
<td>MOSFET or JFET</td>
<td>B = bulk</td>
</tr>
<tr>
<td></td>
<td>D = drain</td>
</tr>
<tr>
<td></td>
<td>S = source</td>
</tr>
<tr>
<td></td>
<td>G = gate</td>
</tr>
<tr>
<td>All other devices</td>
<td>1 = terminal 1</td>
</tr>
<tr>
<td></td>
<td>2 = terminal 2</td>
</tr>
<tr>
<td></td>
<td>etc.</td>
</tr>
</tbody>
</table>
nomod

.options nomod = { true | false }

Default Value

true

Description

 Suppresses printout of all model parameters.
**numdgt**

```
.options numdgt = numdgt
```

where `numdgt` is a non-negative integer

**Default Value**

4

**Description**

Minimum number of digits after the decimal point to be printed to the output file for each requested (.print) output value.
opts

.options opts = { true | false }

Default Value

false

Description

Prints the current settings of all control options. The default behavior (opts = false) is to list only those option values that have been changed.
outputall

.options outputall = { true | false }

Default Value

false

Description

Causes all commands that list nodes, devices, or options to include internal listings. Internal nodes, devices, and options are normally hidden to the user.

The outputall option affects the following commands:

- .options node
  Includes all internal nodes.
- .options list
  Includes all internal nodes and devices.
- .options opts
  Includes all hidden and undocumented options.
- .print
  Tests internal devices and nodes for wildcard matching. When .print is used without arguments, includes all internal nodes and devices in the output.
- .probe
  Tests internal nodes and devices for wildcard matching. When .probe is used without arguments, includes all internal nodes and devices in the output.

See Also

“node” (page 335), “xref” (page 347), “opts” (page 338), “.print” (page 137), “.probe” (page 152)
pathnum

options pathnum = { true | false }

Default Value

false

Description

The pathnum option converts all node and element names in the output listings so that the subcircuit path name portion of each name is converted to a number. An accompanying cross-reference will be printed to show the correspondence of these numbers to the fully qualified subcircuit pathnames.
probelvl

.options probelvl = {level }

Default Value

Default level: 0

Description

Limits .probe and .print output to a given subcircuit level. For example,.options probelvl=level limits .probe output to just the subcircuit components at level \( n \). .options probelvl=2 would only output the main subcircuit elements and nodes.
**probetop**

```
.options probetop = { level }
```

**Default Level**

Default level: 0

**Description**

Sets the lowest level of subcircuit hierarchy that will be printed. (Limits .probe output to the top n subcircuit levels.) Level value equal to zero causes the limiting to be disabled. Valid user settings are level >0.

For example, .options probetop=1 will only output nodes and devices that are at the top-level of the circuit, e.g. N1, N2, out. .options probetop=2 would also include main subcircuit elements and nodes, e.g. x1.d1, xinv.out
prtdel

.options prtdel = prtdel  

where prtdel ≥ 0.0

Default Value

0.0 s

Description

Fixed time delay between output points in transient analysis. This does not affect the internal time step calculations needed to ensure solution accuracy.

.options prtdel affects the .tsim file output. If you want to decrease the number of data points in your .tsim file, you can use .options prtdel to set a larger timestep value for the output resolution which, in turn, will decrease the size of your output file.
prtinterp

.options prtinterp = { true | false }

Default Value

true

Description

When the prtdel option is set, prtinterp determines how solutions are calculated at the output time intervals. When set to true, prtdel timepoints are interpolated rather than computed.

- **prtinterp = 0** — In addition to the internal time steps, the T-Spice simulator takes time steps at the output intervals and calculates those solutions directly.
- **prtinterp = 1** — Output solutions are computed using linear interpolation of the T-Spice engine’s internal time step solutions.

The default value of option *prtinterp* is true, and the prtdel timepoints are interpolated. *prtdel* controls the frequency of printed output during a transient simulation. For example, a *prtdel* value of 1n should result in print statements being executed for each 1 nanosecond. But, prtdel specifically will not change the internal time-stepping algorithm. It will only change the printed timestep values, and use interpolation to get these values (unless *prtinterp* is set to false).
statdelay

.options statdelay = statdelay

Default Value

0.5 s

Description

Minimum delay in real time between updates of status display in the T-Spice user interface (GUI).
verbose

.options verbose = { 0 | 1 | 2 | 3 | 4 }

Default Value

1

Description

Level of detail of circuit and simulation information printed to the Simulation Window.

0
Prints only the processing phase and final runtimes.

1
Prints node and device counts and major runtime statistics. Lists options whose values have been modified from the default.

2
Prints all option settings (equivalent to opts = true) and all runtime statistics (equivalent to acct = true).

3
Prints node connectivity (equivalent to node = true), lists devices (equivalent to list = true), and prints conditional statement, subcircuit, and parameter cross reference listings (equivalent to xref = true).

4
Lists hidden or internal devices, nodes, and options (equivalent to outputall = true), and lists convergence residual statistics (equivalent to expert = true). If the input file contains .alter statements, then all log (listing) printouts will be given for each alter section.

See Also

xref

.options xref = { true | false }

Default Value

false

Description

Generates extensive cross-referencing information listings about the input circuit and simulation commands. The information includes:

- A list of all subcircuit instances;
- A tree outline of all conditional statements (.if () ... .endif);
- A listing of all parameters (.param definitions or subcircuit parameters) that are defined and used in the circuit.
Probing Options

“probei” (page 349)
“probemonte” (page 351)
“probesubckt” (page 353)

“probej” (page 350)
“probeq” (page 352)
“probev” (page 354)
probei

.probes probei = { true | false }

Default Value

false

Description

Instructs T-Spice to include device terminal current values in the output data generated by .probe and .print (when used without arguments).
probej

.options probej = { true | false }

Default Value

false

Description

Instructs T-Spice to include device terminal current density values in the output data generated by .probe and .print (when used without arguments).
probemonte

.options probemonte = { true | false }

Default Value

false

Description

Generate probe outputs during Monte Carlo analysis.
probeq

\[ \text{.options probeq = \{ true | false \}} \]

Default Value

false

Description

Instructs T-Spice to include device terminal charge values in the output data generated by `.probe` and `.print` (when used without arguments).
probesubckt

.options probesubckt = { true | false }

Default Value

false

Description

Generate probe data for all subcircuit pin currents (if option probei is true) and charges (if option probeq is true).
probev

.options probev = { true | false }

Default Value

false

Description

Instructs T-Spice to include node voltage values in the output data generated by .probe and .print (when used without arguments).
Verilog-A/MS Options

“aldecdir” (page 356)
“amsdigitalsim” (page 358)
“vadatabase” (page 360)
“vasearch” (page 362)
“vaverbose” (page 364)

“amscfg” (page 357)
“vacasemod”e” (page 359)
“vaexptol” (page 361)
“vatimetol” (page 363)
aldecdir

.option aldecdir = aldec_installation_path

Default Value

T-Spice will search for a top-level directory named \Aldec, and will select the most recent release of Riviera Pro that is found there. e.g. C:\Aldec\Riviera-PRO-2014.02-x64

Description

Use this command to identify the location of the Riviera Pro installation that you wish to use for Verilog-A/MS co-simulation.
amscfg

.option amscfg = filename

Default Value

ams.cfg

Description

The Verilog-A/MS system uses a set of files named ams.cfg for initialization of all runtime settings. The file can be defined at 3 different levels: computer system level, user level, and local / simulation level. The 3 directories that will be searched for an ams.cfg file are:

[1] [Tanner Tools installation path]\iburonda\config\ams.cfg
[2] %HOMEPATH%\ams.cfg
[3] \ams.cfg

Settings in the ams.cfg files are applied in the order: system-level, then user-level, then local-level. Local-level settings will therefore replace and override duplicate system or user level settings.

The amscfg option provides a mechanism for setting an additional ams.cfg file that will be processed, and can have an arbitrary name.

Example

.option amscfg=modelsim_ams.cfg
amsdigitalsim

.option amsdigitalsim = aldec | modelsim

Default Value

aldec

Description

T-Spice Verilog-AMS is able to co-simulate with either Aldec Riviera Pro or Modelsim digital simulators. Use this option to select the digital simulator that will be driven by Verilog-AMS.
vacasemode

```
.option vacasemode = sensitive | insensitive | preferred
```

Default Value

sensitive

Description

Selects the case-sensitivity mode used for resolving node and parameter names that are passed across the analog simulator / digital simulator boundary.
vadatabase

```
.option vadatabase = precompiled_database_path
```

Default Value

```
-
```

Description

Use this command to identify the location of a pre-compiled Verilog-A database of modules to be used for this simulation.
vaexprtol

.option vaexprtol = tolerance

Default Value

absv

Description

Sets the default expression tolerance ( expr_tol ) for the Verilog-A cross() and above() functions.

See Also

“absv | vntol” (page 230)
vasearch

\[ \text{.option vasearch = path1 [ ; path2 [ ; … ]]} \]

Default Value

None

Description

Adds directories to the search path for Verilog-A files. Pathnames containing spaces must be enclosed in quotes.

T-Spice looks for Verilog-A files in the following default order:

1. Current working directory (directory containing the T-Spice input file)
2. Path(s) specified in the Simulation Settings dialog
3. Path(s) specified by the \text{.option vasearch} command
4. \text{.verilogA/models} subdirectory of the local T-Spice installation
vatimetol

```
.option vatimetol = tolerance
```

Default Value

```
10 * mintimestep
```

Description

Sets the default time tolerance \((time\_tol)\) for the Verilog-A \texttt{cross()}, \texttt{timer()}, and \texttt{above()} functions.

See Also

```
mintimestep
```
vaverbose

.option vaverbose [ = { true | false | 1 | 0 } ]

Default Value

0

Description

Enables or disables verbose printing of Verilog-A compiler settings, search paths, loaded modules, etc. This information is displayed in the Simulation Status window.

Possible settings are:

0 or false  Do not print information.
1 or true    Print information.
8 Device Models

Introduction

This chapter describes the predefined analytical device models. Original documentation from the developers of the models is provided with T-Spice for further reference in the models folder of the standard installation directory.

The T-Spice built-in device models are distributed as a collection of external dynamically linked libraries (DLLs) located in the TSpiceModels subdirectory of the installation. This modularization improves the performance, quality and features of all built-in device models.

Displaying Supported Models and Parameters

The help system provides a general list of all model equations that T-Spice supports, accessible from Help > Models Supported by T-Spice. You can sort this list by any of the columns.
Pressing **Details**... will display the supported parameters, default devices values, units and states for the selected model (as shown below for BJT level 1).

Use the **Export** button on either of these dialogs to export the contents of the current tab to a comma-delimited text file.

You can also see a list of library models, T-Spice models and T-Spice devices for a specific simulation if one of the following conditions is true — probing is enabled for the simulation, the print option is enabled or the list option is set to on.
BIPOLAR Level/Model Cross Reference

T-Spice supports a number of different bipolar models, which are selectable using the `level` parameter in the `.model` statement. The association between model levels and the model type is shown in the following table.

<table>
<thead>
<tr>
<th>Bipolar level</th>
<th>Model</th>
<th>Further Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Gummel-Poon bipolar with extensions</td>
<td>BJT Level 1 (Gummel-Poon) (page 373)</td>
</tr>
<tr>
<td>6</td>
<td>Philips Mextram, Philips SiMKit version 2.4, BJT503 and 504 models</td>
<td>BJT Level 6 (Mextram) (page 393)</td>
</tr>
<tr>
<td>9</td>
<td>VBIC</td>
<td>BJT Level 9 (VBIC) (page 394)</td>
</tr>
<tr>
<td>10</td>
<td>Philips Modella Lateral PNP</td>
<td>BJT Level 10 (Modella) (page 399)</td>
</tr>
</tbody>
</table>

Diode Level/Model Cross Reference

The diode models which are supported by T-Spice, together with their model level associations, are shown in the following table.

<table>
<thead>
<tr>
<th>Diode level</th>
<th>Model</th>
<th>Further Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Non-geometric Junction Diode</td>
<td>Diode (page 403)</td>
</tr>
<tr>
<td>2</td>
<td>Fowler-Nordheim</td>
<td>Diode (page 403)</td>
</tr>
<tr>
<td>3</td>
<td>Geometric Junction Diode</td>
<td>Diode (page 403)</td>
</tr>
<tr>
<td>4</td>
<td>Philips Juncap versions 1 &amp; 2</td>
<td>Diode (page 403)</td>
</tr>
<tr>
<td>500</td>
<td>Philips Advanced Diode SiMKit v4.2, DIO500 model</td>
<td>Diode (page 403)</td>
</tr>
</tbody>
</table>
The diode models which are supported by T-Spice, together with their model level associations, are shown in the following table.

### JFET & MESFET Level/Model Cross Reference

<table>
<thead>
<tr>
<th>JFET/ MESFET level</th>
<th>Model</th>
<th>Further Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Schichmann and Hodges JFET</td>
<td>JFET (page 417)</td>
</tr>
<tr>
<td>1</td>
<td>Curtice MESFET</td>
<td>MESFET (page 420)</td>
</tr>
<tr>
<td>2</td>
<td>Statz MESFET</td>
<td>MESFET (page 420)</td>
</tr>
<tr>
<td>3</td>
<td>extended Curtice GaAs MESFET</td>
<td>MESFET (page 420)</td>
</tr>
</tbody>
</table>

### MOSFET Level/Model Cross Reference

Particular MOSFET models are selected by use of the level parameter in the `.model` statement. The association between model levels and the model type is shown in the following tables, where Berkeley BSIM models are listed in a separate table.

**Note:**
Some models can be referenced using more than one level number. This does not imply that the models are different. It is usually the case that the additional level is added for compatibility of input files from other simulators (Berkeley SPICE, HSPICE®, PSPICE®, etc.). For example, the EKV model can be selected as either MOSFET level 44, for SmartSpice compatibility, or as level 55, for HSPICE compatibility.

<table>
<thead>
<tr>
<th>MOSFET level</th>
<th>Model</th>
<th>Further documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SPICE level 1</td>
<td>MOSFET Levels 1/2/3 (Berkeley SPICE 2G6) (page 430)</td>
</tr>
<tr>
<td>2</td>
<td>SPICE level 2</td>
<td>MOSFET Levels 1/2/3 (Berkeley SPICE 2G6) (page 430)</td>
</tr>
<tr>
<td>3</td>
<td>SPICE level 3</td>
<td>MOSFET Levels 1/2/3 (Berkeley SPICE 2G6) (page 430)</td>
</tr>
<tr>
<td>5</td>
<td>proprietary Maher-Mead model</td>
<td>MOSFET Level 5 (Maher-Mead) (page 451)</td>
</tr>
<tr>
<td>9</td>
<td>Philips MOS 9 (identical to level 50)</td>
<td>MOSFET Level 20 (Philips MOS 20) (page 474)</td>
</tr>
<tr>
<td>11</td>
<td>Philips MOS 11 (identical to level 63)</td>
<td>MOSFET Levels 11 and 63 (Philips MOS 11) (page 463)</td>
</tr>
<tr>
<td>15</td>
<td>RPI Amorphous-Si TFT Model</td>
<td>MOSFET Levels 15 and 61 (RPI Amorphous-Si TFT Model) (page 467)</td>
</tr>
<tr>
<td>16</td>
<td>RPI Poly-Si TFT Model (equivalent to AimSpice level 16 PSIA2)</td>
<td>MOSFET Levels 16 and 62 (RPI Poly-Si TFT Model, 1.0 and 2.0) (page 470)</td>
</tr>
<tr>
<td>20</td>
<td>Philips MOS 20</td>
<td>MOSFET Level 20 (Philips MOS 20) (page 474)</td>
</tr>
</tbody>
</table>
T-Spice supports most of the models that are available in the Philips SiMKit compact transistor model library. The SiMKit library contains a variety of diode, bipolar, and MOSFET models, and has been made freely available to the circuit and model design community.
**Note:**

The Philips models do not support HSPICE extensions (ACM, HSPICE diodes, etc.)

In order to facilitate easy selection of Philips models, an additional parameter has been added to the `.model` statement: `model=modelname`. The `modelname` value will determine the Philips model which will be selected, and will override the `level` and `version` parameter values in the `.model` statement, if they are present. The possible values for `modelname` are listed in the following table.

In this table, the `e/g` column notes whether the model is electrical or geometrical; the `thermal` column states whether the model includes self-heating equations; and the `substrate` column states whether it is a bipolar model which includes the substrate current and charge terms.

<table>
<thead>
<tr>
<th>Model</th>
<th>Type</th>
<th>Level</th>
<th>Modelname</th>
<th>e/g</th>
<th>Thermal</th>
<th>Substrate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Juncap Level 1</td>
<td>d</td>
<td>1</td>
<td>juncap</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Juncap2 Level 200</td>
<td>d</td>
<td>200</td>
<td>juncap200</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Advanced Diode Level 500</td>
<td>d</td>
<td>500</td>
<td>dio500</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modella</td>
<td>pnp</td>
<td>500</td>
<td>btj500</td>
<td>e</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5006</td>
<td>btj500t</td>
<td>e</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Mextram Level 503</td>
<td>npn/pnp</td>
<td>503 / 6</td>
<td>btj503</td>
<td>e</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>503</td>
<td>btjd503</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mextram Level 504</td>
<td>npn/pnp</td>
<td>504</td>
<td>btj504</td>
<td>e</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>504</td>
<td>btj504t</td>
<td>e</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5045</td>
<td>btjd504t</td>
<td>e</td>
<td></td>
<td>yes</td>
</tr>
<tr>
<td>PSP Level 100.1</td>
<td>nmos/pmos</td>
<td>100</td>
<td>psp100 or psp100e</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>1000</td>
<td>psp1000 or psp100g</td>
<td>g</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mos 11 Level 1100</td>
<td>nmos/pmos</td>
<td>1100</td>
<td>mos1100e</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mos1100</td>
<td>g</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mos 11 Level 1101</td>
<td>nmos/pmos</td>
<td>1101</td>
<td>mos1101e</td>
<td>e</td>
<td>yes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mos1101et</td>
<td>e</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>11010</td>
<td>g</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mos11010</td>
<td>g</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mos11010t</td>
<td>g</td>
<td></td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(binning)</td>
<td>11011</td>
<td>g</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mos11011</td>
<td>g</td>
<td></td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>mos11011t</td>
<td>g</td>
<td></td>
<td>yes</td>
</tr>
</tbody>
</table>
For each model class, the following information is typically given.

- The `.model` command to be used in the input file. The `.model` command initializes the model by specifying its name, type, level, and parameter values.
- The parameters that determine the model’s characteristics. These are given in a table of parameter names (as used in the code), symbols (as used in the equations), descriptions, default values, and units.
- The circuit underlying the large-signal behavior of the model.
- The analytical equations that constitute the model.
The following abbreviations and conventions are employed.

| (Vertical bar.) In syntax paradigms, separates alternative values. In parameter tables, separates alternative names (aliases).
Computed. Indicates a parameter whose value is computed, not fixed or assigned.
[n] Indicates the NMOS value of a parameter.
[p] Indicates the PMOS value of a parameter.
Symbol for square.
DIBL Abbreviation for drain-induced barrier lowering.
LDD Abbreviation for lightly-doped drain.

## Constants

The following physical constants are used in parameter and equation evaluation.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varepsilon_0$</td>
<td>Dielectric permittivity of vacuum</td>
<td>$8.85421 \times 10^{-12}$</td>
<td>F/m</td>
</tr>
<tr>
<td>$\varepsilon_{ox}$</td>
<td>Relative dielectric permittivity of SiO$_2$</td>
<td>3.9</td>
<td>—</td>
</tr>
<tr>
<td>$\varepsilon_{si}$</td>
<td>Relative dielectric permittivity of silicon</td>
<td>11.7</td>
<td>—</td>
</tr>
<tr>
<td>$n_i$</td>
<td>Intrinsic carrier concentration at 300 K</td>
<td>$1.45 \times 10^{16}$</td>
<td>m$^{-3}$</td>
</tr>
<tr>
<td>$q$</td>
<td>Elementary electron charge</td>
<td>$1.6021918 \times 10^{-19}$</td>
<td>C</td>
</tr>
<tr>
<td>$\pi$</td>
<td>Pi</td>
<td>3.1415926</td>
<td>—</td>
</tr>
<tr>
<td>$k$</td>
<td>Boltzmann’s constant</td>
<td>$1.3806226 \times 10^{-23}$</td>
<td>V·C/K</td>
</tr>
<tr>
<td>$-0$</td>
<td>0 °C</td>
<td>273.15</td>
<td>K</td>
</tr>
<tr>
<td>$T_{nom}$</td>
<td>Default nominal model temperature</td>
<td>Specified by .options tnom (default: 25)</td>
<td>°C</td>
</tr>
<tr>
<td>$e$</td>
<td>Base of natural logarithms</td>
<td>2.7182818</td>
<td>—</td>
</tr>
</tbody>
</table>
BJT Level 1 (Gummel-Poon)

The level 1 bipolar junction transistor model uses a modified version of the Gummel-Poon charge-control model that was implemented in the original SPICE. The model simplifies to the Ebers-Moll model when certain parameters are not specified. It also includes high-bias and temperature effects.

**Note:**

Gummel-Poon is the default model which will be used for all bipolar models (type npn or pnp) which do not have the level model parameter set.

**Parameters**

The BJT model uses the following syntax.

```
.model name npn | pnp [level=1] [parameters]
```

The following tables describe all of the Gummel-Poon BJT parameters that T-Spice supports. Parameters that are not specified in the Ebers-Moll model are marked with an asterisk (*).
### DC Current Parameters

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>Model selector</td>
<td>1.0</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>subs</td>
<td>Substrate connection selector</td>
<td>1 (npn); -1 (pnp)</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>update</td>
<td>Equation selector for base charge.</td>
<td>0</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>dcap</td>
<td>Equation selector for depletion capacitance.</td>
<td>2</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>bf</td>
<td>$\beta_f$</td>
<td>Ideal forward maximum current gain.</td>
<td>100.0</td>
<td>—</td>
</tr>
<tr>
<td>br</td>
<td>$\beta_r$</td>
<td>Ideal reverse maximum current gain.</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>ibc</td>
<td>$I_{bc}$</td>
<td>Reverse saturation current between base and collector.</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ibe</td>
<td>$I_{be}$</td>
<td>Reverse saturation current between base and emitter.</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>iss</td>
<td>$I_{ss}$</td>
<td>Reverse saturation current between bulk and collector for vertical geometry, or between bulk and base for lateral geometry</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>is</td>
<td>$I_s$</td>
<td>Transport saturation current.</td>
<td>$1.0 \times 10^{-16}$</td>
<td>A</td>
</tr>
<tr>
<td>c2 (jle)</td>
<td>$C_2$</td>
<td>Non-ideality factor for base-emitter leakage saturation current.*</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>c4</td>
<td>$C_4$</td>
<td>Non-ideality factor for base-collector leakage saturation current.*</td>
<td>—</td>
<td></td>
</tr>
<tr>
<td>ise</td>
<td>$I_{se}$</td>
<td>Base-emitter leakage saturation current.*</td>
<td>$c_2 \times i_s$</td>
<td>A</td>
</tr>
<tr>
<td>isc</td>
<td>$I_{sc}$</td>
<td>Base-collector leakage saturation current.*</td>
<td>$c_4 \times i_s$</td>
<td>A</td>
</tr>
<tr>
<td>nf</td>
<td>$\eta_f$</td>
<td>Forward current emission coefficient.*</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>nr</td>
<td>$\eta_r$</td>
<td>Reverse current emission coefficient.*</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>ns</td>
<td>$\eta_s$</td>
<td>Substrate current emission coefficient.</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>nc (nle)</td>
<td>$\eta_c$</td>
<td>Base-collector leakage emission coefficient.*</td>
<td>2.0</td>
<td>—</td>
</tr>
<tr>
<td>ne (nle)</td>
<td>$\eta_e$</td>
<td>Base-emitter leakage emission coefficient.*</td>
<td>1.5</td>
<td>—</td>
</tr>
<tr>
<td>expli</td>
<td>EXPLI</td>
<td>Current explosion model parameter. The PN junction characteristics above the explosion current area are linear, with the slope at the explosion point.</td>
<td>$1.0 \times 10^{15}$</td>
<td>A</td>
</tr>
</tbody>
</table>

### Base Charge Parameters
Base charge parameters are not specified for the Ebers-Moll model.

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>vaf (vbf)</td>
<td>Vaf</td>
<td>Forward early voltage.*</td>
<td>0.0 (indicates infinite value)</td>
<td>V</td>
</tr>
<tr>
<td>var (vb, vbb)</td>
<td>V_{ar}</td>
<td>Reverse early voltage.*</td>
<td>0.0 (indicates infinite value)</td>
<td>V</td>
</tr>
<tr>
<td>ikf (ik, jbf)</td>
<td>I_{kf}</td>
<td>Corner for forward Beta high current roll-off.*</td>
<td>0.0 (indicates infinite value)</td>
<td>A</td>
</tr>
<tr>
<td>ikr (jbr)</td>
<td>I_{kr}</td>
<td>Corner for reverse Beta high current roll-off.*</td>
<td>0.0 (indicates infinite value)</td>
<td>A</td>
</tr>
<tr>
<td>nkf</td>
<td>\eta_{kf}</td>
<td>Exponent for high current Beta roll-off.*</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

**Parasitic Resistor Parameters**

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>irb (jrb,iob)</td>
<td>I_{rb}</td>
<td>Base current, where base resistance falls halfway between ( r_b ) and ( r_{bm} ).*</td>
<td>0.0 (indicates infinite value)</td>
<td>A</td>
</tr>
<tr>
<td>rb</td>
<td>r_{b}</td>
<td>Base resistance.*</td>
<td>0.0</td>
<td>( \Omega )</td>
</tr>
<tr>
<td>rbm</td>
<td>r_{bm}</td>
<td>Minimum high current base resistance.*</td>
<td>( r_{b} )</td>
<td>( \Omega )</td>
</tr>
<tr>
<td>re</td>
<td>r_{e}</td>
<td>Emitter resistance.</td>
<td>0.0</td>
<td>( \Omega )</td>
</tr>
<tr>
<td>rc</td>
<td>r_{c}</td>
<td>Collector resistance.</td>
<td>0.0</td>
<td>( \Omega )</td>
</tr>
</tbody>
</table>

**Parasitic Capacitance Parameters.**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>cbcp</td>
<td>CBCP</td>
<td>External base-collector constant capacitance.</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>cbep</td>
<td>CBEP</td>
<td>External base-emitter constant capacitance.</td>
<td>0.0</td>
<td>( \Phi )</td>
</tr>
<tr>
<td>ccsp</td>
<td>CCSP</td>
<td>External collector-substrate (vertical) or base-substrate (lateral) constant capacitance.</td>
<td>0.0</td>
<td>( \Phi )</td>
</tr>
</tbody>
</table>

**Junction Capacitance Parameters**

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>cjc</td>
<td>Cjc</td>
<td>Base-collector zero-bias depletion capacitance.</td>
<td>0.0</td>
<td>F</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>cje</td>
<td>C_{je}</td>
<td>Base-emitter zero-bias depletion capacitance.</td>
<td>0.0</td>
<td>Φ</td>
</tr>
<tr>
<td>cjs (ccs, csub)</td>
<td>C_{js}</td>
<td>Zero-bias collector-substrate capacitance.</td>
<td>0.0</td>
<td>Φ</td>
</tr>
<tr>
<td>fc</td>
<td>FC</td>
<td>Coefficient for forward bias depletion capacitance.</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>mjc (mc)</td>
<td>m_{jc}</td>
<td>Base-collector junction exponent (grading factor).</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>mje (me)</td>
<td>m_{je}</td>
<td>Base-emitter junction exponent (grading factor).</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>mjs (esub)</td>
<td>m_{js}</td>
<td>Substrate junction exponent (grading factor).</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>vjc (pc)</td>
<td>V_{jc}</td>
<td>Base-collector built-in potential.</td>
<td>0.75</td>
<td>V</td>
</tr>
<tr>
<td>vje (pe)</td>
<td>V_{je}</td>
<td>Base-emitter built-in potential.</td>
<td>0.75</td>
<td>V</td>
</tr>
<tr>
<td>vjs (psub)</td>
<td>V_{js}</td>
<td>Substrate junction built-in potential.</td>
<td>0.75</td>
<td>V</td>
</tr>
<tr>
<td>xcjc (cdis)</td>
<td>X_{cjc}</td>
<td>Internal base fraction of base-collector depletion capacitance.</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

### Transit Time parameters

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ptf</td>
<td>P_{tf}</td>
<td>Frequency multiplier to determine excess phase.</td>
<td>0.0</td>
<td>deg.</td>
</tr>
<tr>
<td>tf</td>
<td>τ_f</td>
<td>Base forward transit time.</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>tr</td>
<td>τ_r</td>
<td>Base reverse transit time.</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>vtf</td>
<td>V_{tf}</td>
<td>Voltage for $V_{bc}$ dependence of $\tau_f$.</td>
<td>0.0 (indicates infinite value)</td>
<td>ζ</td>
</tr>
<tr>
<td>xtf</td>
<td>X_{tf}</td>
<td>Coefficient for bias dependence of $\tau_f$.</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>itf (jtf)</td>
<td>I_{tf}</td>
<td>Parameter for high-current effect on $\tau_f$.</td>
<td>0.0</td>
<td>A</td>
</tr>
</tbody>
</table>
### Noise Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>(af)</td>
<td>(AF)</td>
<td>Flick noise exponent</td>
<td>1.0</td>
</tr>
<tr>
<td>(kf)</td>
<td>(KF)</td>
<td>Flick noise exponent</td>
<td>0.0</td>
</tr>
</tbody>
</table>

### Temperature Effect Parameters

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t_{\text{nom}})</td>
<td>(T_{\text{nom}})</td>
<td>Nominal temperature</td>
<td>global (t_{\text{nom}}) (25.0)</td>
<td>deg</td>
</tr>
<tr>
<td>(t_{\text{lev}})</td>
<td>(t_{\text{lev}})</td>
<td>Temperature equation selector</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>(t_{\text{levc}})</td>
<td>(t_{\text{levc}})</td>
<td>Temperature equation selector for junction capacitances and potentials</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>(b_{\text{ex}})</td>
<td>(B_{\text{ex}})</td>
<td>(VT) temperature exponent. (Level 2 only.)</td>
<td>2.42</td>
<td></td>
</tr>
<tr>
<td>(b_{\text{exv}})</td>
<td>(B_{\text{exv}})</td>
<td>(Rc) temperature exponent. ((t_{\text{lev}}=2) only.)</td>
<td>1.90</td>
<td></td>
</tr>
<tr>
<td>(c_{\text{tc}})</td>
<td>(C_{\text{tc}})</td>
<td>Temperature coefficient for zero-bias base-collector capacitance.</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(c_{\text{te}})</td>
<td>(C_{\text{te}})</td>
<td>Temperature coefficient for zero-bias base-emitter capacitance.</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(c_{\text{ts}})</td>
<td>(C_{\text{ts}})</td>
<td>Temperature coefficient for zero-bias substrate capacitance.</td>
<td>0.0</td>
<td>1/(\delta e\gamma)</td>
</tr>
<tr>
<td>(e_{\text{g}})</td>
<td>(E_{\text{g}}(0))</td>
<td>Energy gap at 0(^{\circ})K for (t_{\text{lev}} = 2) (default):</td>
<td>1.16</td>
<td>eV</td>
</tr>
<tr>
<td></td>
<td></td>
<td>for (t_{\text{lev}} = 0, 1,) or 3:</td>
<td>1.11</td>
<td></td>
</tr>
<tr>
<td>(\text{gap1})</td>
<td>(\text{GAP1})</td>
<td>Coefficient in energy gap temperature equation</td>
<td>(7.02 \times 10^{-4}) eV/deg</td>
<td></td>
</tr>
<tr>
<td>(\text{gap2})</td>
<td>(\text{GAP2})</td>
<td>Coefficient in energy gap temperature equation</td>
<td>1108.0</td>
<td>deg</td>
</tr>
<tr>
<td>(\text{tbf1})</td>
<td>(\text{TB}F_{1})</td>
<td>First order and second order temperature coefficients for (\beta_f).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tbf2})</td>
<td>(\text{TB}F_{2})</td>
<td>First order and second order temperature coefficients for (\beta_f).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tbr1})</td>
<td>(\text{TB}R_{1})</td>
<td>First order and second order temperature coefficients for (\beta_r).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tbr2})</td>
<td>(\text{TB}R_{2})</td>
<td>First order and second order temperature coefficients for (\beta_r).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tikf1})</td>
<td>(\text{TI}KF_{1})</td>
<td>First order and second order temperature coefficients for (\text{Ikf}).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tikf2})</td>
<td>(\text{TI}KF_{2})</td>
<td>First order and second order temperature coefficients for (\text{Ikf}).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tikr1})</td>
<td>(\text{TI}KR_{1})</td>
<td>First order and second order temperature coefficients for (\text{Ikr}).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tikr2})</td>
<td>(\text{TI}KR_{2})</td>
<td>First order and second order temperature coefficients for (\text{Ikr}).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tirb1})</td>
<td>(\text{TI}RB_{1})</td>
<td>First order and second order temperature coefficients for (\text{Irb}).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>(\text{tirb2})</td>
<td>(\text{TI}RB_{2})</td>
<td>First order and second order temperature coefficients for (\text{Irb}).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>Parameter (alias)</td>
<td>Symbol</td>
<td>Description</td>
<td>Default</td>
<td>Units</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
<td>---------</td>
<td>--------</td>
</tr>
<tr>
<td>tisc1</td>
<td>TISC₁</td>
<td>First order and second order temperature coefficients for ( Isc ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TISC₂</td>
<td>(tlev=3 only.)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tis1</td>
<td>TIS₁</td>
<td>First order and second order temperature coefficients for ( Is ) or ( Ibe ) and ( Ibc ). (tlev=3 only).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TIS₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tise1</td>
<td>TISE₁</td>
<td>First order and second order temperature coefficients for ( Is ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TISE₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tiss1</td>
<td>TISS₁</td>
<td>First order and second order temperature coefficients for ( Iss ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TISS₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>titf1</td>
<td>TITF₁</td>
<td>First order and second order temperature coefficients for ( Itf ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TITF₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tmjc1</td>
<td>TMJC₁</td>
<td>First order and second order temperature coefficients for ( mjc ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TMJC₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tmje1</td>
<td>TMJE₁</td>
<td>First order and second order temperature coefficients for ( mje ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TMJE₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tmjs1</td>
<td>TMJS₁</td>
<td>First order and second order temperature coefficient for ( mjs ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TMJS₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tnc1</td>
<td>TNC₁</td>
<td>First order and second order temperature coefficients for ( \eta ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TNC₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tne1</td>
<td>TNE₁</td>
<td>First order and second order temperature coefficients for ( \eta ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TNE₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tnf1</td>
<td>TNF₁</td>
<td>First order and second order temperature coefficients for ( \eta ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TNF₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tnr1</td>
<td>TNR₁</td>
<td>First order and second order temperature coefficients for ( \eta ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TNR₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tns1</td>
<td>TNS₁</td>
<td>First order and second order temperature coefficients for ( \eta ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TNS₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>trb1</td>
<td>TRB₁</td>
<td>First order and second order temperature coefficients for ( rb ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TRB₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>trc1</td>
<td>TRC₁</td>
<td>First order and second order temperature coefficients for ( rc ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TRC₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tre1</td>
<td>TRE₁</td>
<td>First order and second order temperature coefficients for ( re ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TRE₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>trm1</td>
<td>TRM₁</td>
<td>First order and second order temperature coefficients for ( rbm ).</td>
<td>thr₁</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TRM₂</td>
<td>(tlev=3 only)</td>
<td>thr₂</td>
<td>1/deg²</td>
</tr>
<tr>
<td>ttf1</td>
<td>TTF₁</td>
<td>First order and second order temperature coefficients for ( \tau ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TTF₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>ttr1</td>
<td>TTR₁</td>
<td>First order and second order temperature coefficients for ( tr ).</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td></td>
<td>TTR₂</td>
<td>(tlev=3 only)</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
</tbody>
</table>
### Large-Signal Model

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>tvaf1</td>
<td>TVF₁</td>
<td>First order and second order temperature coefficients for $V_{af}$</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>tvaf2</td>
<td>TVF₂</td>
<td>second order temperature coefficients for $V_{af}$</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tvar1</td>
<td>TVR₁</td>
<td>First order and second order temperature coefficients for $V_{ar}$</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>tvar2</td>
<td>TVR₂</td>
<td>second order temperature coefficients for $V_{ar}$</td>
<td>0.0</td>
<td>1/deg²</td>
</tr>
<tr>
<td>tvjc</td>
<td>$T_{vjc}$</td>
<td>Temperature coefficient for $V_{jc}$</td>
<td>0.0</td>
<td>V/deg</td>
</tr>
<tr>
<td>tvje</td>
<td>$T_{vje}$</td>
<td>Temperature coefficient for $V_{je}$</td>
<td>0.0</td>
<td>V/deg</td>
</tr>
<tr>
<td>tvjs</td>
<td>$T_{vjs}$</td>
<td>Temperature coefficient for $V_{js}$</td>
<td>0.0</td>
<td>V/deg</td>
</tr>
<tr>
<td>xtb</td>
<td>$X_{tb}$</td>
<td>Forward and reverse Beta temperature exponent</td>
<td>0.0</td>
<td>V/deg</td>
</tr>
<tr>
<td>(tb, tcb)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>xti</td>
<td>$X_{ti}$</td>
<td>Saturation current temperature exponent. (Use $X_{ti}=3.0$ for Silicon diffused junction, and $X_{ti}=2.0$ for Schottky barrier diode.)</td>
<td>3.0</td>
<td></td>
</tr>
</tbody>
</table>

### Equations

The Gummel-Poon model includes four “second-order” current effects:

- The low-current drop in current gain $\beta$ is due to extra components of base current $IB$. The parameters $I_{sc}$, $\eta_c$ (the low current drop in $\beta_c$) and $I_{se}$, $\eta_e$ (the low current drop in $\beta_e$) describe this effect, and two non-ideal diodes are included in the Large-Signal Model, above as shown.
- Base-width modulation, affecting the BJT output conductance, is described by $V_{ar}$ (reverse Early voltage) and $V_{af}$ (forward Early voltage).
- High-level injection, modifying the slope of the log($IC$) vs. $V_{be}$ characteristic, is described by two “knee” currents, $I_{kf}$ and $I_{kr}$, for the forward and reverse regions of operation.
- Base resistance is current dependent and is modeled by a combination of $r_b$, $r_{bm}$, and $I_{rb}$.

The Gummel-Poon large-signal model for transient simulations is topologically identical to the Ebers-Moll large-signal model except for the inclusion of the distributed base-collector capacitance. In addition, the effect of $\tau_f$ modulation (transit time) is handled.

**Geometry Consideration**

The BJT model has two geometric configurations based on physical layout: vertical and lateral. The geometric configuration is specified by the substrate connection selector parameter, `subs`. Set `subs` equal to 1 to specify vertical geometry or to $-1$ to specify lateral geometry. The default value of `subs` is 1 for npn transistors and $-1$ for pnp transistors.

The instance parameters `area`, `areab`, `areac`, and `M` specify geometric scaling for current and charge values in the Gummel-Poon BJT. (See “BJT (q)” on page 172 for more information about these parameters.)
Scaling: Both Vertical and Lateral Geometry

\[ I_{s_{\text{eff}}} = \text{area} \cdot M \cdot I_s \]
\[ I_{be_{\text{eff}}} = \text{area} \cdot M \cdot I_{be} \]
\[ I_{se_{\text{eff}}} = \text{area} \cdot M \cdot I_{se} \]
\[ I_{kf_{\text{eff}}} = \text{area} \cdot M \cdot I_{kf} \]
\[ I_{kr_{\text{eff}}} = \text{area} \cdot M \cdot I_{kr} \]
\[ I_{rb_{\text{eff}}} = \text{area} \cdot M \cdot I_{rb} \]
\[ I_{ss_{\text{eff}}} = \text{area} \cdot M \cdot I_{ss} \quad \text{if both } I_{be} \text{ and } I_{bc} \text{ are NOT specified} \]
\[ I_{tf_{\text{eff}}} = \text{area} \cdot M \cdot I_{tf} \]

\( \text{EXPLI}_{\text{eff}} = \text{area} \cdot M \cdot \text{EXPLI} \) \hspace{1cm} (8.54)

\[ C_{bcp_{\text{eff}}} = \text{area} \cdot M \cdot C_{bcp} \]
\[ C_{bep_{\text{eff}}} = \text{area} \cdot M \cdot C_{bep} \]
\[ C_{csp_{\text{eff}}} = \text{area} \cdot M \cdot C_{csp} \]
\[ C_{je_{\text{eff}}} = \text{area} \cdot M \cdot C_{je} \] \hspace{1cm} (8.55)

\[ r_{b_{\text{eff}}} = \frac{r_b}{\text{area} \cdot M} \]
\[ r_{bm_{\text{eff}}} = \frac{r_{bm}}{\text{area} \cdot M} \] \hspace{1cm} (8.56)
\[ r_{e_{\text{eff}}} = \frac{r_e}{\text{area} \cdot M} \]
\[ r_{ce_{\text{eff}}} = \frac{r_c}{\text{area} \cdot M} \]

Scaling: Vertical Geometry (subs=1)

\[ I_{bc_{\text{eff}}} = \text{area}_{ab} \cdot M \cdot I_{bc} \]
\[ I_{sc_{\text{eff}}} = \text{area}_{ab} \cdot M \cdot I_{sc} \]
\[ I_{ss_{\text{eff}}} = \text{area}_{ab} \cdot M \cdot I_{ss} \quad \text{if both } I_{be} \text{ and } I_{bc} \text{ are specified} \] \hspace{1cm} (8.57)
\[ C_{js_{\text{eff}}} = \text{area}_{ab} \cdot M \cdot C_{js} \]
\[ C_{jc_{\text{eff}}} = \text{area}_{ab} \cdot M \cdot C_{jc} \]
Scaling: Lateral Geometry (subs=-1)

\[
\begin{align*}
I_{bc_{\text{eff}}} &= areac \cdot M \cdot I_{bc} \\
I_{sc_{\text{eff}}} &= areac \cdot M \cdot I_{sc} \\
I_{ss_{\text{eff}}} &= areac \cdot M \cdot I_{ss} \quad \text{if both } I_{bc} \text{ and } I_{bc} \text{ are specified} \\
C_{jse_{\text{eff}}} &= areac \cdot M \cdot C_{js} \\
C_{jce_{\text{eff}}} &= areac \cdot M \cdot C_{jc} \\
\end{align*}
\]  

(8.58)

Current Equations (Level 1)

The following current equations are used when \text{level}=1, specifying that either the Gummel-Poon or Ebers-Moll model will be used. T-Spice chooses appropriate current equations according to which parameters were specified. The reverse saturation currents between base and collector and between base and emitter (\(I_{bc}\) and \(I_{be}\)) are optional parameters. If they are not specified, T-Spice uses the transport saturation current \(I_s\) in current equations.

There are no explicit switches between the different regions of device operation. Equations (8.59) through (8.71) describe currents in the normal active, inverse, saturation, and cut-off regions of operation. This means that the \(\beta\) roll-off at high collector current is slightly less pronounced in T-Spice.

Base Charge Equations

In order to determine BJT currents, T-Spice must first calculate base charge. There are two sets of base charge equations; you can specify which equations T-Spice will use by changing the model selector \text{update}.

The base charge \(q_b\) is calculated from the following equation:

\[
q_b = \frac{q_1}{2} \left(1 + \left(1 + 4q_2\right)^{\frac{\eta_f}{4}}\right),
\]

(8.59)

where \(q_2\) and \(q_1\) are calculated as follows.

\[
q_2 = \frac{I_{sc_{\text{eff}}}}{I_{k_{f_{\text{eff}}}} \left(e^{\eta_f V_t} - 1\right)} + \frac{I_{se_{\text{eff}}}}{I_{k_{r_{\text{eff}}}} \left(e^{\eta_r V_t} - 1\right)}
\]

(8.60)

The thermal voltage \(V_t\) is defined as:

\[
V_t = kT/q,
\]

(8.61)

where \(k\) is Boltzmann’s constant, \(T\) is temperature, and \(q\) is the elementary electron charge.

If \text{update}=1 and \(\frac{V_{bc}}{V_{af}} + \frac{V_{bc}}{V_{ar}} \geq 0\), then

\[
q_1 = 1 + \frac{V_{bc}}{V_{af}} + \frac{V_{bc}}{V_{ar}}.
\]

(8.62)
If $\text{update}=0$ or \( \frac{V_{bc}}{V_{af}} + \frac{V_{bc}}{V_{ar}} < 0 \), then

\[
q_1 = \left(1 - \frac{V_{bc}}{V_{af}} - \frac{V_{bc}}{V_{ar}}\right)^{-1}. \quad (8.63)
\]

**Collector and Base Current Equations**

If only the saturation current \( I_s \) is specified, T-Spice uses the following equations to calculate collector current \( (I_c) \) and base current \( (I_b) \):

\[
I_c = \frac{I_{sc,eff}}{q_b} \left( e^{\frac{V_{be}}{\eta_f V_t}} - 1 \right) - \frac{I_{sc,eff}}{q_b} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) - I_{sc,eff} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) \quad (8.64)
\]

\[
I_b = \frac{I_{be,eff}}{q_b} \left( e^{\frac{V_{be}}{\eta_f V_t}} - 1 \right) - \frac{I_{be,eff}}{q_b} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) + I_{sc,eff} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) \quad (8.65)
\]

If \( I_{bc} \) and \( I_{bc} \) are both specified, T-Spice uses the following current equations:

\[
I_c = \frac{I_{bc,eff}}{q_b} \left( e^{\frac{V_{bc}}{\eta_f V_t}} - 1 \right) - \frac{I_{bc,eff}}{q_b} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) - I_{sc,eff} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) \quad (8.66)
\]

\[
I_b = \frac{I_{be,eff}}{q_b} \left( e^{\frac{V_{be}}{\eta_f V_t}} - 1 \right) + \frac{I_{be,eff}}{q_b} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) + I_{sc,eff} \left( e^{\frac{V_{bc}}{\eta_r V_t}} - 1 \right) \quad (8.67)
\]

where \( q_b \) is the normalized base charge, \( \beta_f \) and \( \beta_r \) are the ideal maximum reverse and forward current gains, \( \eta_f \) and \( \eta_r \) are the forward and reverse current emission coefficients, and \( \eta_c \) and \( \eta_e \) are the base-collector and emitter-collector leakage emission coefficients.

The last two terms in the \( I_b \) expression represent recombination in the base-emitter and base-collector surfaces and depletion regions. The last term in the \( I_c \) expression represents recombination in the base-collector junction.

The emitter current, \( I_E \), is simply the sum of base and collector currents:

\[
I_E = I_b + I_c. \quad (8.68)
\]

**Excess Phase Equation**

Excess phase represents the extra degrees of phase delay introduced by the BJT as a function of frequency, due to the distributed phenomena in the base region. T-Spice calculates excess phase in AC and transient analyses of the BJT. At any given frequency, the phase multiplier parameter \( p_{tf} \) determines the relationship between excess phase and the base forward transit time, \( \tau_f \).

\[
\text{excess phase} = \left(2\pi \cdot P_{tf} \cdot \frac{\tau_f}{360}\right) \cdot 2\pi f. \quad (8.69)
\]
In AC analysis, T-Spice applies excess phase as a linear phase delay in the transconductance generator, \( g_m \). See the small-signal reference for “BJT Level 1 (Gummel-Poon)” on page 509 for a description of \( g_m \).

In transient analysis, excess phase affects how T-Spice calculates collector current (\( I_c \)). When excess phase is present (i.e., \( ptf \neq 0 \)), T-Spice calculates the collector current as a cumulative function using time step dependent backward Euler integration. Otherwise, if \( ptf \) is not specified, collector current is simply a function of the current time step.

**Substrate Current Equations**

Substrate current is defined according to the geometric configuration of the BJT. The substrate current flows from substrate to collector for vertical BJTs (\( subs=1 \)) and from substrate to base for lateral BJTs (\( subs=-1 \)).

For vertical transistors (\( subs=1 \)),

\[
I_{sc} = I_{s_{\text{eff}}} \left( e^{\frac{V_{sc}}{\eta_s V_t}} - 1 \right) \quad \text{when} \quad V_{sc} > -10 \cdot \eta_s \cdot V_t, \quad (8.70)
\]

\[
I_{sc} = -I_{s_{\text{eff}}} \quad \text{when} \quad V_{sc} \leq -10 \cdot \eta_s \cdot V_t
\]

where \( I_{s_{\text{eff}}} \) is the effective reverse saturation current between bulk and collector.

For lateral transistors (\( subs=-1 \)),

\[
I_{bs} = I_{s_{\text{eff}}} \left( e^{\frac{V_{bs}}{\eta_s V_t}} - 1 \right) \quad \text{when} \quad V_{bs} > -10 \cdot \eta_s \cdot V_t, \quad (8.71)
\]

\[
I_{bs} = -I_{s_{\text{eff}}} \quad \text{when} \quad V_{bs} \leq -10 \cdot \eta_s \cdot V_t
\]

where \( I_{s_{\text{eff}}} \) is the effective reverse saturation current between bulk and base.

**Variable Base Resistance Equations**

The following equations describe the calculation of the base resistance, \( r_{bb} \). Base resistance varies with current and depends on two parameters, the low-current maximum resistance (\( r_{bm} \)) and the high-current minimum resistance (\( r_{bm} \)). There are two ways that T-Spice can calculate \( r_{bb} \), either with or without the parameter \( I_{rb} \). \( I_{rb} \) is the base current that occurs when \( r_{bb} \) is equal to \( 0.5 \times (r_{b} - r_{bm}) \), or the midpoint between minimum and maximum resistance values.

If \( I_{rb} \) is not specified, T-Spice uses the following equation:

\[
r_{bb} = r_{bm_{eff}} + \frac{r_{b_{eff}} - r_{bm_{eff}}}{q_b} \quad (8.72)
\]

If \( I_{rb} \) is specified,

\[
r_{bb} = r_{bm_{eff}} + 3(r_{b_{eff}} - r_{bm_{eff}}) \frac{\tan(z) - z}{z \tan(z) \tan(z)} \quad (8.73)
\]
where

\[
\frac{-1 + \sqrt{1 + \frac{144 \cdot I_b}{\pi^2 \cdot I_{rb,eff}^2}}}{24 \cdot \frac{I_b}{\pi^2 \cdot I_{rb,eff}}} = z.
\]  

(8.74)

**Capacitance Equations**

**Base-Emitter Capacitance**

The total base-emitter capacitance consists of contributions from diffusion capacitance and depletion capacitance:

\[
C_{be} = C_{be,\text{diff}} + C_{be,\text{dep}}
\]  

(8.75)

The diffusion capacitance is determined as follows:

\[
C_{be,\text{diff}} = \frac{\partial}{\partial V_{be}} \left( \tau_f \cdot \frac{i_{be}}{q_b} \right) \quad \text{when } i_{be} \leq 0
\]

(8.76)

\[
C_{be,\text{diff}} = \frac{\partial}{\partial V_{be}} \left( \tau_f \cdot (1 + \arg \tau_f) \cdot \frac{i_{be}}{q_b} \right) \quad \text{when } i_{be} > 0
\]

where

\[
\arg \tau_f = \sqrt{i_{be} + I_{ij}}^2 \cdot e^{-\frac{V_{be}}{1.44 \cdot V_{ef}}}
\]  

(8.77)

and

\[
i_{be} = I_{s,\text{eff}} \left( e^{qV_{be}/kT} - 1 \right).
\]  

(8.78)

T-Spice supports two different models for depletion capacitance. You can specify the depletion capacitance equations using the `dcap` model selector.

If `dcap=1`,

\[
C_{be,\text{dep}} = C_{je,\text{eff}} \left( 1 - \frac{V_{be}}{V_{je}} \right)^{-m_{je}} \quad \text{when } V_{be} < FC \cdot V_{je}
\]  

(8.79)

\[
C_{be,\text{dep}} = C_{je,\text{eff}} \cdot \frac{1 - FC(1 + m_{je}) + m_{je} \cdot \frac{V_{be}}{V_{je}}}{(1 - FC)^{1 + m_{je}}} \quad \text{when } V_{be} \geq FC \cdot V_{je}
\]
If \texttt{dcap=2} (default),

\[
C_{bc_{dep}} = C_{jc_{eff}} \left( 1 - \frac{V_{bc}}{V_{je}} \right)^{-m_{je}} \quad \text{when } V_{bc} < 0
\]

\[
C_{bc_{dep}} = C_{jc_{eff}} \left( 1 + m_{je} \cdot \frac{V_{bc}}{V_{je}} \right) \quad \text{when } V_{bc} \geq 0
\]

**Base-Collector Capacitance**

The total base-collector capacitance consists of contributions from diffusion capacitance and depletion capacitance:

\[
C_{bc} = C_{bc_{diff}} + C_{bc_{dep}} \quad (8.81)
\]

The base-collector diffusion capacitance is determined as follows:

\[
C_{bc_{diff}} = \frac{\partial}{\partial V_{bc}} (\tau_r \cdot i_{bc}) \quad (8.82)
\]

where

\[
i_{bc} = I_{S_{eff}} \left( \frac{V_{bc}}{e^\eta V_t} - 1 \right). \quad (8.83)
\]

T-Spice offers two different models for base-collector depletion capacitance. Use \texttt{dcap} to select a set of equations.

For \texttt{dcap=1}:

\[
C_{bc_{dep}} = X_{cje} \cdot C_{jc_{eff}} \left( 1 - \frac{V_{bc}}{V_{je}} \right)^{-m_{je}} \quad \text{when } V_{bc} < FC \cdot V_{je}
\]

\[
C_{bc_{dep}} = X_{cje} \cdot C_{jc_{eff}} \cdot \frac{1 - FC(1 + m_{je}) + m_{je} \cdot \frac{V_{bc}}{V_{je}}}{(1 - FC)^{1 + m_{je}}} \quad \text{when } V_{bc} \geq FC \cdot V_{je}
\]

For \texttt{dcap=2}:

\[
C_{bc_{dep}} = X_{cje} \cdot C_{jc_{eff}} \left( 1 - \frac{V_{bc}}{V_{je}} \right)^{-m_{je}} \quad \text{when } V_{bc} < 0
\]

\[
C_{bc_{dep}} = X_{cje} \cdot C_{jc_{eff}} \left( 1 + m_{je} \cdot \frac{V_{bc}}{V_{je}} \right) \quad \text{when } V_{bc} \geq 0
\]

where \(X_{cje}\) is the partition parameter specifying the ratio of base-collector junction capacitance distribution between internal base-internal collector and external base-internal collector.

If \(X_{cje}<1\), the external base-internal collector capacitance has to be considered:
Chapter 8: Device Models

BJT Level 1 (Gummel-Poon)

For \( dcap=1 \):

\[
C_{bcx} = (1-X_{cjc}) \cdot C_{jc,eff} \left( 1 - \frac{V_{bcx}}{V_{jc}} \right)^{-m_{jc}} \quad \text{when } V_{bcx} < FC \cdot V_{jc}
\]

\[
C_{bcx} = (1-X_{cjc}) \cdot C_{jc,eff} \cdot \frac{1 - FC(1+m_{jc}) + m_{jc} \cdot \frac{V_{bcx}}{V_{jc}}}{(1-FC)^{1+m_{jc}}} \quad \text{when } V_{bcx} \geq FC \cdot V_{jc}
\]

(8.86)

For \( dcap=2 \):

\[
C_{bcx} = (1-X_{cjc}) \cdot C_{jc,eff} \left( 1 - \frac{V_{bcx}}{V_{jc}} \right)^{-m_{jc}} \quad \text{when } V_{bcx} < 0
\]

\[
C_{bcx} = (1-X_{cjc}) \cdot C_{jc,eff} \left( 1 + m_{jc} \cdot \frac{V_{bcx}}{V_{jc}} \right) \quad \text{when } V_{bcx} \geq 0
\]

(8.87)

where \( V_{bcx} \) is the voltage between the external base node and the internal collector node.

Substrate Capacitance

The definition of substrate capacitance varies with BJT geometry. For vertical transistors (\( subs=1 \)), substrate capacitance is defined for the base to substrate diode. For lateral transistors (\( subs=-1 \)), it is defined as the capacitance of the substrate to collector diode.

For vertical BJTs,

\[
C_{sc} = C_{js,eff} \left( 1 - \frac{V_{sc}}{V_{js}} \right)^{-m_{js}} \quad \text{when } V_{sc} < 0
\]

(8.88)

\[
C_{sc} = C_{js,eff} \left( 1 + m_{js} \cdot \frac{V_{sc}}{V_{js}} \right) \quad \text{when } V_{sc} \geq 0
\]

For lateral BJTs,

\[
C_{bs} = C_{js,eff} \left( 1 - \frac{V_{bs}}{V_{js}} \right)^{-m_{js}} \quad \text{when } V_{bs} < 0
\]

(8.89)

\[
C_{bs} = C_{js,eff} \left( 1 + m_{js} \cdot \frac{V_{bs}}{V_{js}} \right) \quad \text{when } V_{bs} \geq 0
\]

Temperature Dependence

Below is a list of parameters that T-Spice modifies when one or more corresponding temperature coefficients are specified, regardless of \( \text{TLEV} \) values. When neither first nor second order coefficients
are specified for a given parameter, the TLEV-dependent equations in “Saturation Current and Beta” on page 389 take precedence.

Forward and reverse current gain

\[
\beta_f(T) = \beta_f' \cdot (1 + TBF_1 \cdot \Delta T + TBF_2 \cdot \Delta T^2) \\
\beta_r(T) = \beta_r' \cdot (1 + TBR_1 \cdot \Delta T + TBR_2 \cdot \Delta T^2)
\]

Voltage and current parameters

\[
V_{af}(T) = V_{af}' \cdot (1 + TVAF_1 \cdot \Delta T + TVAF_2 \cdot \Delta T^2) \\
V_{ar}(T) = V_{ar}' \cdot (1 + TVAR_1 \cdot \Delta T + TVAR_2 \cdot \Delta T^2) \\
I_f(T) = I_f' \cdot (1 + TITF_1 \cdot \Delta T + TITF_2 \cdot \Delta T^2)
\]

Transit time parameters

\[
\tau_f(T) = \tau_f' \cdot (1 + TTF_1 \cdot \Delta T + TTF_2 \cdot \Delta T^2) \\
\tau_r(T) = \tau_r' \cdot (1 + TTR_1 \cdot \Delta T + TTR_2 \cdot \Delta T^2)
\]

Emission coefficients

\[
\eta_f(T) = \eta_f' \cdot (1 + TNF_1 \cdot \Delta T + TNF_2 \cdot \Delta T^2) \\
\eta_r(T) = \eta_r' \cdot (1 + TNR_1 \cdot \Delta T + TNR_2 \cdot \Delta T^2) \\
\eta_e(T) = \eta_e' \cdot (1 + TNE_1 \cdot \Delta T + TNE_2 \cdot \Delta T^2) \\
\eta_k(T) = \eta_k' \cdot (1 + TNC_1 \cdot \Delta T + TNC_2 \cdot \Delta T^2) \\
\eta_s(T) = \eta_s' \cdot (1 + TNS_1 \cdot \Delta T + TNS_2 \cdot \Delta T^2)
\]
Energy Gap

The calculation of energy gap is dependent on **TLEV**. For **TLEV=0, 1, or 3**, energy gap is always calculated as follows:

\[
E_g(T_{nom}) = 1.16 - (7.02 \times 10^4) \frac{T_{nom}^2}{T_{nom} + 1108.0}.
\]  \hspace{1cm} (8.96)

If **TLEV=2**, the energy gap is calculated as a function of model parameters \(E_g(0), \text{GAP1}, \text{and GAP2}:

\[
E_g(T_{nom}) = E_g(0) - \text{GAP1} \cdot \frac{T_{nom}^2}{T_{nom} + \text{GAP2}}.
\]  \hspace{1cm} (8.97)

Saturation Current and Beta

The following equations show the temperature effects for transport saturation current, base-emitter reverse saturation current, and base-collector reverse saturation current.

**TLEV=0, 1, or 2**

\[
I_s(T) = I_s \cdot e^{facLN}
\]

\[
I_{be}(T) = \frac{facLN}{I_{be} \cdot e^{-\frac{\eta}{T}}}
\]

**TLEV=3**

\[
I_s(T) = I_s(1 + TIS_1 \cdot \Delta T + TIS_2 \cdot \Delta T^2)
\]

\[
I_{be}(T) = I_{be}(1 + TIS_1 \cdot \Delta T + TIS_2 \cdot \Delta T^2)
\]
Temperature-effect equations for leakage saturation currents, bulk-to-collector (or base) saturation current, and beta parameters are shown below.

**TLEV=0 or 2**

\[
I_{se}(T) = \frac{I_{se}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
I_{sc}(T) = \frac{I_{sc}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
I_{ss}(T) = \frac{I_{ss}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
\beta_f(T) = \beta_f \cdot \left( \frac{T}{T_{nom}} \right)^{X_{ib}}
\]

\[
\beta_r(T) = \beta_r \cdot \left( \frac{T}{T_{nom}} \right)^{X_{ib}}
\]

(if TBF₁, TBF₂ are not specified)

**TLEV=1**

\[
I_{bc}(T) = \frac{I_{bc}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
I_{kf}(T) = I_{kf} \cdot (1 + TIKF₁ΔT + TIKF₂ΔT^{2})
\]

\[
I_{kr}(T) = I_{kr} \cdot (1 + TKIR₁ΔT + TKIR₂ΔT^{2})
\]

\[
I_{rb}(T) = I_{rb} \cdot (1 + TIRB₁ΔT + TIRB₂ΔT^{2})
\]

**TLEV=3**

\[
I_{se}(T) = \frac{I_{se}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
I_{sc}(T) = \frac{I_{sc}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
I_{ss}(T) = \frac{I_{ss}}{T_{nom}} \cdot e^{\frac{\text{facLN}}{\eta_v}}
\]

\[
\beta_f(T) = \beta_f \cdot \left( \frac{T}{T_{nom}} \right)^{X_{ib}}
\]

\[
\beta_r(T) = \beta_r \cdot \left( \frac{T}{T_{nom}} \right)^{X_{ib}}
\]

(if TBR₁, TBR₂ are not specified)

For TLEV=0 and 1

\[
facln = \frac{E_g(0)}{V_i(T_{nom})} - \frac{E_g(0)}{V_i(T)} + XT_I \cdot \ln \left( \frac{T}{T_{nom}} \right)
\]

(8.98)

For TLEV=2

\[
facln = \frac{E_g(T_{nom})}{V_i(T_{nom})} - \frac{E_g(T)}{V_i(T)} + XT_I \cdot \ln \left( \frac{T}{T_{nom}} \right)
\]

(8.99)
For $\text{TLEV}=3$

\[
\text{facln} = \frac{V_{SB}}{V(T_{\text{nom}})} - \frac{V_{SB}}{V(T)} + \text{XTI} \cdot \ln \left( \frac{T}{T_{\text{nom}}} \right) \tag{8.100}
\]

**Note:** If $\text{TBF}_1$ or $\text{TBF}_2$ is specified, then $\beta_r$ will be calculated using the equation for $\text{Tlev}=3$, regardless of the actual value of $\text{Tlev}$. Similarly, the $\text{Tlev}=3$ equation for $\beta_f$ will always take precedence if either $\text{TBR}_1$ or $\text{TBR}_2$ is specified.

### Capacitance

Capacitance equations are selected by the parameter $\text{TLEVC}$.

The following capacitance equations are used when $\text{TLEVC}=0$:

\[
\begin{align*}
C_{je}(T) &= C_{je} \cdot \left\{ 1 + m_{je} \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{V_{je}(T)}{V_{je}} + 1 \right) \right\} \\
C_{jc}(T) &= C_{jc} \cdot \left\{ 1 + m_{jc} \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{V_{jc}(T)}{V_{jc}} + 1 \right) \right\} \\
C_{js}(T) &= C_{js} \cdot \left\{ 1 + m_{js} \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{V_{js}(T)}{V_{js}} + 1 \right) \right\}
\end{align*}
\tag{8.101}
\]

where contact voltages are determined by

\[
\begin{align*}
V_{je}(T) &= V_{je} \cdot \left( \frac{T}{T_{\text{nom}}} \right) - V_{l}(T) \cdot \left\{ 3 \ln \left( \frac{T}{T_{\text{nom}}} \right) + \frac{E_g(T_{\text{nom}})}{V(T_{\text{nom}})} - \frac{E_g(T)}{V(T)} \right\} \\
V_{jc}(T) &= V_{jc} \cdot \left( \frac{T}{T_{\text{nom}}} \right) - V_{l}(T) \cdot \left\{ 3 \ln \left( \frac{T}{T_{\text{nom}}} \right) + \frac{E_g(T_{\text{nom}})}{V(T_{\text{nom}})} - \frac{E_g(T)}{V(T)} \right\} \\
V_{js}(T) &= V_{js} \cdot \left( \frac{T}{T_{\text{nom}}} \right) - V_{l}(T) \cdot \left\{ 3 \ln \left( \frac{T}{T_{\text{nom}}} \right) + \frac{E_g(T_{\text{nom}})}{V(T_{\text{nom}})} - \frac{E_g(T)}{V(T)} \right\}
\end{align*}
\tag{8.102}
\]

If $\text{TLEVC}=1$,

\[
\begin{align*}
C_{je}(T) &= C_{je} \cdot (1 + C_{ie} \cdot \Delta T) \\
C_{jc}(T) &= C_{jc} \cdot (1 + C_{ic} \cdot \Delta T) \\
C_{js}(T) &= C_{js} \cdot (1 + C_{is} \cdot \Delta T)
\end{align*}
\tag{8.103}
\]

and contact voltages are determined by

\[
\begin{align*}
V_{je}(T) &= V_{je} - T_{vje} \cdot \Delta T \\
V_{jc}(T) &= V_{jc} - T_{vjc} \cdot \Delta T \\
V_{js}(T) &= V_{js} - T_{vjs} \cdot \Delta T
\end{align*}
\tag{8.104}
\]
If TLEVC=2,

\[
C_{je}(T) = C_{je} \cdot \left( \frac{V_{je}}{V_{je}(T)} \right)^{m_{je}}
\]
\[
C_{jc}(T) = C_{jc} \cdot \left( \frac{V_{jc}}{V_{jc}(T)} \right)^{m_{jc}}
\]
\[
C_{js}(T) = C_{js} \cdot \left( \frac{V_{js}}{V_{js}(T)} \right)^{m_{js}}
\]

where

\[
V_{je}(T) = V_{je} - T_{vje} \cdot \Delta T
\]
\[
V_{jc}(T) = V_{jc} - T_{vjc} \cdot \Delta T
\]
\[
V_{js}(T) = V_{js} - T_{vjs} \cdot \Delta T
\]

**Note:** Use \( m_{je}, m_{jc}, \) and \( m_{js} \) instead of \( m_{je}(T), m_{jc}(T), \) and \( m_{js}(T) \) in the above equations.

When TLEVC=3:

\[
C_{je}(T) = C_{je} \cdot \left( 1 - 0.5 \cdot \frac{dvjedt}{V_{je}} \cdot \frac{\Delta T}{V_{je}} \right)
\]
\[
C_{jc}(T) = C_{jc} \cdot \left( 1 - 0.5 \cdot \frac{dvjcdt}{V_{jc}} \cdot \frac{\Delta T}{V_{jc}} \right)
\]
\[
C_{js}(T) = C_{js} \cdot \left( 1 - 0.5 \cdot \frac{dvjsdt}{V_{js}} \cdot \frac{\Delta T}{V_{js}} \right)
\]

and

\[
V_{je}(T) = V_{je} + dvjedt \cdot \Delta T
\]
\[
V_{jc}(T) = V_{jc} + dvjcdt \cdot \Delta T
\]
\[
V_{js}(T) = V_{js} + dvjsdt \cdot \Delta T
\]

where

\[
dvjedt = -\frac{E_g(T_{nom}) + 3 \cdot V_i(T_{nom}) + [E_g(0) - E_g(T_{nom})] \cdot \left( 2 - \frac{T_{nom}}{GAP^2 + T_{nom}} \right) - V_{je}}{T_{nom}}
\]
\[
dvjcdt = -\frac{E_g(T_{nom}) + 3 \cdot V_i(T_{nom}) + [E_g(0) - E_g(T_{nom})] \cdot \left( 2 - \frac{T_{nom}}{GAP^2 + T_{nom}} \right) - V_{jc}}{T_{nom}}
\]
\[
dvjsdt = -\frac{E_g(T_{nom}) + 3 \cdot V_i(T_{nom}) + [E_g(0) - E_g(T_{nom})] \cdot \left( 2 - \frac{T_{nom}}{GAP^2 + T_{nom}} \right) - V_{js}}{T_{nom}}
\]
BJT Level 6 (Mextram)

The level 6 bipolar junction transistor model is the Philips NXP Mextram model.

Parameters

The Mextram model uses the following syntax.

```
.model name npn|pnp level=[6|503|504] | model=modelname [parameters]
```

The Mextram model is fully described in the document Bipolar Transistor Level 504. For additional detailed information about the Mextram model, please refer to the Philips NXP Compact Model web page:

http://www.nxp.com/models/bi_models/mextram/

T-Spice includes support for Mextram versions 503 and 504. The version 504 model provides optional support for modeling the extrinsic substrate and the self-heating effects. Selection of an appropriate model is accomplished by using the `model=modelname` parameter. If no specific model is specified via the `model` parameter, the `bjt503` model will be selected when `level=503` or `version=503`, otherwise the `bjt504` model will be used.

The available `modelname` values for the Mextram model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bjt503</td>
<td>Mextram version 503 with substrate</td>
</tr>
<tr>
<td>bjd503</td>
<td>Mextram version 503 without substrate</td>
</tr>
<tr>
<td>bjt504 (default)</td>
<td>Mextram version 504 with substrate</td>
</tr>
<tr>
<td>bjd504</td>
<td>Mextram version 504 without substrate (discrete)</td>
</tr>
<tr>
<td>bjt504t</td>
<td>Mextram version 504 with substrate, self-heating temperature</td>
</tr>
<tr>
<td>bjd504t</td>
<td>Mextram version 504 without substrate, self-heating</td>
</tr>
</tbody>
</table>
BJT Level 9 (VBIC)

In addition to the standard Gummel-Poon BJT model, T-Spice supports the Vertical Bipolar Inter-Company model (VBIC) as the level 9 BJT.

Note: If you specify VBIC Level 4 in T-Spice it will use the VBIC 1999 version 1.2 model.

Parameters

The VBIC model uses the following syntax.

```.model name npn | pnp level=9 [parameters]```

The T-Spice VBIC model is based upon release 1.2 of the VBIC code, and includes certain enhancements for improved convergence and stability.

The following tables describe all of the VBIC BJT parameter.

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>afn</td>
<td>base-emitter flicker noise exponent</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>ajc</td>
<td>base-collector capacitance smoothing factor</td>
<td>-.5</td>
<td></td>
</tr>
<tr>
<td>aje</td>
<td>base-emitter capacitance smoothing factor</td>
<td>-.5</td>
<td></td>
</tr>
<tr>
<td>ajs</td>
<td>substrate-collector capacitance smoothing factor</td>
<td>-.5</td>
<td></td>
</tr>
<tr>
<td>art</td>
<td>base-collector reach-through limiting voltage (0 means infinity)</td>
<td>0.1</td>
<td>V</td>
</tr>
<tr>
<td>avc1</td>
<td>base-collector weak avalanche parameter 1</td>
<td>0.0</td>
<td>1/V</td>
</tr>
<tr>
<td>avc2</td>
<td>base-collector weak avalanche parameter 2</td>
<td>0.0</td>
<td>1/V</td>
</tr>
<tr>
<td>bfn</td>
<td>base-emitter flicker noise 1/f dependence</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>cbco (cbe0)</td>
<td>extrinsic base-collector overlap capacitance</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>cbeo (cbe0)</td>
<td>extrinsic base-emitter overlap capacitance</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>ccso</td>
<td>Fixed collector-substrate capacitance</td>
<td>0</td>
<td>C</td>
</tr>
<tr>
<td>cjc</td>
<td>base-collector intrinsic zero bias capacitance</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>cjcp</td>
<td>substrate-collector zero bias capacitance</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>cje</td>
<td>base-emitter zero bias capacitance</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>cjep</td>
<td>base-collector extrinsic zero bias capacitance</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>cth</td>
<td>thermal capacitance</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>dear</td>
<td>Activation energy shift for ISRR (HBTs)</td>
<td>0</td>
<td>V</td>
</tr>
</tbody>
</table>
## VBIC Model Parameters

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>dtemp (dtmp)</td>
<td>local temperature rise</td>
<td>0</td>
<td>deg C</td>
</tr>
<tr>
<td>ea</td>
<td>activation energy for IS</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eaic</td>
<td>activation energy for IBCI/IBEIP</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eaie</td>
<td>activation energy for IBEI</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eaic</td>
<td>activation energy for IBEI</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eanc</td>
<td>activation energy for IBCN/IBENP</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eane</td>
<td>activation energy for IBEN</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eans</td>
<td>activation energy for IBCNP</td>
<td>1.12</td>
<td>eV</td>
</tr>
<tr>
<td>eap</td>
<td>Activation energy for ISP</td>
<td>1.12</td>
<td>V</td>
</tr>
<tr>
<td>fc</td>
<td>forward bias depletion capacitance limit</td>
<td>0.9</td>
<td></td>
</tr>
<tr>
<td>gamm (gamma)</td>
<td>epi doping parameter</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>hrcf</td>
<td>high current RC factor</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>ibbe</td>
<td>Base-Emitter breakdown current</td>
<td>1e-6</td>
<td></td>
</tr>
<tr>
<td>ibci</td>
<td>ideal base-collector saturation current</td>
<td>1.0E-16</td>
<td>A</td>
</tr>
<tr>
<td>ibcip</td>
<td>ideal parasitic base-collector saturation current</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ibcn</td>
<td>non-ideal base-collector saturation current</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ibcnp</td>
<td>non-ideal parasitic base-collector saturation current</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ibei</td>
<td>ideal base-emitter saturation current</td>
<td>1.0E-18</td>
<td>A</td>
</tr>
<tr>
<td>ibeip</td>
<td>ideal parasitic base-emitter saturation current</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>iben</td>
<td>non-ideal base-emitter saturation current</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>ibenp</td>
<td>non-ideal parasitic base-emitter saturation current</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ikf</td>
<td>forward knee current (zero means infinity)</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ikp</td>
<td>parasitic knee current (zero means infinity)</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>ikr</td>
<td>reverse knee current (zero means infinity)</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>is</td>
<td>transport saturation current</td>
<td>1.0E-16</td>
<td>Amps</td>
</tr>
<tr>
<td>isp</td>
<td>parasitic transport saturation current</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>isrr</td>
<td>Reverse saturation current factor (HBTs)</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>itf</td>
<td>coefficient of TF dependence in Ic</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>knf</td>
<td>base-emitter flicker noise constant</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>mc</td>
<td>base-collector grading coefficient</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>me</td>
<td>base-emitter grading coefficient</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>Parameter (alias)</td>
<td>Description</td>
<td>Default</td>
<td>Units</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------------------------------------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>ms</td>
<td>substrate-collector grading coefficient</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>nbbe</td>
<td>Base-Emitter breakdown emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>nei</td>
<td>ideal base-collector emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>ncip</td>
<td>ideal parasitic base-collector emission coefficient</td>
<td>1.0</td>
<td>A</td>
</tr>
<tr>
<td>ncn</td>
<td>non-ideal base-collector emission coefficient</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>ncnp</td>
<td>non-ideal parasitic base-collector emission coefficient</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>nei</td>
<td>ideal base-emitter emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>nen</td>
<td>non-ideal base-emitter emission coefficient</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>nf</td>
<td>forward emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>nfp</td>
<td>parasitic fwd emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>nkf</td>
<td>High current roll-off coefficient</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>nr</td>
<td>reverse emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>pc</td>
<td>base-collector built-in potential</td>
<td>0.75</td>
<td>V</td>
</tr>
<tr>
<td>pe</td>
<td>base-emitter built-in potential</td>
<td>0.75</td>
<td>V</td>
</tr>
<tr>
<td>ps</td>
<td>substrate-collector built-in potential</td>
<td>0.75</td>
<td>V</td>
</tr>
<tr>
<td>qbm</td>
<td>Selector for SGP qb formulation</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>qco (qc0)</td>
<td>epi charge parameter</td>
<td>0.0</td>
<td>C</td>
</tr>
<tr>
<td>qtf</td>
<td>variation of TF with base-width modulation</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>rbi</td>
<td>intrinsic base resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rbp</td>
<td>parasitic base resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rbx</td>
<td>extrinsic base resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rci</td>
<td>intrinsic collector resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rcx</td>
<td>extrinsic collector resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>re</td>
<td>emitter resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rs</td>
<td>substrate resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rth</td>
<td>thermal resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>tavc</td>
<td>temperature coefficient of AVC2</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>td</td>
<td>forward excess-phase delay time</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>tf</td>
<td>forward transit time</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>tnbbe</td>
<td>Temperature coefficient of NBBE</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>tnf</td>
<td>temperature coefficient of NF</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>
### VBIC Model Parameters

<table>
<thead>
<tr>
<th>Parameter (alias)</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>tnom (tref)</td>
<td>nominal measurement temperature of parameters</td>
<td>global tnom (25.0)</td>
<td>deg. C</td>
</tr>
<tr>
<td>tr</td>
<td>reverse transit time</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>tvbbe1</td>
<td>First temperature coefficient of VBBE</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>tvbbe2</td>
<td>Second temperature coefficient of VBBE</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>vbbe</td>
<td>Base-Emitter breakdown voltage (zero means infinity)</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>vef</td>
<td>forward Early voltage (zero means infinity)</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>ver</td>
<td>reverse Early voltage (zero means infinity)</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>vers</td>
<td>Version</td>
<td>1.2</td>
<td></td>
</tr>
<tr>
<td>vo (v0)</td>
<td>epi drift saturation voltage</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>vrev</td>
<td>Revision</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>vrt</td>
<td>base-collector reach-through limiting voltage (0 means infinity)</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>vtf</td>
<td>coefficient of TF dependence on Vbc</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>wbe</td>
<td>portion of IBEI from Vbei (1-WBE from Vbex)</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>wsp</td>
<td>portion of ICCP from Vbep (1-WSP from Vbci)</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>xii</td>
<td>temperature exponent of IBEI/IBC/IBEIP/IBCIP</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>xikf</td>
<td>Temperature exponent of IKF</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>xin</td>
<td>temperature exponent of IBEN/IBC/IBENP/IBCNP</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>xis</td>
<td>temperature exponent of IS</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>xisr</td>
<td>Temperature exponent of ISRR (HBTs)</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>xrb (xrbi)</td>
<td>temperature exponent of base resistance</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>xrbp</td>
<td>Temperature exponent of extrinsic resistance RBP</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>xrbx</td>
<td>Temperature exponent of extrinsic resistance RBX</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>xrc (xrci)</td>
<td>temperature exponent of collector resistance</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>xrcx</td>
<td>Temperature exponent of extrinsic resistance RCX</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>xre</td>
<td>temperature exponent of emitter resistance</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>xrs</td>
<td>temperature exponent of substrate resistance</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>xtf</td>
<td>coefficient of TF bias dependence</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>xvo (xv0)</td>
<td>temperature exponent of VO</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>
The following VBIC model parameters are T-Spice extensions to the standard model.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>model selector</td>
<td>4 V</td>
</tr>
<tr>
<td>ismin</td>
<td>Minimum permissible value of IS</td>
<td>1e-19 A</td>
</tr>
<tr>
<td>ispmin</td>
<td>Minimum permissible value of ISP</td>
<td>1e-19 A</td>
</tr>
<tr>
<td>mcmin</td>
<td>Minimum permissible value of MC</td>
<td>.01</td>
</tr>
<tr>
<td>memin</td>
<td>Minimum permissible value of ME</td>
<td>.01</td>
</tr>
<tr>
<td>msmin</td>
<td>Minimum permissible value of MS</td>
<td>.01</td>
</tr>
<tr>
<td>rbpmin</td>
<td>Minimum permissible value of RBP</td>
<td>.001 Ω</td>
</tr>
</tbody>
</table>

Equations

For a thorough description of the VBIC model and equations, please refer to the following website:

http://www.designers-guide.org/VBIC/index.html

Note:

Please keep in mind that the default operating temperature and model reference temperature in T-Spice is 25 degrees Celsius, whereas many simulators use 27 degrees. When performing simulation comparisons against these simulators, you may want to set the default reference temperature option, “tnom” (page 301), and the operating temperature, “.temp” (page 163), to 27.
BJT Level 10 (Modella)

The level 10 bipolar junction transistor model is the Philips Modella model, a lateral PNP bipolar model.

Parameters

The Modella model uses the following syntax.

```
.model name pnp level=[10|500] | model=modelname [parameters]
```

The complete user manual for the Modella model is located at Bipolar PNP Transistor Level 500. For additional detailed information about the Modella model, please refer to the Philips NXP Compact Model web page:

http://www.nxp.com/models/bi_models/modella/index.html

T-Spice includes support for the Modella model version 500 with and without self-heating effects. When the self-heating model is used, the device statement should include an additional temperature node following the substrate node name.

The available `modelname` values for the Modella model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bjt500 (default)</td>
<td>Modella version 500 with substrate</td>
</tr>
<tr>
<td>bjt500t</td>
<td>Mextram version 500 with substrate, self-heating</td>
</tr>
</tbody>
</table>
Capacitor

Indicates the capacitance of a planar diffused region from geometric and process information.

Parameters

\[
.model \, modelname \, c \, [parameter \, =X]
\]

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cap</td>
<td>( C_0 )</td>
<td>Capacitance</td>
<td>0</td>
<td>F</td>
</tr>
<tr>
<td>Capsw</td>
<td>( C_{sw} )</td>
<td>Sidewall capacitance</td>
<td>0</td>
<td>F/m</td>
</tr>
<tr>
<td>Cox</td>
<td>( C_{ox} )</td>
<td>Bottomwall capacitance</td>
<td>0</td>
<td>F/m^2</td>
</tr>
<tr>
<td>Del</td>
<td>( \delta )</td>
<td>Difference between drawn and actual widths or lengths</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>Di</td>
<td>( \kappa )</td>
<td>Dielectric constant or relative permittivity</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>L</td>
<td>( L )</td>
<td>Length of the capacitor</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>Shrink</td>
<td>( S_{shrink} )</td>
<td>Shrink factor</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>Tc1</td>
<td>( T_{c1} )</td>
<td>The first temperature coefficient for capacitance</td>
<td>0</td>
<td>1/deg C</td>
</tr>
<tr>
<td>Tc2</td>
<td>( T_{c2} )</td>
<td>The second temperature coefficient for capacitance</td>
<td>0</td>
<td>1/(deg C)^2</td>
</tr>
<tr>
<td>Thick</td>
<td>( \tau )</td>
<td>Insulator thickness</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>Tnom</td>
<td>( T_{nom} )</td>
<td>Reference temperature</td>
<td>global</td>
<td>t_{nom} (25.0)</td>
</tr>
<tr>
<td>W</td>
<td>( W )</td>
<td>Width of capacitor</td>
<td>0</td>
<td>m</td>
</tr>
</tbody>
</table>

Equations

The statement “.model” (page 116) must be associated with an element statement; see the third syntax example in “Capacitor (c)” (page 174).

\[
C = MS_{scale} \left[1 + T_{c1} \Delta T + T_{c2} (\Delta T)^2 \right] C_0
\]  

(8.110)

**Note:**

When the calculated capacitance is greater than 0.1 F, T-Spice issues a warning message.

The user supplies the multiplicity factor \( M \) and the scale factor \( Scale \) in the element statement. See “Capacitor” (page 400). The user can supply \( T_{c1}, T_{c2}, \) and \( D_{temp} \) in the element statement, model statement, or both. In the last case, element values override model values. \( C_0 \) will be determined in one of three ways, which are listed in order of selection:

1. \( C_0 \) in the element statement.
2. \( C_{0x} \) in the model statement; \( C_0 \) will be computed using equation 0.111.
3. \( C_0 \) in the model statement.
When \(C_{0x}\) is supplied in the model description, T-Spice would then compute \(C_0\) as follows:

\[
C_0 = L_{\text{eff}}W_{\text{eff}}C_{ox} + 2(L_{\text{eff}} + W_{\text{eff}})C_{sw}
\]

(8.111)

where

\[
L_{\text{eff}} = L_{\text{scaled}} - 2\Delta_{\text{eff}}
\]

(8.112)

\[
W_{\text{eff}} = W_{\text{scaled}} - 2\Delta_{\text{eff}}
\]

(8.113)

If \(L\) is supplied in the element statement:

\[
L_{\text{scaled}} = L_{\text{shrink}}S'_{\text{scale}}
\]

(8.114)

If \(L\) is supplied in the model statement:

\[
L_{\text{scaled}} = L_{\text{shrink}}S'_{\text{scalr}}
\]

(8.115)

If \(W\) is supplied in the element statement:

\[
W_{\text{scaled}} = W_{\text{shrink}}S'_{\text{scale}}
\]

(8.116)

If \(W\) is supplied in the model statement:

\[
W_{\text{scaled}} = W_{\text{shrink}}S'_{\text{scalr}}
\]

(8.117)

The user supplies \(S'_{\text{scale}}\) and \(S'_{\text{scalr}}\) using \texttt{.options scale} or \texttt{.options scalm}. For further information, see \texttt{“.options” (page 125)}.

\[
\Delta = \delta S'_{\text{scalr}}
\]

(8.118)

If \(\tau\) is supplied, T-Spice would compute \(C_{ox}\) from

\[
\text{If } \kappa \neq 0: C_{ox} = \kappa (e_0/\tau)
\]

(8.119)

\[
\text{If } \kappa = 0: C_{ox} = e_0/\tau
\]

(8.120)

The quantities \(e_0, e_0x\) are 8.8542149e-12 and 3.453148e-11, respectively. After this, T-Spice would compute \(C0\) using the formulas above. If \(\tau\) is not supplied, T-Spice sets \(C0 = 0\).

**Examples**

```
V1 a d AC 150 0
R1 a b 10
L1 b c 50
.model capxx c
+ c ox=0.25
+ capsw=1/3.0
+ del=0.1
+ shrink=0.5
C1 c d capxx
+ scale= 0.02/2.5
  l = 2*1.2/27.7
  w = 2*2.2/27.7
.options scale = 27.7
.AC LIN 1 0.5/3.14159 0.5/3.14159
.PRINT AC IM(V1) IP(V1)
```
Coupled Transmission Line (Level 1)

The coupled transmission line model employs variable electrical parameters.

Parameters

```
.model name cpl level=1 [[r]={(matrix)}] [l]={(matrix)} [c]={(matrix)}
    [g]={(matrix)}
```

Matrices are entered as follows.

```
[r]={(r11, r12, r13, ... + {r21, r22, r23, ... + {r31, r32, r33, ... ...
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>[r]</td>
<td>Optional per unit length resistance matrix. Must be positive definite.</td>
<td>[Zero matrix]</td>
<td>Ω/m</td>
</tr>
<tr>
<td>[l]</td>
<td>Per unit length inductance matrix. Must be positive definite. Off-diagonal elements must be non-negative. n and p metric prefixes may be used.</td>
<td>[Zero matrix]</td>
<td>H/m</td>
</tr>
<tr>
<td>[c]</td>
<td>Per unit length capacitance matrix. Must be zero or negative. n and p metric prefixes may be used.</td>
<td>[Zero matrix]</td>
<td>F/m</td>
</tr>
<tr>
<td>[g]</td>
<td>Optional per unit length conductance matrix. Must be positive definite.</td>
<td>[Zero matrix]</td>
<td>S/m</td>
</tr>
</tbody>
</table>

Example

An example using lossless symmetrical coupled lines:

```
.model exCPL CPL level=1
+ [l]={(494.6n, 63,3n, 63.3n, 494.6n )
+ [c]={(62.8p, -4.94p, -4.94p, 62.8p )
```
### Diode

There are five types of diode models in T-Spice.

- **Level 1** describes the Non-Geometric Junction diode model. It is used to model discrete diode devices such as standard and Zener diodes.
- **Level 2** describes the Fowler-Nordheim model that is generally used to characterize the tunneling current flow through thin insulator in nonvolatile memory devices.
- **Level 3** describes the Geometric Junction diode model. It is used to model IC-based standard silicone-diffused diodes, Schottky barrier diodes and Zener diodes.
- **Level 4** describes the Philips Juncap version 2 diode. Note - level 200 and level 9 are also the Juncap model, with different level numbers provided for compatibility with various simulators.
- **Level 500** describes the Philips Advanced diode.

### Parameters

The Philips Juncap2 model is fully described in the document *Philips Juncap2 level 200.1*.

The Philips Advanced Diode is fully described in the document *Philips Diode Model - Level 500*.

The following model parameter and equation descriptions pertain to Diode levels 1-3.

```
.model name d [parameters]
```

#### Model Selectors

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>level</td>
<td>Diode model selector</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>dcap</td>
<td>dcap</td>
<td>Capacitance model selector</td>
<td>2</td>
<td>—</td>
</tr>
<tr>
<td>tlev</td>
<td>tlev</td>
<td>Temperature equation selector ((tlev=3 \text{ for Schottky barrier diode}))</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>tlevc</td>
<td>tlevc</td>
<td>Temperature equation selector for junction capacitance and contact potential ((tlevc=1 \text{ and } 2 \text{ for Schottky barrier diode}))</td>
<td>0</td>
<td>—</td>
</tr>
</tbody>
</table>

#### Geometric and Scaling Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>area</td>
<td>area</td>
<td>Junction area (unitless in level=1)</td>
<td>1.0</td>
<td>m^2</td>
</tr>
<tr>
<td>PJ</td>
<td>PJ</td>
<td>Junction periphery (unitless in level=1)</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>M</td>
<td>M</td>
<td>Multiplier factor to simulate multiple diode in parallel</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>SCALE</td>
<td>SCALE</td>
<td>Scaling factor for geometric parameters</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>SCALM</td>
<td>SCALM</td>
<td>Scaling factor for model parameters</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
### DC Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS</td>
<td>JS</td>
<td>IS</td>
<td>Saturation current (per unit area). Use calculated IS value for Schottky barrier diode if IS is not specified; see “Application Notes—Schottky Barrier Diodes” on page 415.</td>
<td>$1.0 \times 10^{-14}$</td>
</tr>
<tr>
<td>JSW</td>
<td>JSW</td>
<td>Side wall saturation current (per unit length)</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>RS</td>
<td>RS</td>
<td>Ohmic resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>N</td>
<td>N</td>
<td>Emission coefficient ($N=1$ for Schottky barrier diode)</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>BV</td>
<td>VR</td>
<td>BV</td>
<td>Breakdown voltage</td>
<td>$\infty$</td>
</tr>
<tr>
<td>IBV</td>
<td>IB</td>
<td>IBV</td>
<td>Current at breakdown voltage</td>
<td>$1.0 \times 10^{-3}$</td>
</tr>
<tr>
<td>IKF</td>
<td>IKF</td>
<td>Forward knee current (per unit area)</td>
<td>0.0</td>
<td>A</td>
</tr>
<tr>
<td>IKB</td>
<td>IB</td>
<td>IKB</td>
<td>Reverse knee current (per unit area)</td>
<td>0.0</td>
</tr>
<tr>
<td>EXPLI</td>
<td>EXPLI</td>
<td>Current explosion model parameter. The PN junction characteristics above the explosion current area are linear, with the slope at the explosion point.</td>
<td>$1.0 \times 10^{-15}$</td>
<td>A</td>
</tr>
</tbody>
</table>

### Capacitance Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>CJ0</td>
<td>CJ</td>
<td>Cj0</td>
<td>Zero-bias bottom wall junction capacitance (per unit area)</td>
<td>0.0</td>
</tr>
</tbody>
</table>
### Noise Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>AF</td>
<td>Flick noise exponent</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>KF</td>
<td>KF</td>
<td>Flick noise exponent</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

### Temperature Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>TNOM</td>
<td>Tnom</td>
<td>Reference temperature</td>
<td>global <strong>tnom</strong> (25.0)</td>
<td>deg</td>
</tr>
<tr>
<td>EG</td>
<td>Eg(0)</td>
<td>Energy gap at 0°C</td>
<td>1.16</td>
<td>eV</td>
</tr>
<tr>
<td>VSB</td>
<td>VSB</td>
<td>Schottky barrier height (<strong>tlev=3</strong> only)</td>
<td>0.8</td>
<td>V</td>
</tr>
<tr>
<td>GAP1</td>
<td>GAP1</td>
<td>Coefficient in energy gap temperature equation (Si: 4.73×10⁻⁴, Ge: 4.77×10⁻⁴, and GaAs: 5.41×10⁻⁴)</td>
<td>7.02×10⁻⁴</td>
<td>eV/deg</td>
</tr>
<tr>
<td>GAP2</td>
<td>GAP2</td>
<td>Coefficient in energy gap temperature equation (Si: 636, Ge: 235, and GaAs: 204)</td>
<td>1108</td>
<td>deg</td>
</tr>
<tr>
<td>TCV</td>
<td>TCV</td>
<td>Breakdown voltage temperature coefficient</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
</tbody>
</table>
Chapter 8: Device Models

Diode

Fowler-Nordheim Model Parameters (level=2)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>TTT1</td>
<td>τt1</td>
<td>First-order temperature coefficient for τ</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>TTT2</td>
<td>τt2</td>
<td>Second-order temperature coefficient for τ</td>
<td>0.0</td>
<td>1/deg^2</td>
</tr>
<tr>
<td>TPB</td>
<td>TPB</td>
<td>Temperature coefficient for PB</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>TPHP</td>
<td>TPHP</td>
<td>Temperature coefficient for PHP</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>CTA</td>
<td>CTC</td>
<td>CTA</td>
<td>Temperature coefficient for CJ</td>
<td>0.0</td>
</tr>
<tr>
<td>CTP</td>
<td>CTP</td>
<td>Temperature coefficient for CJP</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>TM1</td>
<td>TM1</td>
<td>First order temperature coefficient for mj (TM=1 for Schottky barrier diode)</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>TM2</td>
<td>TM2</td>
<td>Second order temperature coefficient for mjsw (TM=2 for Schottky barrier diode)</td>
<td>0.0</td>
<td>1/deg^2</td>
</tr>
<tr>
<td>TRS</td>
<td>TRS</td>
<td>Resistance temperature coefficient</td>
<td>0.0</td>
<td>1/deg</td>
</tr>
<tr>
<td>XTI</td>
<td>XTI</td>
<td>Saturation current temperature exponent (for Schottky barrier diode)</td>
<td>3.0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>EF</td>
<td>EF</td>
<td>Forward critical electric field</td>
<td>1.0×10^8</td>
<td>V/cm</td>
</tr>
<tr>
<td>ER</td>
<td>ER</td>
<td>Reverse critical electric field</td>
<td>EF</td>
<td>V/cm</td>
</tr>
<tr>
<td>JF</td>
<td>JF</td>
<td>Forward Fowler-Nordheim current coefficient</td>
<td>1.0×10^-10</td>
<td>A/V^2</td>
</tr>
<tr>
<td>JR</td>
<td>JR</td>
<td>Reverse Fowler-Nordheim current coefficient</td>
<td>JF</td>
<td>A/V^2</td>
</tr>
<tr>
<td>TOX</td>
<td>τox</td>
<td>Thickness of oxide layer</td>
<td>100.0</td>
<td>Å</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>Length of diode</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>W</td>
<td>W</td>
<td>Width of diode</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>XW</td>
<td>XW</td>
<td>Masking and etching effects</td>
<td>0.0</td>
<td>m</td>
</tr>
</tbody>
</table>

Current Equations

Level 1 and Level 3

In levels 1 and 3, the diodes are modeled in forward bias, reverse bias, and breakdown regions.

In forward and reverse bias regions: \( Vd > -BV_{eff} \),

\[
I_d = I_{seff} \left( \exp\left(\frac{V_d}{N \cdot V_t} - 1\right) \right) \quad (8.121)
\]
where where $V_t$ is thermal voltage, $V_t = kT_{nom}/q$, $V_d$ is the voltage across the diode, $V_d = V_{node1} - V_{node2}$, $BVe_{ff}$ is the adjusted breakdown voltage, and $N$ is emission coefficient ($N=1$ for Schottky barrier diode).

$BVe_{ff}$ can be determined by

$$BVe_{ff} = BV - (N \cdot V_t \cdot \ln(I_{BVe_{ff}}))$$

(8.122)

when $I_{BVe_{ff}}>I_{break}$.

and

$$I_{BVe_{ff}} = BV, \text{ when } I_{BVe_{ff}} \leq I_{break}.$$  

(8.123)

where

$$I_{break} = -I_{seff} \cdot \left(\exp\left(\frac{BV}{N \cdot V_t} - 1\right)\right)$$

(8.124)

In breakdown region: $BV < BVe_{ff}$

$$I_d = I_{seff} \cdot \left(\exp\left(\frac{V_d + BV}{N \cdot V_t}\right)\right)$$

(8.125)

In all the above equations we assume that the diode has a finite breakdown voltage, that is $BV \neq \infty$. When $BV$ is not given, or the diode has an infinite breakdown voltage,

$$I_d = I_{seff} \cdot \left(\exp\left(\frac{V_d}{N \cdot V_t} - 1\right)\right)$$

(8.126)

$V_d>0$ is the forward-bias region and $V_d<0$ is the reverse-bias region. There is no breakdown region in this case.

If the high-level injection effects are considered, the current equations are

in forward-bias region ($V_d>0$):

$$I^*_{d} = \frac{I_d}{1 + \left(\frac{I_d}{I_{KF}}\right)^2}$$

(8.127)

when $IKF \neq 0$.

$$I^*D = ID$$

(8.128)

when $IKF = 0$. 


and in reverse-bias region \((V_d < 0)\)

\[
P_d^* = \frac{I_d}{1 + \left(\frac{I_d}{I_{KR}}\right)^2}
\]

(8.129)

when \(I_{KR} \neq 0\).

\[
I^*D = ID
\]

(8.130)

when \(I_{KR} = 0\).

### Diode Capacitance Equations

The diode capacitance \(C_d\) consists of the contributions from diffusion capacitance \(C_d\), junction capacitance (depletion capacitance) \(C_j\), metal (contact electrode) capacitance \(C_m\), and polysilicon (contact electrode) capacitance \(C_p\).

#### Diffusion Capacitance

\[
C_d = \pi \cdot \frac{\partial I_d}{\partial V_d}
\]

(8.131)

**Note:** For Schottky barrier diodes, \(C_d = 0\) \((\pi = 0)\) because the Schottky barrier diode is a majority carrier device and minority carrier effect can be ignored.

#### Junction Capacitance (Depletion Capacitance)

The junction capacitance has two components: the junction bottom area capacitance \(C_{ja}\) and the junction periphery capacitance \(C_{jp}\).

\[
C_j = C_{ja} + C_{jp}
\]

There are two sets of junction capacitance equations selected by the parameter \(d_{cap}\).

For \(d_{cap}=1\), the junction bottom area capacitance is given by

\[
C_{ja} = C_{j0eff} \cdot \left(1 - \frac{V_d}{P_B}\right)^{-m_j}
\]

(8.132)

when \(V_d > FC \times PB\)

\[
C_{ja} = C_{j0eff} \cdot \frac{1 - FC \times (1 + m_j) + m_j \times \frac{V_d}{P_B}}{(1 - FC)^{1 + m_j}}
\]

(8.133)

when \(V_d > FC \times PB\)
The junction periphery capacitance is given by

\[ C_{jp} = C_{j0eff} \cdot \left( 1 - \frac{V_d}{PHP} \right)^{-m_{jw}} \]  

(8.134)

when \( V_d > FCS \times PHP \)

\[ C_{ja} = C_{j0eff} \cdot \frac{1 - FCS \times (1 + m_{jsw}) + m_{jsw} \times \frac{V_d}{PB}}{(1 - FCS)^{1 + m_{jw}}} \]  

(8.135)

when \( V_d > FCS \times PHP \)

The total junction capacitance is

\[ C_j = C_{ja} + C_{jp} \]  

(8.136)

For \texttt{dcap=2} (default), the total junction capacitance is given by

\[ C_j = C_{j0eff} \cdot \left( 1 - \frac{V_d}{PB} \right)^{-m_j} + C_{j0eff} \cdot \left( 1 - \frac{V_d}{PHP} \right)^{-m_{jw}} \]  

(8.137)

when \( V_d < 0 \).

\[ C_j = C_{j0eff} \cdot \left( 1 + m_j \cdot \frac{V_d}{PB} \right) + C_{j0eff} \cdot \left( 1 + m_{jsw} \cdot \frac{V_d}{PHP} \right) \]  

(8.138)

when \( V_d > 0 \).

\textbf{Metal (Contact Electrode) Capacitance}

For \texttt{level=3} only

\[ C_m = \frac{\varepsilon_{ox}}{X_{OM}} \cdot (W_{Meff} + X_{Meff}) \cdot (L_{Meff} + X_{Meff}) \]  

(8.139)

\textbf{Polysilicon (Contact Electrode) Capacitance}

For \texttt{level=3} only

\[ C_p = \frac{\varepsilon_{ox}}{X_{DP}} \cdot (W_{Peff} + X_{Peff}) \cdot (L_{Peff} + X_{Peff}) \]  

(8.140)

\textbf{Geometric Scaling Effect}

\textbf{Level 1}

Scaling for \texttt{level=1} involves the use of the junction area (area), junction periphery (PJ), and the dimensionless multiplier factor (M) to simulate multiple diodes.
Geometric parameters include

\[
area_{\text{eff}} = area \cdot M 
\]  
\[ (8.141) \]

\[
PJ_{\text{eff}} = PJ \cdot M 
\]  
\[ (8.142) \]

Element and model parameters include:

\[
I_{KE_{\text{eff}}} = I_{KF} \cdot area_{\text{eff}} 
\]  
\[ (8.143) \]

\[
I_{KRE_{\text{eff}}} = I_{KR} \cdot area_{\text{eff}} 
\]  
\[ (8.144) \]

\[
IBV_{\text{eff}} = IBV \cdot area_{\text{eff}} 
\]  
\[ (8.145) \]

\[
I_{Seff} = I_S \cdot area_{\text{eff}} + JSW \cdot PJ_{\text{eff}} 
\]  
\[ (8.146) \]

\[
JSW_{\text{eff}} = JSW \cdot PJ_{\text{eff}} 
\]  
\[ (8.147) \]

\[
EXPLI_{\text{eff}} = EXPLI \cdot area_{\text{eff}} 
\]  
\[ (8.148) \]

\[
C_{j0_{\text{eff}}} = C_{j0} \cdot area_{\text{eff}} 
\]  
\[ (8.149) \]

\[
C_{jp0_{\text{eff}}} = C_{jp0} \cdot PJ_{\text{eff}} 
\]  
\[ (8.150) \]

\[
R_{Seff} = R_s/(area_{\text{eff}}) 
\]  
\[ (8.151) \]

**Level 3**

Level 3 model scaling is affected by the parameters SCALE, SCALM, SHRINK, and M.

When both \(L\) and \(W\) are specified, geometric parameters include

\[
area_{\text{eff}} = Weff \cdot L_{\text{eff}} \cdot M 
\]  
\[ (8.152) \]

\[
PJ_{\text{eff}} = (2 \cdot Weff + 2 \cdot L_{\text{eff}}) \cdot M 
\]  
\[ (8.153) \]

where

\[
Weff = W \times SCALE \times SHRINK + XWeff
\]

\[
L_{\text{eff}} = L \times SCALE \times SHRINK + XWeff
\]

and

\[
XWeff = XW \times SCALM
\]

otherwise

\[
area_{\text{eff}} = area \times M \times SCALE2 \times SHRINK2
\]
Geometric parameters for polysilicon and metal capacitance include

\[ L_{Meff} = L_M \times \text{SCALE} \times \text{SHRINK} \]
\[ W_{Meff} = W_M \times \text{SCALE} \times \text{SHRINK} \]
\[ X_{Meff} = X_M \times \text{SCALM} \]
\[ L_{Peff} = L_P \times \text{SCALE} \times \text{SHRINK} \]
\[ W_{Peff} = W_P \times \text{SCALE} \times \text{SHRINK} \]
\[ X_{Peff} = X_P \times \text{SCALM} \]

Element and model parameters include

\[ I_{KEff} = I_{KF} \cdot \text{area}_{eff} \] \hspace{1cm} (8.154)
\[ I_{KREff} = I_{KR} \cdot \text{area}_{eff} \] \hspace{1cm} (8.155)
\[ IB_{eff} = \frac{IB \cdot \text{area}_{eff}}{(\text{SCALM})^2} \] \hspace{1cm} (8.156)
\[ IS_{eff} = \frac{I_S \cdot \text{area}_{eff}}{\text{SCALM}^2 + JSW \cdot PJ_{eff}/\text{SCALM}} \] \hspace{1cm} (8.157)
\[ JSW_{eff} = \frac{JSW \cdot PJ_{eff}}{(\text{SCALM})} \] \hspace{1cm} (8.158)
\[ EXPLI = EXPLI \cdot \text{area}_{eff} \] \hspace{1cm} (8.159)
\[ R_{S_{eff}} = \frac{R_s}{(\text{area}_{eff} \cdot \text{SCALM}^2)} \] \hspace{1cm} (8.160)
\[ C_{j0_{eff}} = \frac{C_{j0} \cdot \text{area}_{eff}}{\text{SCALM}^2} \] \hspace{1cm} (8.161)
\[ C_{jpeff} = \frac{C_{jpo} \cdot PJ_{eff}}{(\text{SCALM})} \] \hspace{1cm} (8.162)

**Temperature Effects**

**Energy Gap**

The calculation of energy gap is dependent on TLEV. For TLEV=0, 1, or 3, energy gap is always calculated as follows:

\[ E_g(T_{nom}) = 1.16 - (7.02 \times 10^4) \frac{T_n^2}{T_n^2 + 1108.0}. \] \hspace{1cm} (8.163)
If $TLEV=2$, the energy gap is calculated as a function of model parameters $E_g(0)$, $GAP1$, and $GAP2$:

$$E_g(T_{\text{nom}}) = E_g(0) - GAP1 \cdot \frac{T_{\text{nom}}^2}{T_{\text{nom}} + GAP2}.$$  \hfill (8.164)

**Saturation Current**

$$I_S(T) = I_S \cdot e^{\frac{\text{facin}}{N}}$$  \hfill (8.165)

$$JSW(T) = JSW \cdot e^{\frac{\text{facin}}{N}}$$  \hfill (8.166)

For $TLEV=0$ and $1$:

$$\text{facin} = \frac{E_g(0)}{V_i(T_{\text{nom}})} - \frac{E_g(T)}{V_i(T)} + XTI \cdot \ln\left(\frac{T}{T_{\text{nom}}}\right)$$  \hfill (8.167)

For $TLEV=2$:

$$\text{facin} = \frac{E_g(T_{\text{nom}})}{V_i(T_{\text{nom}})} - \frac{E_g(T)}{V_i(T)} + XTI \cdot \ln\left(\frac{T}{T_{\text{nom}}}\right)$$  \hfill (8.168)

For $TLEV=3$:

$$\text{facin} = \frac{V_{SB}}{V_i(T_{\text{nom}})} - \frac{V_{SB}}{V_i(T)} + XTI \cdot \ln\left(\frac{T}{T_{\text{nom}}}\right)$$  \hfill (8.169)

**Breakdown Voltage**

For $TLEV=0$:

$$BV(T) = BV - TCV \cdot \Delta T$$  \hfill (8.170)

For $TLEV=1,2,$ or $3$:

$$BV(T) = BV \cdot (1 - TCV \cdot \Delta T)$$  \hfill (8.171)

where $\Delta T = T - T_{\text{nom}}$

**Transit Time**

$$\tau(T) = \tau \cdot (1 + \tau_{t1} \cdot \Delta T + \tau_{t2} \cdot \Delta T^2)$$  \hfill (8.172)
Contact Potential

For $TLEV=0$

$$PB(T) = PB \cdot \left( \frac{T}{T_{nom}} \right) - V_i(T) \cdot \left( 3 \cdot \ln \left( \frac{T}{T_{nom}} \right) + \frac{E_g(T_{nom})}{V_i(T_{nom})} - \frac{E_g(T)}{V_i(T)} \right)$$  \hspace{1cm} (8.173)

$$PHP(T) = PHP \cdot \left( \frac{T}{T_{nom}} \right) - V_i(T) \cdot \left( 3 \cdot \ln \left( \frac{T}{T_{nom}} \right) + \frac{E_g(T_{nom})}{V_i(T_{nom})} - \frac{E_g(T)}{V_i(T)} \right)$$  \hspace{1cm} (8.174)

For $TLEV=1$ or $2$

$$PB(T) = PB - TPB \times \Delta T$$  \hspace{1cm} (8.175)

$$PHP(T) = PHP - TPHP \times \Delta T$$  \hspace{1cm} (8.176)

For $TLEV=3$

$$PB(T) = PB + dpbdt \times \Delta T$$  \hspace{1cm} (8.177)

$$PHP(T) = PHP - dphpdt \times \Delta T$$  \hspace{1cm} (8.178)

where

$TLEV=2$

$$dpbdt = \frac{E_g(T_{nom}) + 3 \cdot V_i(T_{nom}) + [E_g(0) - E_g(T_{nom})] \cdot \left( 2 - \frac{T_{nom}}{T_{nom} + GAP2} \right) - PB}{T_{nom}}$$  \hspace{1cm} (8.179)

$$dphpdt = \frac{E_g(T_{nom}) + 3 \cdot V_i(T_{nom}) + [E_g(0) - E_g(T_{nom})] \cdot \left( 2 - \frac{T_{nom}}{T_{nom} + GAP2} \right) - PHP}{T_{nom}}$$  \hspace{1cm} (8.180)

and

$TLEV=0$ or $1$

$E_g(0)$ and GAP2 take their default values in the above equations:

$E_g(0) = 1.16$

$GAP2 = 1108.0$
Chapter 8: Device Models

Junction Capacitance

\[ TLEV=0 \]

\[ C_j(T) = C_{j0} \cdot \left\{ 1 + m_j \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{P_B(T)}{P_B} + 1 \right) \right\} \]  

\[ C_{jsw}(T) = C_{jsw0} \cdot \left\{ 1 + m_j \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{P_H(T)}{P_H} + 1 \right) \right\} \]  

\[ TLEV=1 \]

\[ C_j(T) = C_{j0} \cdot (1 + CTA \cdot \Delta T) \]  

\[ C_{jsw}(T) = C_{jsw0} \cdot (1 + CTP \cdot \Delta T) \]  

\[ TLEV=2 \]

\[ C_j(T) = C_{j0} \cdot \left( \frac{P_B}{P_B(T)} \right)^{mj} \]  

\[ C_{jsw}(T) = C_{jsw0} \cdot \left( \frac{P_H}{P_H(T)} \right)^{mjsw} \]  

**Note:** Use \( mj \) instead of \( mj(T) \) in the above equation.

\[ TLEV=3 \]

\[ C_j(T) = C_{j0} \cdot 1 - 0.5 \cdot dpbdT \cdot \frac{\Delta T}{P_B} \]  

\[ C_{jsw}(T) = C_{jsw0} \cdot \left( 1 - 0.5 \cdot dpHdT \cdot \frac{\Delta T}{P_H} \right) \]  

Grading Coefficient

\[ m_j(T) = m_j \cdot (1 + TM1 \cdot \Delta T + TM2 \cdot \Delta T^2) \]  

Resistance

\[ R_s(T) = R_s \cdot (1 + TRS \cdot \Delta T) \]  

Fowler-Nordheim Model (Level 2)

The Fowler-Nordheim model is used to characterize the tunneling current flow through thin insulators in nonvolatile memory devices such as the floating gate devices and the MIOS (metal-insulator-oxide-
Chapter 8: Device Models

Diode

semiconductor) devices. The insulators in these devices are sufficiently thin (about 100 Å) to permit tunneling of carriers.

**Current Equations**

\[ I_d = \text{area}_{eff} \cdot J_F \cdot \left( \frac{V_{d}}{t_{ox}} \right)^2 \cdot e \cdot \frac{E_F \cdot t_{ox}}{V_d} \]  \hspace{1cm} (8.191)

when \( V_d \geq 0 \)

\[ I_d = -\text{area}_{eff} \cdot J_R \cdot \left( \frac{V_{d}}{t_{ox}} \right)^2 \cdot e \cdot \frac{E_F \cdot t_{ox}}{V_d} \]  \hspace{1cm} (8.192)

when \( V_d < 0 \)

where

\[ \text{area}_{eff} = W_{eff} \times L_{eff} \times M \]

and

\[ W_{eff} = W \times SCALM \times SHRINK + XW_{eff} \]

\[ L_{eff} = L \times SCALM \times SHRINK + XW_{eff} \]

\[ XW_{eff} = XW \times SCALM \]

**Capacitance**

\[ C_D = \text{area}_{eff} \cdot \frac{\varepsilon_{ox}}{t_{ox}} \]  \hspace{1cm} (8.193)

**Application Notes—Schottky Barrier Diodes**

The Schottky barrier diode is not explicitly modeled in T-Spice, but it can be simulated using the PN junction diode model provided extra attention is paid to the differences between the PN junction diode and Schottky barrier diode.

Schottky barrier diodes and PN junction diodes have a similar I-V relation, but their saturation current expressions are quite different. For Schottky barrier diode, the saturation current is given by

\[ I_S = A_{RC} \cdot K_{RC} \cdot T_{nom}^2 \cdot \exp\left(\frac{V_{SB}}{V_t}\right) \]  \hspace{1cm} (8.194)

where \( A_{RC} = 1.2 \times 10^6 (A \cdot m^{-2} \cdot K^{-2}) \) is the Richardson constant, \( K_{RC} \) is the ratio of the effective Richardson constant to the Richardson constant, \( V_{SB} \) is the Schottky barrier height, and \( V_t = (kT_{nom})/q \) is the thermal voltage. Use this calculated \( I_S \) value for Schottky barrier diode if \( I_S \) is not specified.
Some typical $KRC$ values are:

<table>
<thead>
<tr>
<th>Type</th>
<th>Si</th>
<th>Ge</th>
<th>GaAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>p-type</td>
<td>0.66</td>
<td>0.34</td>
<td>0.62</td>
</tr>
<tr>
<td>n-type</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(111): 2.2</td>
<td>(111): 1.11</td>
<td>0.068</td>
</tr>
<tr>
<td></td>
<td>(100): 2.1</td>
<td>(100): 1.19</td>
<td></td>
</tr>
</tbody>
</table>

Some other parameter values intended for Schottky barrier diodes are indicated in the diode parameter list. Use these values instead of default in the simulation of Schottky barrier diode if these parameters are not specified.
JFET

The junction field-effect transistor model uses the basic FET model of Schichmann and Hodges. The DC characteristics are modeled by the threshold voltage and the gain factor, and charge storage is modeled by two reverse-biased PN junctions. Source and drain series resistances are included. JFET models are always level 0.

Parameters

```
.model name njf|pjf [parameters]
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>vto</td>
<td>Vt0</td>
<td>Threshold voltage</td>
<td>-2.0</td>
<td>V</td>
</tr>
<tr>
<td>beta</td>
<td>β</td>
<td>Transconductance parameter</td>
<td>1.0 × 10⁻⁴</td>
<td>A/V²</td>
</tr>
<tr>
<td>lambda</td>
<td>λ</td>
<td>Channel length modulation parameter</td>
<td>0.0</td>
<td>1/V</td>
</tr>
<tr>
<td>rd</td>
<td>Rd</td>
<td>Drain series resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rs</td>
<td>Rs</td>
<td>Source series resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>cgs</td>
<td>Cgs</td>
<td>Zero-bias gate-source junction capacitance</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>cgd</td>
<td>Cgd</td>
<td>Zero-bias gate-drain junction capacitance</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>pb</td>
<td>PB</td>
<td>Gate junction potential</td>
<td>1.0</td>
<td>V</td>
</tr>
<tr>
<td>is</td>
<td>IS</td>
<td>Gate junction saturation current</td>
<td>1.0 × 10⁻¹⁴</td>
<td>A</td>
</tr>
<tr>
<td>fc</td>
<td>FC</td>
<td>Forward-bias depletion capacitance coefficient</td>
<td>0.5</td>
<td>—</td>
</tr>
</tbody>
</table>
Large-Signal Model

Equations

*Currents*

In the normal or forward region of operation, the DC currents are described by the following equations, based on the quadratic FET model of Shichmann and Hodges.

For \((V_{GS} - V_{TO}) \leq 0\),

\[
I_{DS} = 0
\]  

(8.195)

For \(0 < (V_{GS} - V_{TO}) \leq V_{DS}\),

\[
I_{DS} = \beta(V_{GS} - V_{TO})^2(1 + \lambda V_{DS})
\]  

(8.196)

For \(0 < V_{DS} < (V_{GS} - V_{TO})\),

\[
I_{DS} = \beta V_{DS}[2(V_{GS} - V_{TO}) - V_{DS}](1 + \lambda V_{DS})
\]  

(8.197)

For the inverse or reverse region of operation where \(V_{DS} < 0\), the same set of equations is used, with \(V_{GS}\) replaced by \(V_{GD}\) and the signs on the \(V_{DS}\) terms reversed.

For \((V_{GD} - V_{TO}) \leq 0\),

\[
I_{DS} = 0
\]  

(8.198)
For $0 < (V_{GD} - V_{TO}) \leq V_{DS}$,

$$I_{DS} = \beta(V_{GD} - V_{TO})^2(1 - \lambda V_{DS})$$  \hspace{1cm} (8.199)

For $0 < V_{DS} < (V_{GD} - V_{TO})$,

$$I_{DS} = \beta V_{DS}[2(V_{GD} - V_{TO}) - V_{DS}](1 - \lambda V_{DS})$$  \hspace{1cm} (8.200)

The gate-to-drain and gate-to-source leakage currents are:

$$I_{GD} = I_S \cdot \left( \frac{q V_{GD}}{kT} - 1 \right)$$  \hspace{1cm} (8.201)

$$I_{GS} = I_S \cdot \left( \frac{q V_{GS}}{kT} - 1 \right)$$  \hspace{1cm} (8.202)

The total currents are then:

$$I_D = I_{DS} - I_{GD}$$  \hspace{1cm} (8.203)

$$I_G = I_{GD} + I_{GS}$$  \hspace{1cm} (8.204)

$$I_S = -(I_{DS} + I_{GS})$$  \hspace{1cm} (8.205)

**Charges**

The charge equations corresponding to $CGD$ and $CGS$ are based on reverse-biased P-N step junctions.

For $VGX < FC \cdot PB$,

$$Q_{GX} = 2C_{GX} \cdot PB \left[ 1 - \sqrt{1 - \frac{V_{GX}}{PB}} \right]$$  \hspace{1cm} (8.206)

For $VGX \geq FC \cdot PB$,

$$Q_{GX} = C_{GX} \left[ \left(1 - \frac{3}{2} FC\right)(V_{GX} - FC \cdot PB) + \frac{1}{4PB} (V_{GX}^2 - (FC \cdot PB)^2) \right] +$$  \hspace{1cm} (8.207)

$$C_{GX} \cdot PB \cdot (1 - \sqrt{1 - FC})$$

$X$ denotes either source ($S$) or drain ($D$). The junction charge equations are identical to those used by the BJT and MOSFET models, except that the grading coefficient has been fixed at 0.5.
Chapter 8: Device Models

MESFET

Parameters

```
.model name nmf | pmf | njf |pjf [parameters]
```

For HSPICE compatibility, you can create a MESFET device in T-Spice using a name which begins with the letter j. The syntax for the MESFET device statement is exactly as documented in “MESFET (z)” on page 192, except that the device name is jname instead of zname.

Additionally, the MESFET model can use either the .model name nmf | pmf naming convention or .model name njf | pjf.

Submodel Selectors

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>LEVEL</td>
<td>Level selector (1, 2 or 3)</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>sat</td>
<td>SAT</td>
<td>Model selector</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>capop</td>
<td>CAPOP</td>
<td>Capacitance model selector</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>dcap</td>
<td>DCAP</td>
<td>Forward-biased diode equation selector</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>tlev</td>
<td>TLEV</td>
<td>Temperature model selector</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>tlevc</td>
<td>TLEV</td>
<td>Junction capacitance temperature model selector</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>nlev</td>
<td>NLEV</td>
<td>Channel thermal noise equation selector</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

DC Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>alpha</td>
<td>$\alpha$</td>
<td>Saturation voltage factor</td>
<td>2.0</td>
<td>$\text{V}^{-1}$</td>
</tr>
<tr>
<td>b</td>
<td>$b$</td>
<td>Doping tail extending parameter</td>
<td>0.3</td>
<td>$\text{V}^{-1}$</td>
</tr>
<tr>
<td>beta</td>
<td>$\beta$</td>
<td>Transconductance parameter</td>
<td>$2.5 \times 10^{-3}$</td>
<td>$\text{A}/\text{V}^2$</td>
</tr>
<tr>
<td>d</td>
<td>$D$</td>
<td>Channel dielectric constant</td>
<td>11.7 (Si)</td>
<td></td>
</tr>
<tr>
<td>gamma</td>
<td>$\gamma ds$</td>
<td>Drain-induced $VT0$ lowering</td>
<td>0.0</td>
<td>$\text{A}$</td>
</tr>
<tr>
<td>gamds</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>is</td>
<td>Isat</td>
<td>Gate saturation current</td>
<td>$1.0 \times 10^{-14}$</td>
<td>$\text{A}$</td>
</tr>
<tr>
<td>k1</td>
<td>$k1$</td>
<td>Body effect on $VT0$</td>
<td>0.0</td>
<td>$\text{V}^{-1}$</td>
</tr>
<tr>
<td>lambda</td>
<td>$\lambda$</td>
<td>Channel length modulation parameter</td>
<td>0.0</td>
<td>$\text{V}^{-1}$</td>
</tr>
<tr>
<td>nchan</td>
<td>ND</td>
<td>Channel doping concentration</td>
<td>$1.552 \times 10^{16}$</td>
<td>$\text{cm}^{-3}$</td>
</tr>
<tr>
<td>rd</td>
<td>Rd</td>
<td>Drain ohmic resistance</td>
<td>0.0</td>
<td>$\Omega$</td>
</tr>
<tr>
<td>rg</td>
<td>Rg</td>
<td>Gate resistance</td>
<td>0.0</td>
<td>$\Omega$</td>
</tr>
</tbody>
</table>
### Capacitance Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>cgs</td>
<td>Cgs</td>
<td>Zero bias G-S capacitance</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>cgd</td>
<td>Cgd</td>
<td>Zero bias G-D capacitance</td>
<td>0.0</td>
<td>F</td>
</tr>
<tr>
<td>crat</td>
<td>CRAT</td>
<td>Source fraction of GCAP</td>
<td>0.666</td>
<td>—</td>
</tr>
<tr>
<td>fc</td>
<td>FC</td>
<td>Forward bias depletion capacitance coefficient</td>
<td>0.5</td>
<td>—</td>
</tr>
<tr>
<td>gcap</td>
<td>GCAP</td>
<td>Total zero bias gate capacitance</td>
<td>—</td>
<td>F</td>
</tr>
<tr>
<td>interr</td>
<td>Eint</td>
<td>Integration error bound (CAPOP=1)</td>
<td>0.01</td>
<td>—</td>
</tr>
<tr>
<td>m</td>
<td>m</td>
<td>Junction grading coefficient (CAPOP=0)</td>
<td>0.5</td>
<td>—</td>
</tr>
<tr>
<td>pb</td>
<td>φs0</td>
<td>Gate junction potential</td>
<td>0.8</td>
<td>V</td>
</tr>
<tr>
<td>tt</td>
<td>τ</td>
<td>Transit time (CAPOP=0)</td>
<td>0.0</td>
<td>s</td>
</tr>
<tr>
<td>vdel</td>
<td>δ</td>
<td>Transition width for Vgs (CAPOP=1)</td>
<td>0.2</td>
<td>V</td>
</tr>
<tr>
<td>vmax</td>
<td>Vmax</td>
<td>Vn limiting value (CAPOP=1)</td>
<td>0.5</td>
<td>V</td>
</tr>
</tbody>
</table>

### Noise Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>af</td>
<td>Af</td>
<td>Flicker noise exponent</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>gdsnoi</td>
<td>GDSNO</td>
<td>Thermal channel noise coefficient (NLEV=3)</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>kf</td>
<td>Kf</td>
<td>Flicker noise coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
</tbody>
</table>
### Geometry Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>acm</td>
<td>ACM</td>
<td>Area calculation method selector</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>align</td>
<td>ALIGN</td>
<td>Misalignment of gate</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>hdif</td>
<td>Hdif</td>
<td>Space between S/D contacts and junction</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>l</td>
<td>L</td>
<td>Default gate length</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>ldel</td>
<td>Ldel</td>
<td>Delta between drawn and optical gate length</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>ldif</td>
<td>Ldif</td>
<td>Distance from junction to gate edge</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>w</td>
<td>W</td>
<td>Default gate width</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>wdel</td>
<td>Wdel</td>
<td>Delta between drawn and optical gate width</td>
<td>0.0</td>
<td>m</td>
</tr>
</tbody>
</table>

### Area Calculation Method (ACM) Parameters

\[
\begin{align*}
\text{ACM=0} & & \text{ACM=1} \\
\text{AREAeff} & = & \frac{W_{\text{eff}} \cdot M}{L_{\text{eff}}} & = & W_{\text{eff}} \cdot L_{\text{eff}} \cdot M \\
\text{RDeff} & = & \frac{R_D}{\text{AREAeff}} & & \text{if } R_D \neq 0: \frac{R_D}{M} \text{ if } R_D = 0: \\
\text{RSeff} & = & \frac{R_S}{\text{AREAeff}} & & \text{if } R_S \neq 0: \frac{R_S}{M} \text{ if } R_S = 0: \\
\text{RGeff} & = & R_G \cdot \frac{\text{AREAeff}}{M^2} & & \text{if } R_G \neq 0: \frac{R_G}{M} \text{ if } R_G = 0: \frac{R_{\text{SHG}}}{L_{\text{eff}} \cdot M}
\end{align*}
\]
Note that the model parameters $IS$, $CGS$, and $CGD$ are unitless when $ACM=0$ and per square meter when $ACM=1$. For example, when $ACM=0$, $CGS = 5 \times 10^{-12}$, $CGD = 1.4 \times 10^{-11}$, and $IS = 1 \times 10^{-14}$; for $ACM=1$, however, $CGS = 5$, $CGD = 14$, and $IS = 1 \times 10^{-2}$.

### Temperature Dependence Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b_{ex}$</td>
<td>$BEX$</td>
<td>Mobility temperature exponent</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>$c_{td}$</td>
<td>$CTD$</td>
<td>Temperature coefficient for $Cgd$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$c_{ts}$</td>
<td>$CTS$</td>
<td>Temperature coefficient for $Cgs$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$e_{g}$</td>
<td>$Eg$</td>
<td>Energy gap for G-D and G-S diodes</td>
<td>1.16</td>
<td>eV</td>
</tr>
<tr>
<td>$g_{ap1}$</td>
<td>$GAP1$</td>
<td>First-order bandgap correction</td>
<td>7.02e-4</td>
<td>eV/deg</td>
</tr>
<tr>
<td>$g_{ap2}$</td>
<td>$GAP2$</td>
<td>Second-order bandgap correction</td>
<td>1108</td>
<td>deg</td>
</tr>
<tr>
<td>$n$</td>
<td>$N$</td>
<td>G-S and G-D diode emission coefficient</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>$t_{cv}$</td>
<td>$TCV$</td>
<td>Temperature coefficient for $VT0$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$t_{pb}$</td>
<td>$TPB$</td>
<td>Temperature coefficient for $pb$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$t_{rd}$</td>
<td>$TRD$</td>
<td>Temperature coefficient for $Rd$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$t_{rg}$</td>
<td>$TRG$</td>
<td>Temperature coefficient for $Rg$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$t_{rs}$</td>
<td>$TRS$</td>
<td>Temperature coefficient for $Rs$</td>
<td>0.0</td>
<td>deg$^{-1}$</td>
</tr>
<tr>
<td>$x_{ti}$</td>
<td>$XTI$</td>
<td>$Isat$ temperature exponent</td>
<td>0.0</td>
<td>—</td>
</tr>
</tbody>
</table>
Large-Signal Model

Equations

There are three levels of the T-Spice MESFET model.

- **Level 1.** The Curtice model—a diode-based capacitance model and a simplified $I_{ds}$ calculation.
- **Level 2.** The Statz model (Statz et al. 1987)—a revision of the Curtice model, with an improved capacitance model and a more sophisticated $I_{ds}$ calculation.
- **Level 3.** An HSPICE-compatible model—highly customizable.

**Currents**

For all three levels,

$$I_{ds} = \beta \cdot \frac{(V_{gst})^{VGEXP}}{A} \cdot B \cdot (1 + \lambda V_{ds})$$

(8.208)

where

$$V_{gst} = V_{gs} - V_{T0} - \gamma_{ds} \cdot V_{ds}$$

(8.209)

$A$ is the doping profile extension factor. At level 1, $A = 0$; otherwise:

$$A = 1 + b V_{gst}$$

(8.210)

$VGEXP$ can be varied only at level 3; at levels 1 and 2, it is fixed at 2.0.
$B$ is the saturation term; its form depends on the value of $SAT$.

When $SAT = 0$:

$$B = \tanh (\alpha V_{ds})$$

(8.211)

This form is fixed (that is, $SAT$ can only take a value of 0) at level 1.

When $SAT = 1$:

$$B = \tanh \left( \frac{V_{ds}}{V_{gm}} \right)$$

(8.212)

When $SAT = 2$:

$$B = 1 - \left( 1 - \frac{V_{ds}}{\alpha} \right)^3$$

(8.213)

When $V_{ds} > \frac{3}{\alpha}$, $B = 1$. This form is fixed (that is, $SAT$ can only take a value of 2) at level 2.

When $SAT = 3$ and $V_{ds} < \frac{SATEXP}{\alpha}$:

$$B = 1 - \left( 1 - \frac{V_{ds}}{SATEXP} \right)^{SATEXP}$$

(8.214)

Otherwise, $B = 1$.

**Capacitances**

There are two capacitance models, one at level 1 (a “diode-like” model) and one at level 2 (the Statz model). The $CAPOP$ parameter selects between the models.

When $CAPOP = 0$:

When the junctions are reverse-biased ($V_{gd} < FC \cdot \phi_{s0}$):

$$C_{gd} = C_{gd}(0) \cdot \left( 1 - \frac{V_{gd}}{\phi_{s0}} \right)^{-m}$$

(8.215)

$$C_{gs} = C_{gs}(0) \cdot \left( 1 - \frac{V_{gs}}{\phi_{s0}} \right)^{-m}$$

(8.216)
When the junctions are forward-biased, two models are available; the \( DCAP \) parameter selects between them. When \( DCAP = 1 \):

\[
C_{gd} = \tau \cdot \frac{\partial I_{gd}}{\partial V_{gd}} + C_{gd(0)} \cdot \frac{1 - FC(1 + m) + m \frac{V_{gd}}{\Phi_{sd}}}{(1 - FC)^m + 1} \tag{8.217}
\]

\[
C_{gs} = \tau \cdot \frac{\partial I_{gs}}{\partial V_{gs}} + C_{gs(0)} \cdot \frac{1 - FC(1 + m) + m \frac{V_{gs}}{\Phi_{sd}}}{(1 - FC)^m + 1} \tag{8.218}
\]

When \( DCAP = 2 \) (default):

\[
C_{gd} = \tau \cdot \frac{\partial I_{gd}}{\partial V_{gd}} + C_{gd(0)} \cdot \left( 1 + m \frac{V_{gd}}{\Phi_{sd}} \right) \tag{8.219}
\]

\[
C_{gs} = \tau \cdot \frac{\partial I_{gs}}{\partial V_{gs}} + C_{gs(0)} \cdot \left( 1 + m \frac{V_{gs}}{\Phi_{sd}} \right) \tag{8.220}
\]

When \( CAPOP = 1 \):

The basic Statz capacitance equations are:

\[
C_{gs, gd} = \frac{1}{4} \left[ C_{gs(0)} \times \left( 1 + \frac{V_{eff} - V_{T0}}{\sqrt{(V_{eff} - V_{T0})^2 + \delta^2}} \right) \left[ \frac{V_{gs} - V_{gd}}{\sqrt{(V_{gs} - V_{gd})^2 + (\alpha)^2}} \right] \right] + \frac{1}{2} \times C_{gd(0)} \left( 1 + \frac{V_{gs} - V_{gd}}{\sqrt{(V_{gs} - V_{gd})^2 + (\alpha)^2}} \right) \tag{8.221}
\]

where

\[
V_{eff} = \frac{1}{2} [V_{gs} + V_{gd} + \sqrt{(V_{gs} - V_{gd})^2 + (\alpha)^2}] \tag{8.222}
\]

\[
V_n = \frac{1}{2} [V_{eff} + V_{T0} + \sqrt{(V_{eff} - V_{T0})^2 + (\delta)^2}] \tag{8.223}
\]

If \( V_n > V_{max} \), then \( V_n \) is limited to \( V_{max} \). In the plus/minus signs (\( \pm \)) above, the top signs hold for \( C_{gs} \), the bottom ones for \( C_{gd} \).

**Temperature Dependence Equations**

In the following, \( \Delta T = T - T_{ref} \). \( T_{ref} \) is the temperature at which the user-supplied parameters are valid, which defaults to 25° C. All temperatures in the following equations are assumed to be in Kelvin.
Temperature dependence. For all values of TLEV, the energy gap $E_g$ at the reference and simulation temperatures is calculated using

$$E_g(T) = E_G - GAP1 \times \frac{T}{T + GAP2}$$

(8.224)

For TLEV 0 and 1, $E_G$, GAP1, and GAP2 are held fixed at 1.16, $7.02 \times 10^{-4}$, and 1108.0, respectively, regardless of what values the user specifies for these parameters.

Saturation current temperature dependence. The saturation current temperature extrapolation is calculated by the following general equation:

$$I_s(T) = IS \cdot \exp \left\{ \frac{e}{k_B N} \left( \frac{E_g(T_{ref})}{T_{ref}} - \frac{E_g(T)}{T} + XTI \cdot \ln \left( \frac{T}{T_{ref}} \right) \right) \right\}$$

(8.225)

For TLEV 0 and 1, the user-supplied $E_G$ is used in place of $E_g(T)$ for all $T$. For TLEV=2, $E_g(T) = E_g(T)$, as calculated above.

Capacitance parameters temperature dependence. A separate selection parameter, TLEVC, is used to choose one of four methods of temperature-adjusting the gate capacitance values. This parameter also influences the temperature compensation of $\phi_{s0}$.

For TLEVC=0, the gate junction potential ($\phi_{s0}$) is temperature-adjusted as follows:

$$\phi_{s0}(T) = \Phi_{s0} \cdot \frac{T}{T_{ref}} - \frac{3k_B}{e} T \ln \left( \frac{T}{T_{ref}} \right) - E_g(T_{ref}) \cdot \frac{T}{T_{ref}} + E_g(T)$$

(8.226)

The gate capacitances are calculated by

$$C_{gs}(T) = C_{gs} \cdot \left[ 1 + m \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{\phi_{s0}(T)}{\Phi_{s0}(T_{ref})} + 1 \right) \right]$$

(8.227)

$$C_{gd}(T) = C_{gs} \cdot \left[ 1 + m \cdot \left( 4.0 \times 10^{-4} \cdot \Delta T - \frac{\phi_{s0}(T)}{\Phi_{s0}(T_{ref})} + 1 \right) \right]$$

(8.228)

For TLEVC=1, the junction potential is temperature-adjusted with

$$\phi_{s0}(T) = \Phi_{s0} - TPB \cdot \Delta T$$

(8.229)

The gate capacitances are calculated by

$$C_{gs}(T) = C_{gs} \cdot (1 + CTS + \Delta T)$$

(8.230)

$$C_{gd}(T) = C_{gd} \cdot (1 + CTS \cdot \Delta T)$$

(8.231)
For TLEV=2, the junction potential is calculated as it is for TLEV=1. The gate capacitances are calculated by

\[ C_{gs}(T) = C_{gs} \cdot \left( \frac{\phi_s}{\phi_{s0}} \right)^m \]  

\[ C_{gd}(T) = C_{gd} \cdot \left( \frac{\phi_s}{\phi_{s0}} \right)^m \]  

(8.232)  

(8.233)

For TLEV=3, the junction potential calculation attempts to estimate the term \( \frac{d\phi_s}{dT} \) (\( dpbdt \)) using the following equations, depending on TLEV.

For TLEV=0 or 1,

\[ dpbdt = -\left( \frac{1}{T_{ref}} \right) E_g(T_{ref}) + \frac{3k_B}{e} T_{ref} + (1.16 - E_g(T_{ref})) \cdot \left( 2 - \frac{T_{ref}}{1108} \right) (-\phi_{s0}) \]  

(8.234)

For TLEV=2,

\[ dpbdt = -\left( \frac{1}{T_{ref}} \right) E_g(T_{ref}) + \frac{3k_B}{e} T_{ref} + (E_g - E_g(T_{ref})) \cdot \left( 2 - \frac{T_{ref}}{T_{ref} + GAP2} \right) (-\phi_{s0}) \]  

(8.235)

The gate capacitances are then calculated by

\[ C_{gs}(T) = C_{gs} \cdot \left( 1 - 0.5 \cdot dpbdt \times \frac{\Delta T}{\phi_{s0}} \right) \]  

(8.236)

\[ C_{gd}(T) = C_{gd} \cdot \left( 1 - 0.5 \cdot dpbdt \times \frac{\Delta T}{\phi_{s0}} \right) \]  

(8.237)

Note that \( C_{gs, gd} \) becomes what is referred to elsewhere as \( C_{gs, gd}(0) \), the zero-bias gate capacitances.

\( V_{TO} \) temperature dependence. The threshold voltage is temperature-compensated with a simple linear model

\[ V_{TO}(T) = V_{TO} - TCV \times \Delta T \]  

(8.238)

Transconductance temperature dependence. The temperature compensation of the transconductance factor (\( \beta \)) is calculated as follows

\[ \beta(T) = \beta \cdot \left( \frac{T}{T_{ref}} \right)^{BEX} \]  

(8.239)
Parasitic resistance temperature dependence. The source, drain, and gate parasitic resistances are temperature-compensated by the following equations

\[
R_d(T) = R_d \cdot (1 + TRD \times \Delta T) \quad (8.240)
\]

\[
R_s(T) = R_s \cdot (1 + TRS \times \Delta T) \quad (8.241)
\]

\[
R_g(T) = R_g \cdot (1 + TRG \times \Delta T) \quad (8.242)
\]
MOSFET Levels 1/2/3 (Berkeley SPICE 2G6)

Parameters

```
.model name nmos | pmos level=1 | 2 | 3 [parameters]
```

Also see “Additional MOSFET Parameters” on page 492.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>level</td>
<td>LEVEL</td>
<td>Model selector</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>vto</td>
<td>Vt0</td>
<td>Zero-bias threshold voltage</td>
<td>Computed</td>
<td>V</td>
</tr>
<tr>
<td>kp</td>
<td>KP</td>
<td>MOSFET transconductance</td>
<td>Computed</td>
<td>A/V²</td>
</tr>
<tr>
<td>gamma</td>
<td>γ</td>
<td>Bulk threshold parameter</td>
<td>Computed</td>
<td>V¹/²</td>
</tr>
<tr>
<td>phi</td>
<td>φ</td>
<td>Surface potential</td>
<td>Computed</td>
<td>V</td>
</tr>
<tr>
<td>lambda</td>
<td>λ</td>
<td>Channel length modulation</td>
<td>Computed</td>
<td>V⁻¹</td>
</tr>
<tr>
<td>tox</td>
<td>Tox</td>
<td>Gate oxide thickness</td>
<td>Computed</td>
<td>If &lt; 1, m; if &gt; 1, Å</td>
</tr>
<tr>
<td>nsub</td>
<td>Nsub</td>
<td>Bulk doping concentration</td>
<td>0.0</td>
<td>cm⁻³</td>
</tr>
<tr>
<td>dnb</td>
<td>nb</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nss</td>
<td>NSS</td>
<td>Surface state density</td>
<td>0.0</td>
<td>cm⁻²</td>
</tr>
<tr>
<td>NFS</td>
<td>NFS</td>
<td>Fast surface state density</td>
<td>Computed</td>
<td>cm⁻²</td>
</tr>
<tr>
<td>nfs</td>
<td>dnf</td>
<td>nf</td>
<td>dfs</td>
<td></td>
</tr>
<tr>
<td>xj</td>
<td>Xj</td>
<td>Junction depth</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>ld</td>
<td>LD</td>
<td>Lateral diffusion</td>
<td>0.75 × xj</td>
<td>m</td>
</tr>
<tr>
<td>wd</td>
<td>WD</td>
<td>Width diffusion</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>xl</td>
<td>dl</td>
<td>ldel</td>
<td>Xl</td>
<td>0.0</td>
</tr>
<tr>
<td>xw</td>
<td>dw</td>
<td>wdel</td>
<td>Xw</td>
<td>0.0</td>
</tr>
<tr>
<td>uo</td>
<td>μ0</td>
<td>Electron or hole mobility</td>
<td>600.0</td>
<td>cm²/V·s</td>
</tr>
<tr>
<td>ucrit</td>
<td>UCRIT</td>
<td>Critical field mobility degradation coefficient</td>
<td>1.0 × 10⁻⁴</td>
<td>V/cm</td>
</tr>
<tr>
<td>uexp</td>
<td>UEXP</td>
<td>Critical field mobility degradation exponent</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>vmax</td>
<td>VMAX</td>
<td>Maximum carrier drift velocity</td>
<td>0.0</td>
<td>m/s</td>
</tr>
<tr>
<td>neff</td>
<td>NEFF</td>
<td>Total channel charge coefficient</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>delta</td>
<td>δ</td>
<td>Width effect on threshold voltage</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>tpg</td>
<td>TPG</td>
<td>Type of gate material:</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Doping type same as source-drain (tpg=1)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>▪ Doping type reverse of source-drain (tpg=-1)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 8: Device Models

Large-Signal Model

This schematic represents the Level 2 model. The Level 1 model has the same configuration, without the junction diodes.

The terminals (G, D, S, B) represent the gate, drain, source, and bulk connections, respectively, of the MOSFET. \( I_d \) is the current flowing from the drain to the source as a function of \( (V_{gs}, V_{ds}, V_{bs}) \). The ohmic resistance of the drain and source diffusion junctions are represented by \( R_d \) and \( R_s \). A P-channel MOSFET can be modeled by reversing the polarity of \( (V_{gs}, V_{ds}, V_{bs}) \), the current source \( I_d \), and the junction diodes.

When \( nrs \) and \( nrd \) are not given as options on the MOSFET device statement, T-Spice computes them by calculating the number of squares from the geometry of the junction and then multiplying them by \( R_{SH} \) to obtain the drain and source resistors \( R_d \) and \( R_s \).

## Level 1 Equations

The Level 1 model is included for SPICE compatibility, but it is inaccurate when simulating circuits with analog characteristics. The original intent of the model was to provide an approximation for digital circuits that could be evaluated quickly, reducing the simulation time by as much as a factor of two. However, due to the inherent inaccuracies of the level 1 equations in analog simulations, this model should generally be avoided.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>theta</td>
<td>( \theta )</td>
<td>Mobility modulation</td>
<td>0.0</td>
<td>V-1</td>
</tr>
<tr>
<td>eta</td>
<td>( \eta )</td>
<td>Static feedback</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>kappa</td>
<td>( \kappa )</td>
<td>Saturation field factor</td>
<td>0.2</td>
<td>—</td>
</tr>
<tr>
<td>del</td>
<td>( \delta )</td>
<td>Channel length reduction per side</td>
<td>0.0</td>
<td>m</td>
</tr>
</tbody>
</table>
Current

Current behavior is modeled by three equations, representing the three regions of MOSFET operation: cutoff, linear, and saturation.

Cutoff region: \( V_{gs} \leq V_{th} \)

\[ I_{ds} = 0 \quad (8.243) \]

Linear region: \( V_{gs} > V_{th} \) and \( (V_{gs} - V_{th}) > V_{ds} \)

\[ I_{ds} = \frac{\beta}{2} V_{ds} (2(V_{gs} - V_{th}) - V_{ds})(1 + \lambda V_{ds}) \quad (8.244) \]

Saturation region: \( V_{gs} > V_{th} \) and \( (V_{gs} - V_{th}) \leq V_{ds} \)

\[ I_{ds} = \frac{\beta}{2} (V_{gs} - V_{th})^2 (1 + \lambda V_{ds}) \quad (8.245) \]

In equations (8.244) and (8.245),

\[ \beta = \frac{W - (2 \cdot WD)}{L - (2 \cdot LD)} \cdot KP \quad (8.246) \]

Charge

\[ Q_g = C_{gb} V_{gb} + C_{gd} V_{gd} + C_{gs} V_{gs} \quad (8.247) \]

\[ Q_d = C_{gd} V_{gd} + \int C_{dh}(v) dv \quad (8.248) \]

\[ Q_s = C_{gs} V_{gs} + \int C_{sh}(v) dv \quad (8.249) \]

\[ Q_b = -(Q_g + Q_d + Q_s) \quad (8.250) \]

Capacitance

The Meyer gate capacitance model was replaced by a simplified model that provides conservation of charge.

\[ C_{gb} = C_{gs} = C_{gd} = \frac{1}{3} W L \frac{\varepsilon_{ox}}{T_{ox}} \quad (8.251) \]

The junction capacitance for the drain and source is dependent on the drain-bulk and drain-source voltages.

\[ C_{db} = C_{pm}(V_{tb}, AD, PD) \quad (8.252) \]

\[ C_{sb} = C_{pm}(V_{ts}, AS, PS) \quad (8.253) \]
For reverse bias $V_{pn} < FC \cdot PB$,

$$C_{pn}(V_{pn}, A, P) = \left( CJ \cdot \frac{A}{V_{pn}^2} \right) + \left( CJSW \cdot \frac{P}{1 - \frac{V_{pn}}{PB}} \right)$$  \hspace{1cm} (8.254)$$

For forward bias $V_{pn} \geq FC \cdot PB$,

$$C_{pn}(V_{pn}, A, P) = CJ\left( \frac{A}{1 - FC(1 + MJ)} \right) \left( 1 - FC1 + MJ + \frac{V_{pn}}{PB} MJ \right) +$$

$$CJSW\left( \frac{P}{1 - FC(1 + MJSW)} \right) \left( 1 - FC1 + MJSW + \frac{V_{pn}}{PB} MJSW \right)$$  \hspace{1cm} (8.255)$$

In equations (8.254) and (8.255), $A$ denotes area and $P$ denotes perimeter.

The charge can be calculated using

$$Q = \int C(V)dv$$  \hspace{1cm} (8.256)$$

**Threshold Voltage**

An increase in the threshold voltage is due to a reverse bias from the gate to the substrate called the *body effect*, which causes degradation in the current drive of a transistor.

$$V_{to} = V_{FB} + (\gamma \sqrt{\Phi} + \Phi)$$  \hspace{1cm} (8.257)$$

$$\gamma = \sqrt{\frac{2e_0\varepsilon_oxqN_{sub}}{C_{ox}}}$$  \hspace{1cm} (8.258)$$

$$C_{ox} = \frac{e_0\varepsilon_{ox}}{T_{ox}}$$  \hspace{1cm} (8.259)$$

$$KP = \mu_0 C_{ox}$$  \hspace{1cm} (8.260)$$

$$V_{FB} = \left( -TPG \cdot \frac{E_F}{2} \right) \frac{\Phi}{2} - \frac{qN_{ss}}{C_{ox}}$$  \hspace{1cm} (8.261)$$

Equation (8.257) is dependent on $V_{bs}$ and is implemented as follows:

$$V_{th} = V_{to} + \gamma(bodyterm - \sqrt{2\Phi_F})$$  \hspace{1cm} (8.262)$$

When $V_{bs} \leq 0.0$,

$$bodyterm = \sqrt{2\Phi_F - V_{bs}}$$  \hspace{1cm} (8.263)$$
When $V_{bs} > 0.0$,

$$bodyterm = \max \left( \sqrt{2\phi_F} - \frac{V_{bs}}{2\sqrt{2\phi_F}}, 0 \right)$$  \hspace{1cm} (8.264)$$

In Equations (8.262) through (8.264),

$$2\phi_F = \phi = \frac{kT}{q} \ln \left( \frac{N_{sub}}{n_i} \right)$$  \hspace{1cm} (8.265)$$

**Level 2 Equations**

The Level 2 model uses two current equations. Since the MOSFET is not an ideal switch, the current begins to flow before the transistor reaches the turn-on voltage. This region is called weak inversion. As the voltage on the gate approaches the threshold voltage, it conducts current much more vigorously. At this point the channel is in the strong inversion region. The strong inversion region includes the linear and saturation regions. Many second-order effects that control the amount of current are calculated in the Level 2 model, including backgate bias and short and narrow channel effects. The Level 2 current equations are similar to the Grove equation for a MOSFET. These equations insure the continuity of current at $V_{on}$ (threshold point) through the transistor regions.

**Subthreshold Region**

The weak inversion current equation is used when the transistor is in the subthreshold region, $V_{gs} < V_{on}$.

$$I_{ds} = \beta \left( \left( V_{on} - V_{bin} - \frac{\eta V_{on}}{2} \right) V_{on} - \frac{2}{3} \gamma_s (2\phi_F - V_{bs} + V_{on})^{3/2} - (2\phi_F - V_{bs})^{3/2} \right) e^{\frac{qV_{gs}}{nkT}}$$  \hspace{1cm} (8.266)$$

The strong inversion current equation is used to calculate the current when $V_{gs} \geq V_{on}$:

$$I_{ds} = \beta \left( \left( V_{gs} - V_{bin} - \frac{\eta V_{ds}}{2} \right) V_{ds} - \frac{2}{3} \gamma_s (2\phi_F - V_{bs} + V_{ds})^{3/2} - \left( 2\phi_F - V_{bs} \right)^{3/2} \right)$$  \hspace{1cm} (8.267)$$

where $V_{ds} = V_{dsat}$ for $V_{ds} > V_{dsat}$.
The voltages calculated to determine the second order effects of the transistor currents are defined as:

\[ V_{th} = V_{bin} + \gamma_s n \sqrt{2 \Phi_F - V_{bs}} \]  
\[ V_{bin} = V_{bs} + \delta \left( \frac{\pi \varepsilon_0 \varepsilon_{si}}{4 C_{ox} W_{eff}} \right) (2 \Phi_F - V_{bs}) \]

\[ V_{bs} = V_{fb} + 2 \Phi_F \]

\[ 2 \Phi_F = \phi = 2 \frac{kT}{q} \ln \left( \frac{N_{sub}}{n_i} \right) \]

\[ V_{fb} = \phi_{mu} - \frac{q \cdot N_{SS}}{C_{ox}} \]

The voltage \( V_{on} \) is used to switch between the weak and strong inversion model regions. When \( NFS \) is not specified,

\[ V_{on} = V_{th} \]

\[ n = \infty \]

However, a more accurate cut-on point can be modeled by using a curve fitting parameter \( NFS \) in the evaluation of \( V_{on} \):

\[ V_{on} = V_{th} + \frac{nkT}{q} \]

where

\[ n = 1 + \frac{C_{FS}}{C_{ox}} + \frac{C_D}{C_{ox}} \]

\[ C_{FS} = q \cdot NFS \]

\[ C_D = \frac{\partial Q_B}{\partial V_{bs}} = C_{ox} \left[ -\gamma_s \left( \frac{d \sqrt{2 \Phi_F - V_{bs}}}{dV_{bs}} \right) + \left( \frac{\partial \gamma_s}{\partial V_{bs}} \right) \sqrt{2 \Phi_F - V_{bs}} + \delta \frac{\pi \varepsilon_0 \varepsilon_{si}}{4 C_{ox} W_{eff}} \right] \]
Taking the partial derivatives and applying the chain rule yields

\[
C_D = C_{ox}\left[\frac{1}{2} \frac{1}{\sqrt{\Phi_F - V_{bs}}} - \gamma_s \frac{2}{\sqrt{\Phi_F - V_{bs}}} \right] \left(\frac{X_D}{4L} \frac{1}{1 + \frac{2X_D}{X_f} \frac{2}{\sqrt{\Phi_F - V_{bs}}}} \right)^{1/2} \cdot \frac{1}{\sqrt{\Phi_F - V_{bs}}}
\]

(8.279)

\[
+ \frac{X_D}{4L} \frac{1}{\left(1 + \frac{2X_D}{X_f} \frac{2}{\sqrt{\Phi_F - V_{bs} + V_{ds}}}ight)^{1/2}} \cdot \frac{1}{\sqrt{\Phi_F - V_{bs} + V_{ds}}} + \delta \frac{\pi \varepsilon_0 \varepsilon_r}{4C_{ox} W_{eff}}
\]

Equation (8.279) assumes that \( V_{bs} \) is negative. When \( V_{bs} \) is positive, \( \sqrt{2\Phi_F - V_{bs}} \) should be replaced with

\[
\frac{\sqrt{2\Phi_F}}{1 + \frac{V_{bs}}{4\Phi_F}}
\]

(8.280)

**Linear and Saturation Regions**

The saturation voltage \( V_{dsat} \) can be computed with either of two methods. Which method is chosen depends on whether the input parameter \( VMAX \) has been defined.

**Method 1.** When \( VMAX \) is not defined, the model computes \( V_{dsat} \) assuming that the channel is pinched off at the drain:

\[
V_{dsat} = \frac{V_{gs} - V_{bin}}{\eta} + \frac{1}{2} \left(\frac{2}{\eta}\right)^2 \left(1 - \frac{1}{\eta} \left[1 + \frac{2}{\gamma_s} \left(\frac{V_{gs} - V_{bin}}{\eta} + 2\Phi_F - V_{bs}\right)\right]\right)
\]

(8.281)

If the parameter \( \lambda \) is not defined while using the pinch-off method, then it can be computed as

\[
\lambda = \frac{X_D}{L_i V_{ds}} \sqrt{\frac{V_{ds} - V_{dsat}}{4}} + \sqrt{\frac{V_{ds} - V_{dsat}}{4}}^2
\]

(8.282)

The channel length becomes smaller due to the pinch-off of the channel. The effective channel length is

\[
L_{eff} = L_i (1 - \lambda V_{ds})
\]

(8.283)

**Method 2.** When \( VMAX \) is defined as \( V_{dsat} \) and the channel length modulation is calculated using the carrier scattering limited velocity model, this method appropriately models saturation current for short-
channel MOSFETs as charge carriers reaching their maximum scattering limited velocity before the pinch-off effect comes into play.

\[
V_{\text{MAX}} = \frac{\mu_s \left( \frac{V_{gs} - V_{\text{bin}} - \eta V_{\text{dsat}}}{2} \right) V_{\text{dsat}} - \frac{2\gamma_s}{3} \left( V_{\text{dsat}} + 2\phi_F - V_{bs} \right)^{3/2} - \left( 2\phi_F - V_{bs} \right)^{3/2}}{L_{\text{eff}} \left( V_{gs} - V_{\text{bin}} - \eta V_{\text{dsat}} - \gamma_{\text{N}} \left( V_{\text{dsat}} + 2\phi_F - V_{bs} \right) \right)}
\]

(8.284)

The effective channel length is dependent on \(V_{\text{MAX}}\) and \(V_{\text{dsat}}\) and computed as

\[
L_{\text{eff}} = L_1 - X_D \left( \frac{(X_D V_{\text{MAX}})^2}{2\mu_s} + (V_{ds} - V_{\text{dsat}}) \right)^{1/2} + \frac{(X_D)^2 V_{\text{MAX}}}{2\mu_s}
\]

(8.285)

Solving for \(V_{\text{dsat}}\) is difficult because it requires the simultaneous solution of the nonlinear equations (8.284) and (8.285). A less computationally expensive approach is desirable:

Assume that \(L_{\text{eff}}\) and \(L_1\) in equation (8.285) are equal. Then by selecting parameter \(N_{\text{EFF}}\), adjust the value of \(X_D\) to obtain a good fit to the I-V characteristics of the MOSFET.

\[
X_D = \sqrt{\frac{2\varepsilon_0 \varepsilon_{\text{si}}}{qN_{\text{sub}} N_{\text{EFF}}}}
\]

(8.286)

Equation (8.285) can be solved using Ferrari’s method. There will be 2 or 4 real roots. The smallest positive real root is the correct value of \(V_{\text{dsat}}\).

If \(\lambda\) is not defined, then it can be computed using \(V_{\text{dsat}}\) and equations (8.284) and (8.287):

\[
\lambda = \frac{X_D}{L_1 V_{ds}} \left( \left( \frac{(X_D V_{\text{MAX}})^2}{2\mu_s} + (V_{ds} - V_{\text{dsat}}) \right)^{1/2} + \frac{X_D V_{\text{MAX}}}{2\mu_s} \right)
\]

(8.287)

Then \(L_{\text{eff}}\) can be computed as

\[
L_{\text{eff}} = L_1 (1 - \lambda V_{ds})
\]

(8.288)

Thus the value of \(L_{\text{eff}}\) is available from either of the two current saturation methods—channel pinch-off or carrier scattering limited velocity. If \(L_{\text{eff}}\) is smaller than the zero-bias depletion layer width \(W_B = X_D \sqrt{P_B}\), then \(L_{\text{eff}}\) must be recomputed as

\[
L_{\text{eff}} = \frac{W_B}{1 + \Delta L - L_{\text{max}}} \frac{W_B}{W_B}
\]

(8.289)

where

\[
\Delta L = \lambda V_{ds} L_1
\]

(8.290)

\[
L_{\text{max}} = L_1 - W_B
\]

(8.291)
Equation (8.289) prevents numerical non-convergence, but does not model the punch-through effect.

The voltage $V_{dsat}$ is compared with $V_{ds}$. If $V_{ds}$ is greater than $V_{dsat}$, then $V_{ds}$ is set equal to $V_{dsat}$, thus clamping the drain-to-source voltage to $V_{dsat}$ in the saturation region.

The result is a potential drop ($V_{ds}$) across the saturated transistor and a smaller effective transistor gate length that increases the gain ($\beta$).

$$\beta = \frac{W_{\text{eff}}}{L_{\text{eff}}} K_P \quad (8.292)$$

$$K_P = \mu_s C_{ox} \quad (8.293)$$

$K_P$ is the transconductance of the MOSFET. If it is not specified, it will take the calculated value.
Second-Order Effects

Other variables, coefficients, and modified input parameters that are used in the model are defined as follows:

\[
\phi_{ms} = -\left(\frac{2\phi_F + E_g}{2}\right)
\]

(8.294)

\[
C_{ox} = \frac{\varepsilon_0 \varepsilon_{ox}}{T_{ox}}
\]

(8.295)

\[
L_I = L - (2 \cdot LD)
\]

(8.296)

\[
W_{eff} = W - (2 \cdot WD)
\]

(8.297)

\[
\eta = 1 + \delta \left(\frac{\pi \varepsilon_0 \varepsilon_{ox}}{4C_{ox}W_{eff}}\right)
\]

(8.298)

\[
\mu_s = \frac{U_{CRIT}V_{gs}V_{ds}}{C_{ox}(V_{gs} - V_{os})}^{UEXP}
\]

(8.299)

\[
\gamma_s = \gamma(1 - \alpha_s - \alpha_d)
\]

(8.300)

\[
\gamma = \sqrt{\frac{2q\varepsilon_0\varepsilon_{si}N_{sub}}{C_{ox}}}
\]

(8.301)

\[
\alpha_s = \frac{X_S}{L_I} \cdot \left(1 + 2\frac{W_S}{X_J} - 1\right)
\]

(8.302)

\[
\alpha_D = \frac{X_D}{L_I} \cdot \left(1 + 2\frac{W_D}{X_J} - 1\right)
\]

(8.303)

\[
W_S = X_D \sqrt{2\phi_F - V_{bs}}
\]

(8.304)

\[
W_D = X_D \sqrt{2\phi_F - V_{bs} + V_{ds}}
\]

(8.305)

\[
X_D = \frac{2\varepsilon_0 \varepsilon_{si}}{q N_{sub}}
\]

(8.306)

Ward-Dutton Charge Model

The Ward-Dutton charge model was used to replace the shortcomings of the Meyer capacitance model. It has been shown that assuming capacitance reciprocity leads to non-conservation of charge. This non-conservation of bogus charge can cause various charge-dependent circuits to be simulated incorrectly.
The Ward-Dutton model derives equations based on charge instead of capacitance and conserves charge through all regions of the transistor. This model applies to Level 2 and Level 3 MOSFET models only.

The following table lists the parameters used by the equations describing the Ward-Dutton charge model (see “MOSFET Levels 4 and 13 (BSIM1)” on page 447).

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_{dc}$</td>
<td>DC drain current</td>
<td>A</td>
</tr>
<tr>
<td>$V_d$</td>
<td>Drain voltage</td>
<td>V</td>
</tr>
<tr>
<td>$V_g$</td>
<td>Gate voltage</td>
<td>V</td>
</tr>
<tr>
<td>$V_s$</td>
<td>Source voltage</td>
<td>V</td>
</tr>
<tr>
<td>$V_{dsat}$</td>
<td>Saturation voltage</td>
<td>V</td>
</tr>
<tr>
<td>$Q_b$</td>
<td>Total bulk charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_g$</td>
<td>Total gate charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_s$</td>
<td>Total source charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_d$</td>
<td>Total drain charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{gi}$</td>
<td>Intrinsic gate charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{si}$</td>
<td>Intrinsic source charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{di}$</td>
<td>Intrinsic drain charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{go}$</td>
<td>Overlap gate charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{so}$</td>
<td>Overlap source charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{do}$</td>
<td>Overlap drain charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{sj}$</td>
<td>Junction source charge</td>
<td>C</td>
</tr>
<tr>
<td>$Q_{dj}$</td>
<td>Junction drain charge</td>
<td>C</td>
</tr>
</tbody>
</table>

The Ward-Dutton model defines these normalized voltages as:

\[
V_g = V_{gs} - V_{bs} - V_{fb} \tag{8.307}
\]

\[
V_d = V_{ds} - V_{bs} + 2\phi_F \tag{8.308}
\]

\[
V_s = -V_{bs} + 2\phi_F \tag{8.309}
\]

The charge on the gate, drain, and source of a transistor is the total charge due to the overlap, junction, and intrinsic channel charge.

\[
Q_g = Q_{gi} + Q_{go} \tag{8.310}
\]

\[
Q_d = Q_{di} + Q_{do} + Q_{dj} \tag{8.311}
\]

\[
Q_s = Q_{si} + Q_{so} + Q_{sj} \tag{8.312}
\]
The overlap capacitors are due to the gate edges overlapping the substrate and diffusion junctions.

\[ Q_{go} = C_{gso}W_{eff}V_{gs} + C_{gso}W_{eff}(V_{gs} - V_{ds}) + 2C_{gbo}L_I(V_{gs} - V_{bs}) \]  
(8.313)

\[ Q_{do} = C_{gdo}W_{eff}(V_{ds} - V_{gs}) \]  
(8.314)

\[ Q_{so} = -C_{gso}W_{eff}V_{gs} \]  
(8.315)

The Ward-Dutton model addresses only the intrinsic channel charge. Two sets of equations model the regions of transistor charge.

**Cut-off region.** The charge model is in the cut-off region when the gate voltage is less than the threshold voltage.

When \( V_{gs} < V_{th} \) and \( (V_{gs} - V_{bs}) < V_{fb} \),

\[ Q_{di} = 0 \]  
(8.316)

\[ Q_{si} = 0 \]  
(8.317)

\[ Q_{gi} = C_{ao}W_{eff}L_{I}V_{g} \]  
(8.318)

When \( V_{gs} < V_{th} \) and \( (V_{gs} - V_{bs}) > V_{fb} \),

\[ Q_{di} = 0 \]  
(8.319)

\[ Q_{si} = 0 \]  
(8.320)

\[ Q_{gi} = C_{ao}W_{L} \cdot \frac{1}{2} \gamma_{s} \sqrt{\frac{V_{g}}{4}} + 4V_{g} \]  
(8.321)

**Linear and saturation regions.** The model uses the same equation for the linear and saturation regions. In the saturation region the voltage \( V_d \) is clamped at the saturation voltage \( V_{dsat} \) when the following condition is true:

\[ V_{d} + \gamma_{s} \sqrt{V_{d}} \geq V_{g} \]  
(8.322)

Then

\[ 2\sqrt{V_{dsat}} = -\gamma_{s} + \sqrt{\gamma_{s}^{2} + 4V_{g}} \]  
(8.323)

\[ V_{d} = (\sqrt{V_{dsat}})^{2} \]  
(8.324)
The intrinsic charge for the linear or saturation region is computed as

\[
Q_{si} = \frac{3}{5} Q_{CSAT} + \frac{3}{10} (Q_C - Q_{CSAT})
\]  
(8.325)

\[
Q_{di} = \frac{23}{5} Q_{CSAT} + \frac{7}{10} (Q_C - Q_{CSAT})
\]  
(8.326)

\[
Q_C = -(Q_{gi} + Q_{bi})
\]  
(8.327)

\[
Q_{CSAT} = -\frac{2}{3} C_{ox} W_{eff} L_1 (V_g - V_s - \gamma_s \sqrt{V_s})
\]  
(8.328)

\[
Q_{gi} = C_{ox} W_{eff} L_1 \left[ V_g - \frac{1}{I_{dc}} \left( \frac{V_g}{g_2} (V_d^2 - V_s^2) - \frac{2}{5} \gamma_s (V_d^{5/2} - V_s^{5/2}) - \frac{1}{3} (V_d^3 - V_s^3) \right) \right]
\]  
(8.329)

\[
Q_{bi} = -\frac{C_{ox} W_{eff} L_1}{I_{dc}} \left[ \frac{2}{3} \gamma_s (V_d^{3/2} - V_s^{3/2}) - \frac{1}{2} (V_d^2 - V_s^2) \right]
\]  
(8.330)

\[
I_{dc} = V_g (V_d - V_s) - \frac{2}{3} \gamma_s (V_d^{3/2} - V_s^{3/2}) - \frac{1}{2} (V_d^2 - V_s^2)
\]  
(8.331)

The diffusion junction capacitance equations for the drain and source are the same as in the Level 1 model. The capacitance is dependent on the drain to bulk or drain to source voltage.

For reverse bias \( V_{bs} < FC \cdot PB \),

\[
C_{bd} = CJ \left( \frac{AS}{1 - \frac{V_{bs}}{PB}} \right) + CJSW \left( \frac{PS}{1 - \frac{V_{bs}}{PB}} \right)
\]  
(8.332)

For forward bias \( V_{bs} > FC \cdot PB \),

\[
C_{bd} = CJ \left( \frac{AS}{(1 - FC)(1 + MJSW)} \right) \left[ 1 - FC(1 + MJ) + \frac{V_{bs}}{PB} MJ \right] + \]

\[
CJSW \left( \frac{PS}{(1 - FC)(1 + MJSW)} \right) \left[ 1 - FC(1 + MJSW) + \frac{V_{bs}}{PB} MJSW \right]
\]  
(8.333)

To obtain the charge function at the junction of the drain or source, integrate the junction capacitance over the range of \( V_{bs} \):

\[
Q = \int C(v) dv
\]  
(8.334)

Now integrate the reverse bias region from \( V_{bs} \) to \( (FC \cdot PB) \).
For reverse bias $V_{bs} < FC \cdot PB$,

$$Q_{Dj} = \int_{V_{bv}}^{FC \cdot PB} C_{bd} V_{bs} dV$$

$$Q_{Dj} = (CJ \cdot AS \cdot PB) \left( \frac{V_{bs} - (FC \cdot PB)}{PB} \right)^{(1-MJ)} - \frac{1}{(1-MJ)} \left( 1 - FC \right)$$

$$Q_{Dj} = (CJSW \cdot PS \cdot PB) \left( \frac{V_{bs} - (FC \cdot PB)}{PB} \right)^{(1-MJSW)} - \frac{1}{(1-MJSW)} \left( 1 - FC \right)$$

For forward bias $V_{bs} > FC \cdot PB$,

$$Q_{Dj} = \int_{V_{bv}}^{V_{bs}} C_{bd} V_{bs} dV$$

$$Q_{Dj} = \frac{CJ \cdot AS \cdot PB}{(10 - FC)^2 - MJ} \left[ (V_{bs} - (FC \cdot PB))(1.0 - FC(2 - MJ)) + \frac{1}{2} \left( \frac{V_{bs}^2}{PB} - FC^2PB \right)(1 - MJ) \right]$$

$$Q_{Dj} = \frac{CJ \cdot SW \cdot PS \cdot PB}{(1.0 - FC)^2 + MJSW} \left( V_{bs} - (FC \cdot PB) \right)(1.0 - FC(2 - MJSW)) + \frac{1}{2} \left( \frac{V_{bs}^2}{PB} - FC^2PB \right)(1 - MJ)$$

Level 3 Equations

The Level 3 MOSFET model is a semi-empirical model developed to handle small-geometry devices.

The drain current in the linear and saturation regions (for $V_{gs} > V_{th}$) is calculated using

$$I_{ds} = \beta \cdot \left[ \frac{1}{1 + \frac{V_{gs} - V_{th}}{\frac{\mu_s V_{max}}{L_1}}} \right] \cdot \left[ V_{gs} - V_{th} - \left( \frac{1 + F_B}{2} \right) V_{ds} \right] \cdot V_{ds}$$

where $V_{ds} = V_{dsat}$ for $V_{ds} > V_{dsat}$ and $V_{dsat}$ is defined as follows:

If $V_{max} \leq 0.0$,

$$V_{dsat} = \frac{V_{gs} - V_{th}}{1 + F_B}$$

If $V_{max} > 0.0$,

$$V_{dsat} = \frac{V_{gs} - V_{th}}{1 + F_B} + \frac{V_{max} \cdot L_1}{\mu_s} \left( \frac{V_{gs} - V_{th}}{1 + F_B} \right)^2 + \left( \frac{V_{max} \cdot L_1}{\mu_s} \right)^2$$
and

$$\mu_s = \frac{\mu_0}{1 + 0(V_{gs} - V_{th})}$$  \hspace{1cm} (8.342)

$$\beta = \frac{W_{eff}}{L_1}\mu_s C_{ox}$$  \hspace{1cm} (8.343)

$$F_B = \frac{\gamma F_s}{4\sqrt{2\phi_F - V_{bs}}} - F_n$$  \hspace{1cm} (8.344)

$$F_s = 1 - \frac{X_d}{X_j} \left( \frac{LD + W}{X_j} \frac{1 - \left( \frac{W_p}{X_j + W_p} \right)^2 - \frac{LD}{X_j}}{\eta} \right)$$  \hspace{1cm} (8.345)

$$W_p = X_{ds}\sqrt{2\phi_F - V_{bs}}$$  \hspace{1cm} (8.346)

$$X_d = \frac{2\varepsilon_s\varepsilon_0}{qN_A}$$  \hspace{1cm} (8.347)

$$\frac{W_c}{X_j} = 0.0631353 + 0.8013292 \frac{W_p}{X_j} - 0.0110777 \left( \frac{W_p}{X_j} \right)^2$$  \hspace{1cm} (8.348)

$$F_n = \frac{\varepsilon_s\varepsilon_0\delta\pi}{2C_{ox}W_{eff}}$$  \hspace{1cm} (8.349)

$$V_{th} = V_{FB} + 2\phi_F - \sigma V_{ds} + \gamma F_s\sqrt{2\phi_F - V_{bs}} + F_n(2\phi_F - V_{bs})$$  \hspace{1cm} (8.350)

$$\sigma = \eta \times 8.15 \times 10^{-22} \frac{\varepsilon_0}{C_{ox} L_1}$$  \hspace{1cm} (8.351)

Channel length modulation is calculated for $V_{ds} > V_{dsat}$ as follows:

$$I_{dsat} = I_{ds}(V_{ds} = V_{dsat})$$  \hspace{1cm} (8.352)

$$G_{dsat} = \frac{\partial I_{ds}}{\partial V_{ds}}(V_{ds} = V_{dsat})$$  \hspace{1cm} (8.353)

$$I_{ds} = \frac{I_{dsat}}{1 - \left( \frac{L_1 - L'}{L_1} \right)}$$  \hspace{1cm} (8.354)
where

\[ L_1 - L' = \left( \frac{E_pX_D^2}{2} \right)^2 + KX_D(V_{ds} - V_{dsat}) - \left( \frac{E_pX_D^2}{2} \right) \]  

(8.355)

\[ E_p = \frac{I_{dsat}}{G_{dsat} + 1} \]  

(8.356)

For \( V_{gs} < V_{on} \), \( I_{ds} \) is modified to include the weak inversion component of the drain current.

\[ V_{on} = V_{th} + \frac{kT}{q} \cdot N \]  

(8.357)

\[ N = 1 + \frac{q}{C_{ox}} \left( \frac{\gamma F_s \sqrt{2\phi_F - V_{bs}} + F_a(2\phi_F - V_{bs})}{2(2\phi_F - V_{bs})} \right) \]  

(8.358)

\[ I'_{ds} = I_{ds} \cdot e^{\frac{q}{nkT} \cdot (V_{gs} - V_{on})} \]  

(8.359)

**Temperature Dependence**

The MOSFET model at Levels 1, 2, and 3 contains temperature-dependent model parameters. If a new temperature is specified with the `.temp` command, then these parameters must be modified before they are used in the current equations. The default temperature is 25 °C, which is equivalent to 300.15 K. The parameters affected by temperature changes are: \( \phi_F \) (Fermi potential), \( E_g \) (energy gap), \( P_B \) (built-in potential of the drain and source), \( \mu_0 \) (mobility), and \( I_s \) (reverse current of the diffused junctions).

The energy gap between the conduction band and the valence band for polysilicon at \( T_{ref} = 300.15 \) K and at the new temperature \( T_{new} \) is:

\[ E_{g,ref} = 1.16 - \left( \frac{7.02 \times 10^{-4} \cdot (T_{ref})^2}{1108 + T_{ref}} \right) \]  

(8.360)

\[ E_{g,new} = 1.16 - \left( \frac{7.02 \times 10^{-4} \cdot (T_{new})^2}{1108 + T_{new}} \right) \]  

(8.361)

The intrinsic doping is adjusted by different equations depending whether the parameter \( PHI \) is defined. When \( PHI \) is not defined,

\[ n_{i,new} = n_i \left( \frac{T_{new}}{T_{ref}} \right)^{3/2} \left( \frac{E_{g,ref}}{E_{g,new}} \right)^{1/2} \]  

(8.362)

When \( PHI \) is defined,

\[ n_{i,new} = N_{sub} \left( \frac{T_{new}}{T_{ref}} \right)^{3/2} \left( \frac{E_{g,ref}}{E_{g,new}} - \frac{2\phi_F}{T_{ref}} \right)^{1/2} \]  

(8.363)
PHI for the new temperature is computed using $n_{i,new}$:

$$\phi_{new} = 2\phi_{F,new} = \left(\frac{2kT_{new}}{q}\right)\ln\left(\frac{N_{sub}}{n_{i,new}}\right)$$  \hfill (8.364)

The conduction factor $K_P$ and the mobility vary with temperature as

$$\frac{\mu_{\theta,new}}{\mu_{\theta,ref}} = \left(\frac{T_{new}}{T_{ref}}\right)^{-3/2}$$  \hfill (8.365)

The parameter $K_P$ or $\beta$ contains the temperature adjustment when computed from $\mu_0$; however, when $K_P$ is entered from the model statement, the value is modified as

$$\frac{\beta_{new}}{\beta_{ref}} = \left(\frac{T_{new}}{T_{ref}}\right)^{-3/2}$$  \hfill (8.366)

The MOSFET substrate junction diode saturation current varies as

$$\frac{I_{s,new}}{I_{s,ref}} = \left(\frac{T_{new}}{T_{ref}}\right)^3\left(\frac{qE_{sat}}{kT_{ref}} - \frac{E_{sat}}{T_{ref}}\right)$$  \hfill (8.367)

The built-in potential $P_B$ is adjusted as

$$P_B = P_B \left(\frac{T_{new}}{T_{ref}}\right) - \left(\frac{2kT_{new}}{q}\right)\ln\left(\frac{T_{new}}{T_{ref}}\right)^{3/2} + \left[\left(\frac{T_{new}}{T_{ref}}\right)E_{g,ref} - E_{g,new}\right]$$  \hfill (8.368)
MOSFET Levels 4 and 13 (BSIM1)

Parameters

```
.model name nmos | pmos level= 4|13 [parameters]
```

*Based on the Berkeley short-channel IGFET model, ©1990 Regents of the University of California. Also see “Additional MOSFET Parameters” on page 492.*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>vfb</td>
<td>Flat band voltage</td>
<td>-0.3</td>
<td>V</td>
</tr>
<tr>
<td>lvfb</td>
<td>Length sensitivity of vfb</td>
<td>0.0</td>
<td>μm · V</td>
</tr>
<tr>
<td>wvfb</td>
<td>Width sensitivity of vfb</td>
<td>0.0</td>
<td>μm · V</td>
</tr>
<tr>
<td>pvfb</td>
<td>WL-product sensitivity of vfb</td>
<td>0.0</td>
<td>μm² · V</td>
</tr>
<tr>
<td>phi</td>
<td>Surface potential (double the Fermi potential)</td>
<td>0.7</td>
<td>V</td>
</tr>
<tr>
<td>lphi</td>
<td>Length sensitivity of phi</td>
<td>0.0</td>
<td>μm · V</td>
</tr>
<tr>
<td>wphi</td>
<td>Width sensitivity of phi</td>
<td>0.0</td>
<td>μm · V</td>
</tr>
<tr>
<td>pphi</td>
<td>WL-product sensitivity of phi</td>
<td>0.0</td>
<td>μm² · V</td>
</tr>
<tr>
<td>k1</td>
<td>$\sqrt{V_{sb}}$ threshold coefficient</td>
<td>0.5</td>
<td>V(^{1/2})</td>
</tr>
<tr>
<td>lk1</td>
<td>Length sensitivity of k1</td>
<td>0.0</td>
<td>μm · V(^{1/2})</td>
</tr>
<tr>
<td>wk1</td>
<td>Width sensitivity of k1</td>
<td>0.0</td>
<td>μm · V(^{1/2})</td>
</tr>
<tr>
<td>pk1</td>
<td>WL-product sensitivity of k1</td>
<td>0.0</td>
<td>μm² · V(^{1/2})</td>
</tr>
<tr>
<td>k2</td>
<td>Linear $V_{sb}$ threshold coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>lk2</td>
<td>Length sensitivity of k2</td>
<td>0.0</td>
<td>μm</td>
</tr>
<tr>
<td>wk2</td>
<td>Width sensitivity of k2</td>
<td>0.0</td>
<td>μm</td>
</tr>
<tr>
<td>pk2</td>
<td>WL-product sensitivity of k2</td>
<td>0.0</td>
<td>μm²</td>
</tr>
<tr>
<td>eta</td>
<td>Linear $V_{ds}$ threshold coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>leta</td>
<td>Length sensitivity of eta</td>
<td>0.0</td>
<td>μm</td>
</tr>
<tr>
<td>weta</td>
<td>Width sensitivity of eta</td>
<td>0.0</td>
<td>μm</td>
</tr>
<tr>
<td>peta</td>
<td>WL-product sensitivity of eta</td>
<td>0.0</td>
<td>μm²</td>
</tr>
<tr>
<td>muz</td>
<td>Low drain field first-order mobility</td>
<td>600</td>
<td>cm²/(V · s)</td>
</tr>
<tr>
<td>lmuz</td>
<td>Length sensitivity of muz</td>
<td>0.0</td>
<td>μm · cm²/(V · s)</td>
</tr>
<tr>
<td>wmuz</td>
<td>Width sensitivity of muz</td>
<td>0.0</td>
<td>μm · cm²/(V · s)</td>
</tr>
</tbody>
</table>
### Parameter Description Default Units

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>pmuz</td>
<td>WL-product sensitivity of ( muz )</td>
<td>0.0</td>
<td>( \mu m^2 \cdot cm^2 / (V \cdot s) )</td>
</tr>
<tr>
<td>dl</td>
<td>Channel length reduction</td>
<td>0.0</td>
<td>( \mu m )</td>
</tr>
<tr>
<td>dw</td>
<td>Channel width reduction</td>
<td>0.0</td>
<td>( \mu m )</td>
</tr>
<tr>
<td>u0</td>
<td>Gate field mobility reduction</td>
<td>0.0</td>
<td>( 1 / V )</td>
</tr>
<tr>
<td>lu0</td>
<td>Length sensitivity of ( u0 )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>wu0</td>
<td>Width sensitivity of ( u0 )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>pu0</td>
<td>WL-product sensitivity of ( u0 )</td>
<td>0.0</td>
<td>( \mu m^2 / V )</td>
</tr>
<tr>
<td>u1</td>
<td>Drain field mobility reduction</td>
<td>0.0</td>
<td>( 1 / V )</td>
</tr>
<tr>
<td>lu1</td>
<td>Length sensitivity of ( u1 )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>wu1</td>
<td>Width sensitivity of ( u1 )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>pu1</td>
<td>WL-product sensitivity of ( u1 )</td>
<td>0.0</td>
<td>( \mu m^2 / V )</td>
</tr>
<tr>
<td>x2mz</td>
<td>( V_{sb} ) correction to low-field 1st-order mobility</td>
<td>0.0</td>
<td>( cm^2 / (V^2 \cdot s) )</td>
</tr>
<tr>
<td>lx2mz</td>
<td>Length sensitivity of ( x2mz )</td>
<td>0.0</td>
<td>( \mu m \cdot cm^2 / (V^2 \cdot s) )</td>
</tr>
<tr>
<td>wx2mz</td>
<td>Width sensitivity of ( x2mz )</td>
<td>0.0</td>
<td>( \mu m \cdot cm^2 / (V^2 \cdot s) )</td>
</tr>
<tr>
<td>px2mz</td>
<td>WL-product sensitivity of ( x2mz )</td>
<td>0.0</td>
<td>( \mu m^2 \cdot cm^2 / (V^2 \cdot s) )</td>
</tr>
<tr>
<td>x2e</td>
<td>( V_{sb} ) correction to linear ( V_{ds} ) threshold coefficient</td>
<td>0.0</td>
<td>( 1 / V )</td>
</tr>
<tr>
<td>lx2e</td>
<td>Length sensitivity of ( x2e )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>wx2e</td>
<td>Width sensitivity of ( x2e )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>px2e</td>
<td>WL-product sensitivity of ( x2e )</td>
<td>0.0</td>
<td>( \mu m^2 / V )</td>
</tr>
<tr>
<td>x3e</td>
<td>( V_{ds} ) correction to linear ( V_{ds} ) threshold coefficient</td>
<td>0.0</td>
<td>( 1 / V )</td>
</tr>
<tr>
<td>lx3e</td>
<td>Length sensitivity of ( x3e )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>wx3e</td>
<td>Width sensitivity of ( x3e )</td>
<td>0.0</td>
<td>( \mu m / V )</td>
</tr>
<tr>
<td>px3e</td>
<td>WL-product sensitivity of ( x3e )</td>
<td>0.0</td>
<td>( \mu m^2 / V )</td>
</tr>
<tr>
<td>x2u0</td>
<td>( V_{sb} ) reduction to gate field mobility reduction</td>
<td>0.0</td>
<td>( 1 / V^2 )</td>
</tr>
<tr>
<td>lx2u0</td>
<td>Length sensitivity of ( x2u0 )</td>
<td>0.0</td>
<td>( \mu m / V^2 )</td>
</tr>
<tr>
<td>wx2u0</td>
<td>Width sensitivity of ( x2u0 )</td>
<td>0.0</td>
<td>( \mu m / V^2 )</td>
</tr>
<tr>
<td>px2u0</td>
<td>WL-product sensitivity of ( x2u0 )</td>
<td>0.0</td>
<td>( \mu m^2 / V^2 )</td>
</tr>
<tr>
<td>x2u1</td>
<td>( V_{sb} ) reduction to drain field mobility reduction</td>
<td>0.0</td>
<td>( 1 / V^2 )</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Default</td>
<td>Units</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>lx2u1</td>
<td>Length sensitivity of x2u1</td>
<td>0.0</td>
<td>μm/V²</td>
</tr>
<tr>
<td>wx2u1</td>
<td>Width sensitivity of x2u1</td>
<td>0.0</td>
<td>μm/V²</td>
</tr>
<tr>
<td>px2u1</td>
<td>$WL$-product sensitivity of x2u1</td>
<td>0.0</td>
<td>μm²/V²</td>
</tr>
<tr>
<td>mus</td>
<td>High drain field mobility</td>
<td>600</td>
<td>cm²/(V·s)</td>
</tr>
<tr>
<td>lmus</td>
<td>Length sensitivity of mus</td>
<td>0.0</td>
<td>μm·cm²/(V·s)</td>
</tr>
<tr>
<td>wmus</td>
<td>Width sensitivity of mus</td>
<td>0.0</td>
<td>μm·cm²/(V·s)</td>
</tr>
<tr>
<td>pmus</td>
<td>$WL$-product sensitivity of mus</td>
<td>0.0</td>
<td>μm·cm²/(V·s)</td>
</tr>
<tr>
<td>x2ms</td>
<td>$V_{bs}$ reduction to high drain field mobility</td>
<td>0.0</td>
<td>cm²/(V²·s)</td>
</tr>
<tr>
<td>lx2ms</td>
<td>Length sensitivity of x2ms</td>
<td>0.0</td>
<td>μm·cm²/(V²·s)</td>
</tr>
<tr>
<td>wx2ms</td>
<td>Width sensitivity of x2ms</td>
<td>0.0</td>
<td>μm·cm²/(V²·s)</td>
</tr>
<tr>
<td>px2ms</td>
<td>$WL$-product sensitivity of x2ms</td>
<td>0.0</td>
<td>μm·cm²/(V·s)</td>
</tr>
<tr>
<td>x3ms</td>
<td>$V_{ds}$ reduction to high drain field mobility</td>
<td>5.0</td>
<td>cm²/(V²·s)</td>
</tr>
<tr>
<td>lx3ms</td>
<td>Length sensitivity of x3ms</td>
<td>0.0</td>
<td>μm·cm²/(V²·s)</td>
</tr>
<tr>
<td>wx3ms</td>
<td>Width sensitivity of x3ms</td>
<td>0.0</td>
<td>μm·cm²/(V²·s)</td>
</tr>
<tr>
<td>px3ms</td>
<td>$WL$-product sensitivity of x3ms</td>
<td>0.0</td>
<td>μm²·cm²/(V²·s)</td>
</tr>
<tr>
<td>x3u1</td>
<td>$V_{ds}$ reduction to drain field mobility reduction</td>
<td>0.0</td>
<td>1/V²</td>
</tr>
<tr>
<td>lx3u1</td>
<td>Length sensitivity of x3u1</td>
<td>0.0</td>
<td>μm/V²</td>
</tr>
<tr>
<td>wx3u1</td>
<td>Width sensitivity of x3u1</td>
<td>0.0</td>
<td>μm/V²</td>
</tr>
<tr>
<td>px3u1</td>
<td>$WL$-product sensitivity of x3u1</td>
<td>0.0</td>
<td>μm²/V²</td>
</tr>
<tr>
<td>tox</td>
<td>Gate oxide thickness</td>
<td>0.02</td>
<td>μm; Å if &gt;1</td>
</tr>
<tr>
<td>temp</td>
<td>Temperature</td>
<td>25.0</td>
<td>°C</td>
</tr>
<tr>
<td>vdd</td>
<td>Critical voltage for high drain field mobility reduction</td>
<td>5.0</td>
<td>V</td>
</tr>
<tr>
<td>cgdo</td>
<td>Gate/drain parasitic capacitance per unit channel width</td>
<td>$1.5 \times 10^{-9}$</td>
<td>F/m</td>
</tr>
<tr>
<td>cgso</td>
<td>Gate/source parasitic capacitance per unit channel width</td>
<td>$1.5 \times 10^{-9}$</td>
<td>F/m</td>
</tr>
</tbody>
</table>
### Parameters and Description

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>cgbo</code></td>
<td>Gate/bulk parasitic capacitance per unit channel length</td>
<td>$2.0 \times 10^{-10}$</td>
<td>F/m</td>
</tr>
<tr>
<td><code>xpart</code></td>
<td>Flag for channel charge partitioning</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td><code>n0</code></td>
<td>Low-field weak inversion gate drive coefficient. A value $\geq 200$ disables weak inversion calculation.</td>
<td>0.5</td>
<td>—</td>
</tr>
<tr>
<td><code>ln0</code></td>
<td>Length sensitivity of <code>n0</code></td>
<td>0.0</td>
<td>$\mu$m</td>
</tr>
<tr>
<td><code>wn0</code></td>
<td>Width sensitivity of <code>n0</code></td>
<td>0.0</td>
<td>$\mu$m</td>
</tr>
<tr>
<td><code>pn0</code></td>
<td>$WL$-product sensitivity of <code>n0</code></td>
<td>0.0</td>
<td>$\mu$m$^2$</td>
</tr>
<tr>
<td><code>nb</code></td>
<td>$V_{sb}$ reduction to <code>n0</code></td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td><code>lnb</code></td>
<td>Length sensitivity of <code>nb</code></td>
<td>0.0</td>
<td>$\mu$m</td>
</tr>
<tr>
<td><code>wnb</code></td>
<td>Width sensitivity of <code>nb</code></td>
<td>0.0</td>
<td>$\mu$m</td>
</tr>
<tr>
<td><code>pnb</code></td>
<td>$WL$-product sensitivity of <code>nb</code></td>
<td>0.0</td>
<td>$\mu$m$^2$</td>
</tr>
<tr>
<td><code>nd</code></td>
<td>$V_{ds}$ reduction to <code>n0</code></td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td><code>lnnd</code></td>
<td>Length sensitivity of <code>nd</code></td>
<td>0.0</td>
<td>$\mu$m</td>
</tr>
<tr>
<td><code>wnd</code></td>
<td>Width sensitivity of <code>nd</code></td>
<td>0.0</td>
<td>$\mu$m</td>
</tr>
<tr>
<td><code>pnd</code></td>
<td>$WL$-product sensitivity of <code>nd</code></td>
<td>0.0</td>
<td>$\mu$m$^2$</td>
</tr>
<tr>
<td><code>xl</code></td>
<td>Mask and etching length change</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td><code>xw</code></td>
<td>Mask and etching width change</td>
<td>0.0</td>
<td>m</td>
</tr>
</tbody>
</table>

*Note: `xpart`, `ln0`, `wn0`, `pn0`, `nb`, `lnb`, `wnb`, `pnb`, `nd`, `lnnd`, `wnd`, `pnd`, `xl`, `xw`, `xw` are parameters in the BSIM1 model.*
MOSFET Level 5 (Maher-Mead)

The Maher-Mead MOSFET model is accurate, physically based, continuous over all transistor regions of operation, including subthreshold, and scales to submicron channel lengths.

Parameters

```
.model name nmos | pmos level=5 [parameters]
```

Also see “Additional MOSFET Parameters” on page 492.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>solver</td>
<td>solver</td>
<td>Nonlinear system solver selector: bisection: \texttt{solver}=0 secant: \texttt{solver}=1</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>tox</td>
<td>Tox</td>
<td>Oxide thickness</td>
<td>1.0e-7</td>
<td>m</td>
</tr>
<tr>
<td>vmax</td>
<td>Vmax</td>
<td>Saturated velocity of electrons or holes</td>
<td>0</td>
<td>m/s</td>
</tr>
<tr>
<td>mu0</td>
<td>(\mu_0)</td>
<td>Zero gate field mobility</td>
<td>Computed.</td>
<td>cm²/V·s</td>
</tr>
<tr>
<td>nsub</td>
<td>Nsub</td>
<td>Substrate doping</td>
<td>—</td>
<td>cm⁻³</td>
</tr>
<tr>
<td>vfb</td>
<td>Vfb</td>
<td>Flat band voltage</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>eghalf</td>
<td>Eghalf</td>
<td>Electric field where mobility = (\mu_0)</td>
<td>(1.0 \times 10^{10})</td>
<td>V/m</td>
</tr>
<tr>
<td>ld</td>
<td>Ld</td>
<td>Length adjustment parameter</td>
<td>(0.75 \times x_j)</td>
<td>m</td>
</tr>
<tr>
<td>wd</td>
<td>Wd</td>
<td>Width adjustment parameter</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>xl</td>
<td>dl</td>
<td>ldel</td>
<td>Xl</td>
<td>Mask and etching length change</td>
</tr>
<tr>
<td>xw</td>
<td>dw</td>
<td>wdel</td>
<td>Xw</td>
<td>Mask and etching width change</td>
</tr>
<tr>
<td>xj</td>
<td>Xj</td>
<td>Junction depth</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>del</td>
<td>del</td>
<td>Channel length reduction per side</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>tnom</td>
<td>Tnom</td>
<td>Reference temperature</td>
<td>global (t_{nom}) (25.0)</td>
<td>°C</td>
</tr>
<tr>
<td>qstol</td>
<td>qstol</td>
<td>Tolerance for nonlinear solution of source charge</td>
<td>(1.0 \times 10^{-8})</td>
<td></td>
</tr>
<tr>
<td>qdtol</td>
<td>qdtol</td>
<td>Tolerance for nonlinear solution of drain charge</td>
<td>(1.0 \times 10^{-8})</td>
<td></td>
</tr>
<tr>
<td>qdmindydx</td>
<td>Lower bound on derivative in nonlinear solution of drain charge</td>
<td>0.01</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Characteristics

The Maher-Mead model is a physically based, charge-controlled model for the DC current, the intrinsic terminal charges, and the transcapacitances in the MOSFET.
The model expresses the current in the MOSFET in terms of the mobile charge per unit area in the channel, and uses a complete set of natural units for velocity, voltage, length, charge, and current. The current-flow equation for the transistor includes both a drift term and a diffusion term, so that the formulation applies equally over the subthreshold, saturation, and “ohmic” regions of transistor operation and includes the effect of velocity saturation.

The model uses physical parameters derived from the fabrication process by direct measurement and from the dimensions of the device. The model agrees closely with measurements on the scaling of current with channel length down to submicron channel lengths.
MOSFET Levels 8, 49 and 53 (BSIM3 Revision 3.3)

Parameters

```plaintext
.model name nmos | pmos level=8 | 49 | 53 [parameters]
```

Levels 49 and 53 are based upon the 3.3 version of Berkeley SPICE. They contain the most commonly used HSPICE ACM, parasitic resistor, and parasitic diode extensions. These extensions are described in “Additional MOSFET Parameters” on page 492.

Level 8 is a strict Berkeley v3.30 implementation with the addition of the HSPICE Effective area and Perimeter calculations and parasitic resistor equations, but not the diode equations.

Refer to the Berkeley manual BSIM3v3.3 MOSFET Model for further model parameters and equations.

Model Selectors

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>binflag</td>
<td>binflag</td>
<td>Flag for use of xwref and xhref parameters</td>
<td>0 (Off)</td>
</tr>
<tr>
<td>capmod</td>
<td>capmod</td>
<td>Flag for short channel capacitance model</td>
<td>1 [v3.0]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0 [v3.1]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3 [≥v3.2]</td>
</tr>
<tr>
<td>mobmod</td>
<td>mobmod</td>
<td>Mobility model selector.</td>
<td>1</td>
</tr>
<tr>
<td>nqsmod</td>
<td>nqsmod</td>
<td>Flag for NQS model. This turns the non-quasistatic model equations on or off, and overrides the model parameter value.</td>
<td>0</td>
</tr>
</tbody>
</table>

**Note:** This option applies to BSIM3 v3.2 and v3.3 only.

nqsmod is a device parameter as well as a device statement.

version | version | Select version of Berkeley BSIM3: 3.0, 3.1, 3.2, or 3.3. | 3.3 |

Basic Model Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>$A_1$</td>
<td>Non-saturation factor 1</td>
<td>0.0 [n]</td>
<td>$V^{-1}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.23 [p]</td>
<td></td>
</tr>
<tr>
<td>a2</td>
<td>$A_2$</td>
<td>Non-saturation factor 2</td>
<td>1.0 [n]</td>
<td>$V^{-1}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.08 [p]</td>
<td></td>
</tr>
<tr>
<td>ags</td>
<td>$A_{gs}$</td>
<td>Gate bias coefficient of Abulk</td>
<td>0.0</td>
<td>$V^{-1}$</td>
</tr>
<tr>
<td>alpha0</td>
<td>$\alpha_0$</td>
<td>1st parameter of impact ionization current</td>
<td>0</td>
<td>mV</td>
</tr>
<tr>
<td>Parameter</td>
<td>Symbol</td>
<td>Description</td>
<td>Default</td>
<td>Units</td>
</tr>
<tr>
<td>-----------</td>
<td>--------</td>
<td>-------------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>b0</td>
<td>B0</td>
<td>Bulk charge effect coefficient for channel width</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>b1</td>
<td>B1</td>
<td>Bulk charge effect width offset</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>beta0</td>
<td>β0</td>
<td>2nd parameter of impact ionization current</td>
<td>30</td>
<td>V</td>
</tr>
<tr>
<td>cdsc</td>
<td>Cdsc</td>
<td>Drain/source and channel coupling capacitance</td>
<td>$2.4 \times 10^{-4}$</td>
<td>F/m²</td>
</tr>
<tr>
<td>cdscb</td>
<td>Cdscb</td>
<td>Body effect coefficient of cdsc</td>
<td>0.0</td>
<td>F/V·m²</td>
</tr>
<tr>
<td>cdscd</td>
<td>Cdscd</td>
<td>Drain-bias sensitivity of cdsc</td>
<td>0.0</td>
<td>F/V·m²</td>
</tr>
<tr>
<td>cit</td>
<td>Cit</td>
<td>Interface state capacitance</td>
<td>0.0</td>
<td>F/m²</td>
</tr>
<tr>
<td>delta</td>
<td>δ</td>
<td>Effective Vds parameter</td>
<td>0.01</td>
<td>—</td>
</tr>
<tr>
<td>drout</td>
<td>DRout</td>
<td>DIBL effect on Rout coefficient</td>
<td>0.56</td>
<td>—</td>
</tr>
<tr>
<td>dsub</td>
<td>Dsub</td>
<td>DIBL effect coefficient in subthreshold region</td>
<td>drout</td>
<td>—</td>
</tr>
<tr>
<td>dvt0</td>
<td>Dvt0</td>
<td>Short channel effect coefficient 0</td>
<td>2.2</td>
<td>—</td>
</tr>
<tr>
<td>dvt0w</td>
<td>Dvt0w</td>
<td>1st coefficient of narrow width effect on vth at small L</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>dvt1</td>
<td>Dvt1</td>
<td>Short channel effect coefficient 1</td>
<td>0.53</td>
<td>—</td>
</tr>
<tr>
<td>dvt1w</td>
<td>Dvt1w</td>
<td>2nd coefficient of narrow width effect on vth at small L</td>
<td>$5.3 \times 10^6$</td>
<td>m⁻¹</td>
</tr>
<tr>
<td>dvt2</td>
<td>Dvt2</td>
<td>Short channel effect coefficient 2</td>
<td>−0.032</td>
<td>V⁻¹</td>
</tr>
<tr>
<td>dvt2w</td>
<td>Dvt2w</td>
<td>Body-bias coefficient of narrow width effect on vth at small L</td>
<td>−0.032</td>
<td>V⁻¹</td>
</tr>
<tr>
<td>eta0</td>
<td>η0</td>
<td>Subthreshold region DIBL coefficient</td>
<td>0.08</td>
<td>—</td>
</tr>
<tr>
<td>etab</td>
<td>ηb</td>
<td>Subthreshold region DIBL coefficient</td>
<td>−0.07</td>
<td>V⁻¹</td>
</tr>
<tr>
<td>is</td>
<td>Is</td>
<td>Bulk saturation current</td>
<td>$1.0 \times 10^{-14}$</td>
<td>A</td>
</tr>
<tr>
<td>k1</td>
<td>K1</td>
<td>1st order bulk effect coefficient</td>
<td>0.53</td>
<td>V^{1/2}</td>
</tr>
<tr>
<td>k2</td>
<td>K2</td>
<td>2nd order bulk effect coefficient</td>
<td>−0.0186</td>
<td>—</td>
</tr>
<tr>
<td>k3</td>
<td>K3</td>
<td>Narrow width effect coefficient</td>
<td>80.0</td>
<td>—</td>
</tr>
<tr>
<td>k3b</td>
<td>K3b</td>
<td>Body effect coefficient of k3</td>
<td>0.0</td>
<td>V⁻¹</td>
</tr>
<tr>
<td>keta</td>
<td>K_{η}</td>
<td>Body bias coefficient of non-uniform depletion width effect</td>
<td>−0.047</td>
<td>V⁻¹</td>
</tr>
</tbody>
</table>
### Parameter | Symbol | Description | Default | Units |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>nch</td>
<td>Nch</td>
<td>Peak doping concentration</td>
<td>$1.7 \times 10^{17}$ cm$^{-3}$</td>
<td></td>
</tr>
<tr>
<td>npeak</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>nfactor</td>
<td>Nfactor</td>
<td>Subthreshold swing coefficient</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>ngate</td>
<td>Ngate</td>
<td>Poly gate doping concentration</td>
<td>0 cm$^{-3}$</td>
<td></td>
</tr>
<tr>
<td>nlx</td>
<td>Nlx</td>
<td>Lateral non-uniform doping effect</td>
<td>$1.74 \times 10^{-7}$ m</td>
<td></td>
</tr>
<tr>
<td>nsub</td>
<td>Nsub</td>
<td>Doping concentration</td>
<td>$6.0 \times 10^{16}$ cm$^{-3}$</td>
<td></td>
</tr>
<tr>
<td>pelm</td>
<td>Pelm</td>
<td>Channel-length modulation effect coefficient</td>
<td>1.3</td>
<td>—</td>
</tr>
<tr>
<td>pdiblc1</td>
<td>Pdiblc1</td>
<td>1st output resistance DIBL effect correction parameter</td>
<td>0.39</td>
<td>—</td>
</tr>
<tr>
<td>pdiblc2</td>
<td>Pdiblc2</td>
<td>2nd output resistance DIBL effect correction parameter</td>
<td>0.0086</td>
<td>—</td>
</tr>
<tr>
<td>pdiblcb</td>
<td>Pdiblcb</td>
<td>Body effect coefficient of DIBL correction parameters</td>
<td>0.0 V$^{-1}$</td>
<td></td>
</tr>
<tr>
<td>prwb</td>
<td>Prwb</td>
<td>Body effect coefficient of $R_{ds}$</td>
<td>0 V$^{1/2}$</td>
<td></td>
</tr>
<tr>
<td>prwg</td>
<td>Prwg</td>
<td>Gate bias coefficient of $R_{ds}$</td>
<td>0 V$^{-1}$</td>
<td></td>
</tr>
<tr>
<td>pscbe1</td>
<td>Pscbe1</td>
<td>Substrate current body effect coefficient 1</td>
<td>$4.24 \times 10^{8}$ V/m</td>
<td></td>
</tr>
<tr>
<td>pscbe2</td>
<td>Pscbe2</td>
<td>Substrate current body effect coefficient 2</td>
<td>$1.0 \times 10^{-5}$ V/m</td>
<td></td>
</tr>
<tr>
<td>pvag</td>
<td>Pvag</td>
<td>$V_g$ dependence of $R_{out}$ coefficient</td>
<td>0.0 V</td>
<td></td>
</tr>
<tr>
<td>rdsw</td>
<td>Rdsw</td>
<td>Source/drain resistance per unit width</td>
<td>0.0 Ω·μm</td>
<td></td>
</tr>
<tr>
<td>tox</td>
<td>Tox</td>
<td>Gate oxide thickness</td>
<td>$1.50 \times 10^{-8}$ m</td>
<td></td>
</tr>
<tr>
<td>u0</td>
<td>$\mu_0$</td>
<td>Low-field mobility at $t_{nom}$</td>
<td>$670$ [n] $250$ [p] cm$^2$V$^{-1}$s$^{-1}$</td>
<td></td>
</tr>
</tbody>
</table>

Note: T-Spice assigns units of m$^{-3}$ to values > 10$^{23}$. 

Note: T-Spice assigns units of m$^{-3}$ to values > 10$^{23}$. 

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ua</td>
<td>$U_a$</td>
<td>Linear $V_{gs}$ dependence of mobility</td>
<td>$2.25 \times 10^{-9}$</td>
<td>m/V</td>
</tr>
<tr>
<td>ub</td>
<td>$U_b$</td>
<td>Quadratic $V_{gs}$ dependence of mobility</td>
<td>$5.87 \times 10^{-18}$</td>
<td>m$^2$/V$^2$</td>
</tr>
<tr>
<td>uc</td>
<td>$U_c$</td>
<td>Body-bias dependence of mobility</td>
<td>$-4.65 \times 10^{-11}$</td>
<td>V$^{-1}$</td>
</tr>
<tr>
<td>vbm</td>
<td>$V_{bm}$</td>
<td>Maximum body voltage</td>
<td>$v3.0: -5.0 \geq v3.1: -3.0$</td>
<td>V</td>
</tr>
<tr>
<td>vfb</td>
<td>$V_{fb}$</td>
<td>DC flatband voltage.</td>
<td>-1</td>
<td>V</td>
</tr>
<tr>
<td>vfbcv</td>
<td>$V_{fbcv}$</td>
<td>Flatband voltage used in charge/capacitance equations when $vfbflag=1$ and $capmod=0$.</td>
<td>-1</td>
<td>V</td>
</tr>
<tr>
<td>vfbflag</td>
<td>$vfbflag$</td>
<td>Selects $vfb$ for $capmod=0$. (Vers. 3.2+)</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>voff</td>
<td>$V_{off}$</td>
<td>Threshold voltage offset</td>
<td>-0.08</td>
<td>V</td>
</tr>
<tr>
<td>voffcv</td>
<td>$V_{offcv}$</td>
<td>C-V parameter for weak to strong inversion transition</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>vsat</td>
<td>$V_{sat}$</td>
<td>Saturation velocity at $tnom$</td>
<td>$8.0 \times 10^4$</td>
<td>m/s</td>
</tr>
<tr>
<td>vth0</td>
<td>$V_{th0}$</td>
<td>Threshold voltage.</td>
<td>0.7 [n] ~0.7 [p]</td>
<td>V</td>
</tr>
<tr>
<td>w0</td>
<td>$W_0$</td>
<td>Narrow width effect coefficient</td>
<td>$2.5 \times 10^{-6}$</td>
<td>m</td>
</tr>
<tr>
<td>wr</td>
<td>$W_r$</td>
<td>Width offset from $W_{off}$ for $R_{ds}$ calculation</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>xj</td>
<td>$X_j$</td>
<td>Junction depth</td>
<td>$1.5 \times 10^{-7}$</td>
<td>m</td>
</tr>
</tbody>
</table>
## AC and Capacitance Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>cf</td>
<td>$C_f$</td>
<td>Fringing field capacitance</td>
<td>Computed</td>
<td>F/m</td>
</tr>
<tr>
<td>cgbo</td>
<td>$C_{gbo}$</td>
<td>Gate/bulk overlap capacitance per unit channel length</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>cgdl</td>
<td>$C_{gd}$</td>
<td>Light doped drain-gate region overlap capacitance</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>cgdo</td>
<td>$C_{gdo}$</td>
<td>Gate/drain overlap capacitance per unit channel width</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>cgsl</td>
<td>$C_{gs}$</td>
<td>Light doped source-gate region overlap capacitance</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>cgso</td>
<td>$C_{gso}$</td>
<td>Gate/source overlap capacitance per unit channel width</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>ckappa</td>
<td>$C_\kappa$</td>
<td>Coefficient for lightly doped region overlap capacitance</td>
<td>0.6</td>
<td>F/m</td>
</tr>
<tr>
<td>clc</td>
<td>$CLC$</td>
<td>Constant term for short-channel model</td>
<td>$0.1 \times 10^{-6}$</td>
<td>m</td>
</tr>
<tr>
<td>cle</td>
<td>$CLE$</td>
<td>Exponential term for short-channel model</td>
<td>0.6</td>
<td>—</td>
</tr>
<tr>
<td>xpart</td>
<td>$Xpart$</td>
<td>Flag for channel charge partitioning</td>
<td>0</td>
<td>—</td>
</tr>
</tbody>
</table>

## Length and Width Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>dlc</td>
<td>$DLC$</td>
<td>Length offset fitting parameter from C-V</td>
<td>$lint$</td>
</tr>
<tr>
<td>dwb</td>
<td>$dWb$</td>
<td>Coefficient of $W_{eff}$ substrate body bias dependence</td>
<td>0.0</td>
</tr>
<tr>
<td>dwc</td>
<td>$DWC$</td>
<td>Width offset fitting parameter from C-V</td>
<td>$wint$</td>
</tr>
<tr>
<td>dwg</td>
<td>$dWg$</td>
<td>Coefficient of $W_{eff}$ gate dependence</td>
<td>0.0</td>
</tr>
<tr>
<td>lint</td>
<td>$Lint$</td>
<td>Length offset fitting parameter from I-V without bias</td>
<td>0.0</td>
</tr>
<tr>
<td>ll</td>
<td>$Ll$</td>
<td>Coefficient of length dependence for length offset</td>
<td>0.0</td>
</tr>
<tr>
<td>llc</td>
<td>$Llc$</td>
<td>Coefficient of length dependence for C-V channel width offset</td>
<td>0.0</td>
</tr>
<tr>
<td>lln</td>
<td>$Lln$</td>
<td>Power of length dependence for length offset</td>
<td>1.0</td>
</tr>
<tr>
<td>lw</td>
<td>$Lw$</td>
<td>Coefficient of width dependence for length offset</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Chapter 8: Device Models

Temperature Parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>at</td>
<td>$At$</td>
<td>Temperature coefficient of $v_{sat}$</td>
<td>$3.3 \times 10^4$ m/s</td>
</tr>
<tr>
<td>CTA</td>
<td>CTC</td>
<td>$CTA$</td>
<td>Temperature coefficient for $C_j$</td>
</tr>
<tr>
<td>CTP</td>
<td>$CTP$</td>
<td>Temperature coefficient for $C_{jsw}$</td>
<td>0.0 deg$^{-1}$</td>
</tr>
<tr>
<td>EG</td>
<td>$Eg(0)$</td>
<td>Energy gap at 0° K (Si: 1.166, Ge: 0.74, and GaAs: 1.52)</td>
<td>1.16 eV</td>
</tr>
<tr>
<td>GAP1</td>
<td>$GAP1$</td>
<td>Coefficient in energy gap temperature equation (Si: $4.73 \times 10^{-4}$, Ge: $4.77 \times 10^{-4}$, and GaAs: $5.41 \times 10^{-4}$)</td>
<td>$7.02 \times 10^{-4}$ eV/deg</td>
</tr>
</tbody>
</table>
### Process Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>em</td>
<td>Em</td>
<td>Maximum electric field</td>
<td>$4.1 \times 10^7$</td>
<td>V/m</td>
</tr>
</tbody>
</table>
Chapter 8: Device Models

MOSFET Levels 8, 49 and 53 (BSIM3 Revision 3.3)

NonQuasi-Static Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>gamma1</td>
<td>γ1</td>
<td>$V_{th}$ coefficient 1</td>
<td>Computed</td>
<td>V$^{1/2}$</td>
</tr>
<tr>
<td>gamma2</td>
<td>γ2</td>
<td>$V_{th}$ coefficient 2</td>
<td>0.0</td>
<td>V$^{1/2}$</td>
</tr>
<tr>
<td>vbx</td>
<td>Vbx</td>
<td>$V_{th}$ transition body voltage</td>
<td>Computed</td>
<td>V</td>
</tr>
<tr>
<td>xt</td>
<td>$X_t$</td>
<td>Doping depth</td>
<td>1.55 $\times$ 10$^{-7}$</td>
<td>m</td>
</tr>
</tbody>
</table>

Equations

For the complete set of equations describing the Level 49 model, see the BSIM3v3 Manual (Ko et al. 1995).

Drain Current

$$I_{ds} = \frac{I_{ds0(V_{gs,0})}}{1 + \frac{V_{ds} - V_{ds,eff}}{V_A}} \left(1 + \frac{V_{ds} - V_{ds,eff}}{V_{ASCBE}}\right)$$ (8.369)

where

$$I_{ds0} = \frac{W_{eff} L_{eff} C_{ox} V_{gs,eff} (1 - A_{bulk} \frac{V_{ds,eff}}{2(V_{gs,eff} + 2V_1)})}{1 \frac{V_{ds,eff}}{E_{sat} L_{eff}}}$$ (8.370)

$$V_{ds,eff} = V_{dsat} - \frac{1}{2}(V_{dsat} - V_{ds} - \delta + \sqrt{(V_{dsat} - V_{ds})^2 + 4\delta V_{dsat}})$$ (8.371)

$$E_{sat} = \frac{2v_{sat}}{V_{eff}}$$ (8.372)

$$v_1 = \frac{kT}{q}$$ (8.373)

Variables for which equations are not given here are as follows.

$V_A$ Early voltage
\( V_{ASCBE} \) 
Early voltage due to substrate current-induced body effect

\( V_{gs, eff} \) 
Effective \( V_{gs} - V_{th} \) (\( V_{th} \): effective threshold voltage)

\( \mu_{eff} \) 
Effective mobility

\( A_{bulk} \) 
Bulk charge effect factor

\( V_{dsat} \) 
Drain saturation voltage

### Gate Charge

\[
Q_g = -(Q_{acc} + Q_{sub0} + \delta Q_{sub} + Q_{inv}) \tag{8.374}
\]

Variables for which equations are not given here are as follows.

\( Q_{acc} \) 
Channel majority or accumulation charge

\( Q_{sub0} \) 
Substrate charge at \( V_{ds} = 0 \)

\( Q_{sub} \) 
Non-uniform substrate charge in presence of drain bias

\( Q_{inv} \) 
\( Q_s + Q_d \) = channel minority or inversion charge
MOSFET Levels 9 and 50 (Philips MOS 9)

Philips MOS Model 9 is a compact MOS-transistor model, intended for the simulation of circuit behaviour with emphasis on analog applications. The model gives a complete description of all transistor-action related quantities: nodal currents and charges, noise-power spectral densities and weak-avalanche currents. The equations describing these quantities are based on the gradual-channel approximation with a number of first-order corrections for small-size effects. The consistency is maintained by using the same carrier-density and electrical-field expressions in the calculation of all model quantities. MOS Model 9 only provides a model for the intrinsic transistor. Junction charges and leakage currents are not included. They are covered by the separate Juncap model.

The MOS 9 model is fully documented in Philips MOS Model, level 903. For further detailed information about the MOS 9 model, please refer to the Philips NXP Compact Model web page:


Parameters

The MOS 9 model uses the following syntax.

```
.model name nmos|pmos level=[9|50] | model=modelname [parameters]
```

T-Spice includes support for MOS 9 versions 902 and 903, and for both geometrical and electrical based model parameter sets.

The available modelname values for the MOS 9 model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mos902</td>
<td>Mos 9 level 902, geometrical</td>
</tr>
<tr>
<td>mos902e</td>
<td>Mos 9 level 902, electrical</td>
</tr>
<tr>
<td>mos903 (default)</td>
<td>Mos 9 level 903, geometrical</td>
</tr>
<tr>
<td>mos903e</td>
<td>Mos 9 level 903, electrical</td>
</tr>
</tbody>
</table>
MOSFET Levels 11 and 63 (Philips MOS 11)

MOS Model 11 has been developed as the successor of MOS Model 9. It is a symmetrical, surface-potential-based model, giving an accurate physical description of the transition from weak to strong inversion. MOS 11 includes an accurate description of all physical effects important for modern and future CMOS technologies, such as:

- mobility reduction
- bias-dependent series resistance
- velocity saturation
- conductance effects (CLM, DIBL, etc.)
- gate leakage current
- gate-induced drain leakage
- gate depletion
- quantum-mechanical effects
- bias-dependent overlap capacitances

The description of the source-bulk and drain-bulk junction diode is not included in MOS Model 11. The behaviour of these junction diodes is modelled by the Juncap model. This model has to be added between the source and bulk node and between the drain and bulk node. The MOS 11 model is fully documented in Philips MOS 11.

The MOS 11 model is fully documented in Philips MOS 11, level 1102. For further detailed information about the MOS 11 model, please refer to the Philips NXP Compact Model web page:


Parameters

The MOS 11 model uses the following syntax.

```
.model name nmos | pmos level=[11|63] | model=modelname [parameters]
```

T-Spice includes support for Mos 11 versions 1100, 1101, and 1102. Each of these versions, in turn, offers a selection of electrical or geometrical based parameterization, modeling of self-heating effects, and model binning.

The available `modelname` values for the Mos 11 model are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mos1100</td>
<td>Mos 11 level 1100, geometrical</td>
</tr>
<tr>
<td>mos1100e</td>
<td>Mos 11 level 1100, electrical</td>
</tr>
<tr>
<td>mos1101</td>
<td>Mos 11 level 1101, electrical</td>
</tr>
<tr>
<td>mos1101t</td>
<td>Mos 11 level 1101, electrical, self-heating</td>
</tr>
<tr>
<td>mos11010</td>
<td>Mos 11 level 1101, geometrical</td>
</tr>
<tr>
<td>mos11010t</td>
<td>Mos 11 level 1101, geometrical, self-heating</td>
</tr>
<tr>
<td>mos11011</td>
<td>Mos 11 level 1101, geometrical, binning</td>
</tr>
<tr>
<td>mos11011t</td>
<td>Mos 11 level 1101, geometrical, binning, self-heating</td>
</tr>
<tr>
<td>mos1102</td>
<td>Mos 11 level 1102, electrical</td>
</tr>
<tr>
<td>mos1102t</td>
<td>Mos 11 level 1102, electrical, self-heating</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td>mos11020 (default)</td>
<td>Mos 11 level 1102, geometrical</td>
</tr>
<tr>
<td>mos11020t</td>
<td>Mos 11 level 1102, geometrical, self-heating</td>
</tr>
<tr>
<td>mos11021</td>
<td>Mos 11 level 1102, geometrical, binning</td>
</tr>
<tr>
<td>mos11021t</td>
<td>Mos 11 level 1102, geometrical, binning, self-heating</td>
</tr>
</tbody>
</table>
MOSFET Levels 14 and 54 (BSIM4 Revision 5)

Parameters

.model name nmos | pmos level=14 | 54 [parameters]

Levels 14 and 54 are fully compliant with the original UC Berkeley release of BSIM4 Revision 5. For standard model parameters and equations, refer to the Berkeley manual BSIM4.5.0 MOSFET Model. Specific device instance statements and their parameters are shown in the following section.

Syntax

General MOSFET device parameters (length, width, drain, source, etc.) are described in the device statement chapter under “MOSFET (m)” on page 193. In the case of device values which have corresponding model values, the device settings override the model settings.

Device instance parameters for BSIM4 are as follows:

m name drain gate source bulk model [l=L] [w=W] [ad=Ad] [pd=Pd] [as=As]
[ps=Ps] [nrd=Nrd] [nrs=Nrs] [M=M] [acnqsmod=acnqsmod] [geomod=geomod]
[min=min] [nrd=nrd] [nrs=nrs] [rbdb=rbdb] [rbodymod=rbodymod] [rbpb=rbpb]
[rbpd=rbpd] [rbps=rbps] [rbsb=rbsb] [rgatemod=rgatemod] [rgeomod=rgeomod]
[trnqsmod=trnqsmod] [sa=sa] [sb=sb] [sd=sd]

acnqsmod
AC small-signal NQS model selector

geomod
Geometry-dependent parasitics model selector - specifying how the end S/D diffusions are connected

min
Whether to minimize the number of drain or source diffusions for even-number finger devices

nf
Number of device fingers

nrd
Number of drain diffusion squares

nrs
Number of source diffusion squares

rbdb
Resistance connected between dbNode and bNode

rbodymod
Substrate resistance network model selector

rbpb
Resistance connected between bNodePrime and bNode

rbpd
Resistance connected between bNodePrime and dbNode

rbps
Resistance connected between bNodePrime and sbNode

rbsb
Resistance connected between sbNode and bNode

rgatemod
Gate resistance model selector

rgeomod
Source/drain diffusion resistance and contact model selector - specifying the end S/D contact type: point, wide or merged, and how S/D parasitics resistance is computed

trnqsmod
Transient NQS model selector

sa
Distance between OD edge to Poly from one side
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sb</td>
<td>Distance between OD edge to Poly from other side</td>
</tr>
<tr>
<td>sd</td>
<td>Distance between neighboring fingers</td>
</tr>
</tbody>
</table>
Level 15 is a thin-film transistor (TFT) amorphous silicon (a-Si) model developed at Rensselaer Polytechnic Institute (RPI).

This model is based on the universal charge control concept, which allows for currents and the large and small-signal parameters to be written as continuous functions of the applied bias, providing smooth transitions between the different operating regimes. Interpolation techniques are applied to the equations to unify the model.

Note that contrary to devices such as SOI MOSFETs, self-heating in a-Si TFT leads to an increase in current, since the carrier mobility increases with the temperature.

Physical effects included in MOS Model 15 include:
Above threshold:
- Modified charge control model; induced charge trapped in localized states
- Field effect mobility becoming a function of gate bias
- Band mobility dominated by lattice scattering
Below threshold:
- Fermi level located in deep localized states
- Relate position of Fermi level, including the deep DOS back to the gate bias
Empirical expression for current at large negative gate biases for hole-induced leakage current.
Interpolation techniques are applied to the equations to unify the model.

**Note:**
The T-Spice implementation of the RPI a-Si TFT model supports model binning. The additional model parameters for this are $l_{min}$, $l_{max}$, $w_{min}$, $w_{max}$, $x_l$, $x_l_{ref}$, $x_w$, and $x_w_{ref}$, described in the section “Additional MOSFET Parameters” on page 492.

**Parameters**

This is a 3-terminal model. Because no bulk node exists, no parasitic drain-bulk or source-build diodes are appended to the model. You can specify a fourth node but it will not affect simulation results. The drain and source areas and perimeters are not used either, since the model equation is based solely upon the width and length.

Device instance parameters for MOS level 15 are as follows:

```
mname drain gate source model [L=l] [W=w] [M=m][TEMP=t]
```

The parameter definitions ($L$, $W$, and $M$) are the MOSFET standards (see “MOSFET (m)” on page 193), and $TEMP$ is the device temperature ($C$).

For further detailed information please refer to the Rensselaer Polytechnic Institute research papers in the T-Spice models folder.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHASAT</td>
<td>Saturation modulation parameter</td>
<td>0.6</td>
<td>—</td>
</tr>
<tr>
<td>CGDO</td>
<td>Gate-drain overlap capacitance per meter channel width</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Default</td>
<td>Units</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------</td>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>CGSO</td>
<td>Gate-source overlap capacitance per meter channel width</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>DEFO</td>
<td>Dark Fermi level position</td>
<td>0.6</td>
<td>eV</td>
</tr>
<tr>
<td>DELTA</td>
<td>Transition width parameter</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>EL</td>
<td>Activation energy of the hole leakage current</td>
<td>0.35</td>
<td>eV</td>
</tr>
<tr>
<td>EMU</td>
<td>Field effect mobility activation energy</td>
<td>0.06</td>
<td>eV</td>
</tr>
<tr>
<td>EPS</td>
<td>Relative dielectric constant of substrate</td>
<td>11</td>
<td>-</td>
</tr>
<tr>
<td>EPSI</td>
<td>Relative dielectric constant of gate insulator</td>
<td>7.4</td>
<td>-</td>
</tr>
<tr>
<td>GAMMA</td>
<td>Power law mobility parameter</td>
<td>0.4</td>
<td>-</td>
</tr>
<tr>
<td>GMIN (GO, G0)</td>
<td>Minimum density of deep states</td>
<td>1E23</td>
<td>m⁻³eV⁻¹</td>
</tr>
<tr>
<td>IOL</td>
<td>Zero bias leakage current parameter</td>
<td>3E-14</td>
<td>A</td>
</tr>
<tr>
<td>KASAT</td>
<td>Temperature coefficient of ALPHASAT</td>
<td>0.006</td>
<td>1/°C</td>
</tr>
<tr>
<td>KVT</td>
<td>Threshold voltage temperature coefficient</td>
<td>-0.036</td>
<td>V/°C</td>
</tr>
<tr>
<td>LAMDBA</td>
<td>Output conductance parameter</td>
<td>0.0008</td>
<td>1/V</td>
</tr>
<tr>
<td>M (MSAT)</td>
<td>Knee shape parameter</td>
<td>2.5</td>
<td>-</td>
</tr>
<tr>
<td>MUBAND</td>
<td>Conduction band mobility</td>
<td>0.001</td>
<td>m²/Vs</td>
</tr>
<tr>
<td>RD</td>
<td>Drain resistance</td>
<td>0.0</td>
<td>μ</td>
</tr>
<tr>
<td>RS</td>
<td>Source resistance</td>
<td>0.0</td>
<td>μ</td>
</tr>
<tr>
<td>SIGMA0 (sigma0)</td>
<td>Minimum leakage current parameter</td>
<td>1E-14</td>
<td>A</td>
</tr>
<tr>
<td>TNOM (TREF)</td>
<td>Parameter measurement temperature</td>
<td>global Tnom</td>
<td>oC</td>
</tr>
<tr>
<td>TOX</td>
<td>Thin-oxide thickness</td>
<td>1E-7</td>
<td>m</td>
</tr>
<tr>
<td>V0</td>
<td>Characteristic voltage for deep states</td>
<td>0.12</td>
<td>V</td>
</tr>
<tr>
<td>VAA</td>
<td>Characteristic voltage for field effect mobility</td>
<td>7.5E3</td>
<td>V</td>
</tr>
<tr>
<td>VDSL</td>
<td>Hole leakage current drain voltage parameter</td>
<td>7</td>
<td>V</td>
</tr>
<tr>
<td>VFB</td>
<td>Flat band voltage</td>
<td>-3</td>
<td>V</td>
</tr>
<tr>
<td>VGSL (VGL)</td>
<td>Hole leakage current gate voltage parameter</td>
<td>7</td>
<td>V</td>
</tr>
<tr>
<td>VMIN</td>
<td>Convergence parameter</td>
<td>0.3</td>
<td>V</td>
</tr>
<tr>
<td>VTO (VT0)</td>
<td>Zero-bias threshold voltage</td>
<td>0.0</td>
<td>V</td>
</tr>
</tbody>
</table>
Chapter 8: Device Models

MOSFET Levels 15 and 61 (RPI Amorphous-Si TFT Model)

Equivalent Circuit

![Equivalent Circuit Diagram]
MOSFET Levels 16 and 62 (RPI Poly-Si TFT Model, 1.0 and 2.0)

Level 16 is a thin-film transistor (TFT) poly-silicon (Poly-Si) model developed at Rensselaer Polytechnic Institute (RPI).

MOS level 16 improves on existing devices models by including the necessary dependencies to make it scalable from long-channel to short-channel devices.

This model accounts for some effects that are specific to poly-Si TFTs, including the kink effect, the increase of the field-effect mobility as the gate voltage is increased in moderate inversion, the off-current, the DIBL (Drain Induced Barrier Lowering) and velocity saturation effects, as well as series resistances. The DIBL effect is more pronounced in poly-Si TFTs than in crystalline MOSFETs and cannot be neglected even for long-channel devices.

More specifically,

- Field effect mobility becomes a function of gate bias
- Effective mobility that accounts for trap states, for low $V_{gs}$ using a power law, for high $V_{gs}$ a constant.
- A unified DC model that includes all four regimes for channel lengths down to 4 m—leakage (thermionic emission), subthreshold (diffusion-like model), above threshold (c-Si-like, with mFet) and kink (impact ionization with feedback).
- An AC model that accurately reproduces $C_{ge}$ frequency dispersion
- An automatic scaling of model parameters to accurately model a wide range of device geometries

Note: The T-Spice implementation of the RPI Poly-Si TFT model supports model binning. The additional model parameters for this are $l_{min}$, $l_{max}$, $w_{min}$, $w_{max}$, $x_l$, $x_{lref}$, $x_w$, and $x_{wref}$, described in the section “Additional MOSFET Parameters” on page 492.

Parameters

This is a 3-terminal model. MOS level 16 does not use a bulk node; there are no corresponding drain-bulk or source-bulk diodes to be modeled. You can specify a fourth node but it does it not affect simulation results.

Device instance parameters for MOS level 16 are as follows:

```plaintext
mname drain gate source model [L=l] [W=w] [NRS=nrs] [NRD=nrd] [M=m] [TEMP=t]
```

The parameter definitions (L, W, NRS, NRD, M, etc.) are the MOSFET standard ones (see “MOSFET (m)” on page 193). TEMP is the device temperature (C).

For further detailed information please refer to the Rensselaer Polytechnic Institute papers in the T-Spice models folder.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASAT</td>
<td>Proportionality constant of $V_{sat}$</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>AT</td>
<td>DIBL parameter 1</td>
<td>3E-8</td>
<td>m/V</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Default</td>
<td>Units</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------------------------------------------------</td>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>BLK</td>
<td>Leakage barrier lowering constant</td>
<td>0.001</td>
<td>-</td>
</tr>
<tr>
<td>BT</td>
<td>DIBL parameter 2</td>
<td>1.9E-6</td>
<td>m-V</td>
</tr>
<tr>
<td>CAPMOD</td>
<td>Capacitance model selector</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>CGDO</td>
<td>Gate-drain overlap capacitance per meter channel width</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>CGSO</td>
<td>Gate-source overlap capacitance per meter channel width</td>
<td>0.0</td>
<td>F/m</td>
</tr>
<tr>
<td>DASAT</td>
<td>Temperature coefficient of ASAT</td>
<td>0</td>
<td>1/°C</td>
</tr>
<tr>
<td>DD</td>
<td>Vds field constant</td>
<td>1400 Å</td>
<td>m</td>
</tr>
<tr>
<td>DELTA</td>
<td>Transition width parameter</td>
<td>4.0</td>
<td>-</td>
</tr>
<tr>
<td>DG</td>
<td>Vgs field constant</td>
<td>2000 Å</td>
<td>m</td>
</tr>
<tr>
<td>DMU1</td>
<td>Temperature coefficient of MU1</td>
<td>0</td>
<td>cm²/Vs °C</td>
</tr>
<tr>
<td>DVT</td>
<td>The difference between VON and the threshold voltage</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>DVTO</td>
<td>Temperature coefficient of VTO</td>
<td>0</td>
<td>V/°C</td>
</tr>
<tr>
<td>EB</td>
<td>Barrier height of diode</td>
<td>0.68</td>
<td>EV</td>
</tr>
<tr>
<td>ETA (ETAI)</td>
<td>Subthreshold ideality factor</td>
<td>7</td>
<td>-</td>
</tr>
<tr>
<td>ETAC0</td>
<td>Capacitance subthreshold ideality factor at zero drain bias</td>
<td>ETA</td>
<td>ETAC00</td>
</tr>
<tr>
<td>ETAC00</td>
<td>Capacitance subthreshold coefficient of drain bias</td>
<td>0</td>
<td>1/V</td>
</tr>
<tr>
<td>I0 (CLK)</td>
<td>Leakage scaling constant</td>
<td>6.0</td>
<td>A/m</td>
</tr>
<tr>
<td>I00</td>
<td>Reverse diode saturation current</td>
<td>150</td>
<td>A/m</td>
</tr>
<tr>
<td>LASAT</td>
<td>Coefficient for length dependence of ASAT</td>
<td>0</td>
<td>M</td>
</tr>
<tr>
<td>LKINK</td>
<td>Kink effect constant</td>
<td>19E-6</td>
<td>M</td>
</tr>
<tr>
<td>MC</td>
<td>Capacitance knee shape parameter</td>
<td>3.0</td>
<td>-</td>
</tr>
<tr>
<td>MK (MKINK)</td>
<td>Kink effect exponent</td>
<td>1.3</td>
<td>-</td>
</tr>
<tr>
<td>MMU (M)</td>
<td>Low field mobility exponent</td>
<td>1.7</td>
<td>-</td>
</tr>
<tr>
<td>MU0</td>
<td>High field mobility</td>
<td>100</td>
<td>cm²/Vs</td>
</tr>
<tr>
<td>MU1</td>
<td>Low field mobility parameter</td>
<td>0.0022</td>
<td>cm²/Vs</td>
</tr>
<tr>
<td>MUS</td>
<td>Subthreshold mobility</td>
<td>1.0</td>
<td>cm²/Vs</td>
</tr>
<tr>
<td>RD</td>
<td>Drain resistance</td>
<td>0.0</td>
<td>W</td>
</tr>
<tr>
<td>RDX</td>
<td>Resistance in series with Cgd (RF)</td>
<td>0</td>
<td>W</td>
</tr>
<tr>
<td>RS</td>
<td>Source resistance</td>
<td>0.0</td>
<td>W</td>
</tr>
<tr>
<td>RSH</td>
<td>Sheet resistance</td>
<td>0</td>
<td>Ω/sq</td>
</tr>
<tr>
<td>RSX (RI)</td>
<td>Resistance in series with Cgs</td>
<td>0</td>
<td>W</td>
</tr>
</tbody>
</table>
Version 2.0 Parameters

Version 1.0 is the standard release of the Poly-Si model, equivalent to the AimSpice level 16 PSIA2 model. Version 2.0 (selected using the model parameter VERSION=2.0) is a newer release, and includes an intrinsic resistance model, alternate channel length modulation equations, and DIBL (drain induced barrier lowering) equations.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>TNOM (TREF)</td>
<td>Parameter measurement temperature</td>
<td>27</td>
<td>°C</td>
</tr>
<tr>
<td>TOX</td>
<td>Thin-oxide thickness</td>
<td>1.0e-7</td>
<td>m</td>
</tr>
<tr>
<td>VFB</td>
<td>Flat band voltage</td>
<td>-0.1</td>
<td>V</td>
</tr>
<tr>
<td>VKINK</td>
<td>Kink effect voltage</td>
<td>9.1</td>
<td>V</td>
</tr>
<tr>
<td>VON</td>
<td>On-voltage</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>VTO</td>
<td>Zero-bias threshold voltage</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>INTDSNOD</td>
<td>Extrinsic resistance mode selector</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>ISUBMOD</td>
<td>Channel length modulation equation selector</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>Channel length modulation parameter</td>
<td>.048</td>
<td>m/V</td>
</tr>
<tr>
<td>LS</td>
<td>Channel length modulation parameter</td>
<td>35e-9</td>
<td>m/V</td>
</tr>
<tr>
<td>ME (MS)</td>
<td>Long channel saturation transition parameter</td>
<td>2.5</td>
<td>-</td>
</tr>
<tr>
<td>META</td>
<td>Eta floating body parameter</td>
<td>1.0</td>
<td>-</td>
</tr>
<tr>
<td>MSS</td>
<td>Vdse transition parameter</td>
<td>1.5</td>
<td>-</td>
</tr>
<tr>
<td>THETA</td>
<td>Mobility degradation parameter</td>
<td>0</td>
<td>1/V</td>
</tr>
<tr>
<td>VMAX</td>
<td>Saturation velocity</td>
<td>4e4</td>
<td>m/s</td>
</tr>
<tr>
<td>VP</td>
<td>Channel length modulation parameter</td>
<td>0.2</td>
<td>V</td>
</tr>
<tr>
<td>VSIGMA</td>
<td>Above threshold DIBL parameter</td>
<td>0.2</td>
<td>V</td>
</tr>
<tr>
<td>VSIGMAT</td>
<td>Above threshold DIBL parameter</td>
<td>1.7</td>
<td>V</td>
</tr>
</tbody>
</table>
Equivalent Circuit
MOSFET Level 20 (Philips MOS 20)

MOS Model 20 is a compact LDMOS model, which combines the MOSFET operation of the channel region with that of the drift region under the thin gate oxide. As such, it is aimed as a successor of MOS Model 9 in series with MOS Model 31. MOS Model 20 has especially been developed to improve the convergence behaviour during simulation, by having the voltage at the transition from the channel region to the drift region calculated inside the model itself.

The MOS 20 model is fully documented in Philips MOS 20, level 2001. For further detailed information about the MOS 20 model, please refer to the Philips NXP Compact Model web page:

http://www.nxp.com/models/hv_models/model20/index.html

Parameters

The MOS 20 model uses the following syntax.

```
.model name nmos | pmos level=20 | model=modelname [parameters]
```

T-Spice includes support for MOS 20 version 2001 with geometrical and electrical model parameterization, and self-heating effects.

The available `modelname` values for the MOS 20 model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mos2001 (default)</td>
<td>Mos 20 level 2001, geometrical</td>
</tr>
<tr>
<td>mos2001e</td>
<td>Mos 20 level 2001, electrical</td>
</tr>
<tr>
<td>mos2001et</td>
<td>Mos 20 level 2001, electrical, self-heating</td>
</tr>
</tbody>
</table>
MOSFET Level 28 (Extended BSIM1)

```
.model name nmos | pmos level=28 [parameters]
```

T-Spice supports MOSFET model level 28, based on the Berkeley short-channel IGFET models. The Extended BSIM1 level 28 model is a proprietary Tanner Research extension to the core BSIM1 model developed at Berkeley. The extensions attempt to fix many of the problems in the original model equations, including:

- Negative output conductance in the saturation region of operation.
- Discontinuities in the output conductance at the transition between the linear and saturation regions of operation.
- Discontinuities in the subthreshold current and transconductance characteristics near threshold.

In addition, the core equations have been enhanced to correct deficiencies in the original model equations:

- Temperature compensation equations have been added for proper handling of temperature effects.
- Effective length-width product scaling factors have been added for use with the BSIM3 revision 3 scaling equations.

**Note:**

T-Spice accepts BSIM1 model parameters entered in the level 13 and level 28 conventions used by HSPICE™. Level 13 model parameters are translated to standard BSIM1 model (see “MOSFET Levels 4 and 13 (BSIM1)” on page 447.) Level 28 model parameters are automatically translated to the extended BSIM1 model as shown below.

### Parameters

Following are the Tanner Extended BSIM1 MOSFET model parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>tempmod</td>
<td>Temperature model selector. <strong>tempmod=0</strong> disables the temperature compensation equation. <strong>tempmod&gt;0</strong> enables the temperature compensation equation.</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>ute</td>
<td>Temperature exponent for mobility</td>
<td>1.5</td>
<td>—</td>
</tr>
<tr>
<td>lute</td>
<td>Length sensitivity of ute</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>wute</td>
<td>Width sensitivity of ute</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>pute</td>
<td><strong>WL</strong>-product sensitivity of ute</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>kt1</td>
<td>Temperature coefficient for flat band voltage</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>lkt1</td>
<td>Length sensitivity of kti</td>
<td>0.0</td>
<td>μm·V</td>
</tr>
<tr>
<td>wkt1</td>
<td>Width sensitivity of kti</td>
<td>0.0</td>
<td>μm·V</td>
</tr>
<tr>
<td>pkt1</td>
<td><strong>WL</strong>-product sensitivity of kti</td>
<td>0.0</td>
<td>μm²·V</td>
</tr>
</tbody>
</table>

---

1 HSPICE(®) is a registered trademark of Synopsys, Inc.
### Equations

#### Process Parameters

Process parameters express the sensitivity of the BSIM1 electrical parameters to device length, width, and the product of length and width (\(WL\)-product). The SPICE names of process parameters are derived from the related electrical parameter names by prefixing the letters \(l\), \(w\), and \(p\).

The actual value of an electrical parameter \(P\) is

\[
P = P_0 + \frac{P_L}{L_i - \Delta L} + \frac{P_W}{W_i - \Delta W} + \frac{P_P}{(L_i - \Delta L)(W_i - \Delta W)}
\]  

(8.375)

where \(P_L\), \(P_W\), and \(P_P\) denote \(P\)'s length, width, and product sensitivity parameters, respectively.

#### Drain Current

Drain current is modeled by three equations, representing the three regions of MOSFET operation: cutoff, linear, and saturation.
Cutoff region: \( V_{gs} \leq V_{th} \)

\[
I_{ds} = 0
\]  \hspace{1cm} (8.376)

Linear region: \( V_{gs} > V_{th} \) and \( 0 < V_{ds} < V_{d,sat} \)

\[
I_{ds} = \frac{\mu_0}{1 + U_0(V_{gs} - V_{th})} \cdot \frac{C_{ox} W_{eff}}{L_{eff}} \cdot \left[ (V_{gs} - V_{th})V_{ds} - \frac{d}{2}V_{ds}^2 \right]
\]  \hspace{1cm} (8.377)

where

\[
a = \frac{1 + gK_j}{2\sqrt{\phi_s - V_{bs}}} > 1
\]  \hspace{1cm} (8.378)

and

\[
g = 1 - \frac{1}{1.744 + 0.8364(\phi_s - V_{bs})}
\]  \hspace{1cm} (8.379)

Saturation region: \( V_{gs} > V_{th} \) and \( V_{ds} \geq V_{d,sat} \)

\[
I_{ds} = \frac{\mu_0}{1 + U_0(V_{gs} - V_{th})} \cdot \frac{C_{ox} W_{eff}}{L_{eff}} \cdot \frac{2aK}{(V_{gs} - V_{th})^2}
\]  \hspace{1cm} (8.380)

where

\[
K = \frac{1 + v_c + \sqrt{1 + 2v_c}}{2}
\]  \hspace{1cm} (8.381)

and

\[
v_c = \frac{U_1}{L_{eff}} \cdot \frac{(V_{gs} - V_{th})}{a}
\]  \hspace{1cm} (8.382)

In the linear and saturation regions,

\[
V_{d,sat} = \frac{V_{gs} - V_{th}}{a\sqrt{K}}
\]  \hspace{1cm} (8.383)

\[
U_0(V_{ds}, V_{bs}) = U_{0z} + U_{0b}V_{bs}
\]  \hspace{1cm} (8.384)

\[
U_f(V_{ds}, V_{bs}) = U_{1z} + U_{1b}V_{bs} + U_{1d}(V_{ds} - V_{dd})
\]  \hspace{1cm} (8.385)
\( \mu_0 \) is computed by quadratic interpolation given three conditions:

\[
\mu_0(V_{ds} = 0) = \mu_z + \mu_{zb} V_{bs} \tag{8.386}
\]

\[
\mu_0(V_{ds} = V_{dd}) = \mu_s + \mu_{sb} V_{bs} \tag{8.387}
\]

and the sensitivity of \( \mu_0 \) to the drain bias at \( V_{ds} = V_{dd} \).

**Subthreshold Current**

The total drain current is modeled as the sum of two components: the *strong inversion* component \( I_{ds,s} \), equivalent to the drain current modeled in equations (8.376) through (8.387), and the weak inversion component \( I_{ds,w} \):

\[
I_{ds,w} = \frac{I_{exp} I_{limit}}{I_{exp} + I_{limit}} \tag{8.388}
\]

where

\[
I_{exp} = \mu_0 C'_{ox} \frac{W_{eff} (kT)^2}{L_{eff}} \frac{e^{1.8 e^{q(V_g - V_a)/n k T}}}{q^2} e^{q(V_g - V_a)/n k T} (1 - e^{-q V_a/k T}) \tag{8.389}
\]

and

\[
I_{limit} = \frac{\mu_0 C'_{ox} W_{eff} (3kT)^2}{2 L_{eff} q^2} \tag{8.390}
\]

The subthreshold parameter \( n \) is modeled as

\[
n(V_{ds}, V_{bs}) = n_a + n_b V_{bs} + n_d V_{ds} > 0.5 \tag{8.391}
\]
MOSFET Level 30 (Philips MOS 30)

MOS Model 30 is a long channel JFET/MOSFET model developed to describe the drift region of LDMOS, EPMOS and VDMOS devices.

Note: MOS Model 30 has been replaced with the MOS 31 model, and is provided for historical and compatibility purposes only. It’s use is not recommended.

The MOS 30 model is fully documented in Philips MOS Model, level 3002.

Parameters

The MOS 30 model uses the following syntax.

```plaintext
.model name nmos | pmos level=30 | model=mos3002 [parameters]
```

T-Spice includes support for MOS 30 version 3002 with electrical model parameterization.
MOSFET Level 31 (Philips MOS 31)

MOS Model 31 is a physics based transistor model to be used in circuit simulation and IC-design of analog high-voltage applications. The model describes the electrical behaviour of a junction-isolated accumulation/depletion-type MOSFET. The model is used as the drain extension of high-voltage MOS devices, like the Lateral Double-diffused MOS (LDMOS), the Vertical Double-diffused MOS (VDMOS), and the Extended MOS transistors. Physical effects included in MOS Model 31:
- Both accumulation and depletion underneath the gate oxide;
- Depletion from the substrate (a pn-junction);
- Pinch-off effects;
- Velocity saturation; and
- Temperature scaling.

The MOS 31 model is fully documented in Philips MOS Model, level 3100. For further detailed information about the MOS 31 model, please refer to the Philips NXP Compact Model web page:


Parameters

The MOS 31 model uses the following syntax.

```
.model name nmos | pmos level=31 | model=modelname [parameters]
```

T-Spice includes support for MOS 31 version 3100 with and without self-heating effects.

The available `modelname` values for the MOS 31 model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mos3100</td>
<td>Mos 31 level 3100, electrical</td>
</tr>
<tr>
<td>mos3100t</td>
<td>Mos 31 level 3100, electrical, self-heating</td>
</tr>
</tbody>
</table>
MOSFET Level 40 (Philips MOS 40)

MOS Model 40 is a physics based transistor model to be used in circuit simulation and IC-design of analogue high-voltage applications processed in Silicon-on-Insulator (SOI). The model describes the electrical behavior of an accumulation/depletion-type MOSFET in SOI. The model is used as drain extension of high-voltage MOS devices, like the Lateral Double-diffused MOS (LDMOS), the Vertical Double-diffused MOS (VDMOS), and the Extended MOS transistors.

Physical effects in MOS Model 40 include:
- Both accumulation and depletion underneath the gate oxide;
- Both accumulation and depletion from the substrate (an oxide layer);
- Pinch-off effects;
- Velocity saturation; and
- Temperature scaling.

The MOS 40 model is fully documented in Philips MOS Model, Level 40. For further detailed information about the MOS 40 model, please refer to the Philips NXP Compact Model web page:

http://www.nxp.com/models/hv_models/model40/index.html

Parameters

The MOS 40 model uses the following syntax.

```
.model name nmos | pmos level=40 | model=modelname [parameters]
```

T-Spice includes support for MOS 40 version 40 with and without self-heating effects.

The available `modelname` values for the MOS 40 model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mos40 (default)</td>
<td>Mos 40 level 40, electrical</td>
</tr>
<tr>
<td>mos40t</td>
<td>Mos 40 level 40, electrical, self-heating</td>
</tr>
</tbody>
</table>
MOSFET Levels 44 and 55 (EKV Revision 2.6)

Parameters

```
.model name nmos| pmos level=44|55 [parameters]
```

For model parameters, model equations and device instance statements and their parameters, refer to the Swiss Federal Institute of Technology manual The EPFL-EKV MOSFET Model Equations for Simulation.
MOSFET Level 47 (BSIM3 Revision 2)

Parameters

```
.model name nmos|pmos level=47 [parameters]
```

Based on the Berkeley short-channel IGFET model, ©1990 Regents of the University of California.

Also see “Additional MOSFET Parameters” on page 492.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>subthmod</td>
<td>subthmo d</td>
<td>Subthreshold model selector</td>
<td>2</td>
<td>—</td>
</tr>
<tr>
<td>satmod</td>
<td>satmod</td>
<td>Saturation model selector</td>
<td>2</td>
<td>—</td>
</tr>
<tr>
<td>bulkmod</td>
<td>bulkmo d</td>
<td>Bulk charge effect model selector</td>
<td>1 [n]</td>
<td>—</td>
</tr>
<tr>
<td>mobmod</td>
<td>mobmo d</td>
<td>Mobility model selector</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>tox</td>
<td>Tox</td>
<td>Gate oxide thickness</td>
<td>$1.50 \times 10^{-8}$</td>
<td>m</td>
</tr>
<tr>
<td>cdsc</td>
<td>Cdsc</td>
<td>Drain/source and channel coupling capacitance</td>
<td>$2.4 \times 10^{-4}$</td>
<td>F/m$^2$</td>
</tr>
<tr>
<td>cdscb</td>
<td>Cdscb</td>
<td>Body effect coefficient of cdsc</td>
<td>0.0</td>
<td>F/V·m$^2$</td>
</tr>
<tr>
<td>cit</td>
<td>Cit</td>
<td>Interface state capacitance</td>
<td>0.0</td>
<td>F/m$^2$</td>
</tr>
<tr>
<td>nfactor</td>
<td>Nfactor</td>
<td>Swing coefficient</td>
<td>1</td>
<td>—</td>
</tr>
<tr>
<td>xj</td>
<td>Xj</td>
<td>Junction depth</td>
<td>1.50e-7</td>
<td>m</td>
</tr>
<tr>
<td>vsat</td>
<td>vsat</td>
<td>Saturation velocity at tnom</td>
<td>$8.0 \times 10^{-6}$</td>
<td>cm/s</td>
</tr>
<tr>
<td>at</td>
<td>At</td>
<td>Temperature coefficient of vsat</td>
<td>$3.3 \times 10^{-4}$</td>
<td>m/s</td>
</tr>
<tr>
<td>a0</td>
<td>A0</td>
<td>Non-uniform depletion width effect coefficient</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>a1</td>
<td>A1</td>
<td>Non-saturation factor 1</td>
<td>0.0 [n]</td>
<td>1/V</td>
</tr>
<tr>
<td>a2</td>
<td>A2</td>
<td>Non-saturation factor 2</td>
<td>1.0 [n]</td>
<td>—</td>
</tr>
<tr>
<td>keta</td>
<td>Keta</td>
<td>Body bias coefficient of non-uniform depletion width effect</td>
<td>$-0.047$</td>
<td>1/V</td>
</tr>
<tr>
<td>vghigh</td>
<td>Vhigh</td>
<td>High bound of transition region</td>
<td>0.12</td>
<td>V</td>
</tr>
<tr>
<td>vglow</td>
<td>Vglow</td>
<td>Low bound of transition region</td>
<td>$-0.12$</td>
<td>V</td>
</tr>
<tr>
<td>nsub</td>
<td>Nsub</td>
<td>Doping concentration</td>
<td>$6.0 \times 10^{16}$</td>
<td>cm$^{-3}$ (≤10$^{20}$)</td>
</tr>
</tbody>
</table>

(>10$^{20}$)
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>nch</td>
<td>Npeak</td>
<td>Peak doping concentration</td>
<td>$1.7 \times 10^{17}$</td>
<td>cm$^{-3}$ (≤$10^{23}$), m$^{-3}$ (&gt;10$^{23}$)</td>
</tr>
<tr>
<td>npeak</td>
<td>Ngate</td>
<td>Gate doping concentration</td>
<td>0.0</td>
<td>cm$^{-3}$</td>
</tr>
<tr>
<td>gamma1</td>
<td>$\gamma_1$</td>
<td>$V_{th}$ coefficient</td>
<td>0.0</td>
<td>V$^{1/2}$</td>
</tr>
<tr>
<td>gamma2</td>
<td>$\gamma_2$</td>
<td>$V_{th}$ coefficient 2</td>
<td>0.0</td>
<td>V$^{1/2}$</td>
</tr>
<tr>
<td>vbx</td>
<td>$V_{bx}$</td>
<td>$V_{th}$ transition body voltage</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>vbi</td>
<td>$V_{bi}$</td>
<td>Drain/source junction built-in potential</td>
<td>0.0</td>
<td>V</td>
</tr>
<tr>
<td>vbm</td>
<td>$V_{bm}$</td>
<td>Maximum body voltage</td>
<td>–5.0</td>
<td>V</td>
</tr>
<tr>
<td>xt</td>
<td>$X_t$</td>
<td>Doping depth</td>
<td>$1.55 \times 10^{-7}$</td>
<td>m</td>
</tr>
<tr>
<td>phi</td>
<td>$\phi$</td>
<td>Strong inversion surface potential</td>
<td>Computed</td>
<td>V</td>
</tr>
<tr>
<td>litl</td>
<td>Litl</td>
<td>Depth of current path</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>em</td>
<td>$E_m$</td>
<td>Maximum electric field</td>
<td>$4.1 \times 10^{7}$</td>
<td>V/m</td>
</tr>
<tr>
<td>k1</td>
<td>$K_1$</td>
<td>Bulk effect coefficient 1</td>
<td>0.0</td>
<td>V$^{1/2}$</td>
</tr>
<tr>
<td>kt1</td>
<td>$K_{t1}$</td>
<td>Temperature coefficient of $V_{th}$</td>
<td>–0.11</td>
<td>V</td>
</tr>
<tr>
<td>kt1l</td>
<td>$K_{t1l}$</td>
<td>Channel length sensitivity of $K_{t1}$</td>
<td>0.0</td>
<td>V·m</td>
</tr>
<tr>
<td>kt2</td>
<td>$K_{t2}$</td>
<td>Body bias coefficient of $K_{t1}$</td>
<td>0.022</td>
<td>—</td>
</tr>
<tr>
<td>k2</td>
<td>$K_2$</td>
<td>Bulk effect coefficient 2</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>k3</td>
<td>$K_3$</td>
<td>Narrow width effect coefficient</td>
<td>80.0</td>
<td>—</td>
</tr>
<tr>
<td>k3b</td>
<td>$K_{3b}$</td>
<td>Body effect coefficient of $K_3$</td>
<td>0.0</td>
<td>1/N</td>
</tr>
<tr>
<td>w0</td>
<td>$W_0$</td>
<td>Narrow width effect coefficient</td>
<td>$2.5 \times 10^{-6}$</td>
<td>m</td>
</tr>
<tr>
<td>nlx</td>
<td>$N_{lx}$</td>
<td>Lateral non-uniform doping effect</td>
<td>$1.74 \times 10^{-7}$</td>
<td>m</td>
</tr>
<tr>
<td>dvt0</td>
<td>$D_{vt0}$</td>
<td>Short channel effect coefficient 0</td>
<td>2.2</td>
<td>—</td>
</tr>
<tr>
<td>dvt1</td>
<td>$D_{vt1}$</td>
<td>Short channel effect coefficient 1</td>
<td>0.53</td>
<td>—</td>
</tr>
<tr>
<td>dvt2</td>
<td>$D_{vt2}$</td>
<td>Short channel effect coefficient 2</td>
<td>–0.032</td>
<td>1/N</td>
</tr>
<tr>
<td>drout</td>
<td>$D_{Rout}$</td>
<td>DIBL effect on $R_{out}$ coefficient</td>
<td>0.56</td>
<td>—</td>
</tr>
<tr>
<td>dsub</td>
<td>$D_{sub}$</td>
<td>DIBL effect coefficient in subthreshold region</td>
<td>$D_{Rout}$</td>
<td>—</td>
</tr>
<tr>
<td>vtho</td>
<td>$V_{th}$</td>
<td>Threshold voltage</td>
<td>0.7 [n]</td>
<td>V</td>
</tr>
<tr>
<td>vth0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ua</td>
<td>$U_a$</td>
<td>Linear $V_{gs}$ dependence of mobility</td>
<td>$2.25 \times 10^{-9}$</td>
<td>m/V</td>
</tr>
<tr>
<td>ua1</td>
<td>$U_{a1}$</td>
<td>Temperature coefficient of $U_a$</td>
<td>$4.31 \times 10^{-9}$</td>
<td>m/V</td>
</tr>
<tr>
<td>ub</td>
<td>$U_b$</td>
<td>Quadratic $V_{gs}$ dependence of mobility</td>
<td>$5.87 \times 10^{-19}$</td>
<td>m$^2$/V$^2$</td>
</tr>
<tr>
<td>ub1</td>
<td>$U_{b1}$</td>
<td>Temperature coefficient of $U_b$</td>
<td>$–7.61 \times 10^{-18}$</td>
<td>m$^2$/V$^2$</td>
</tr>
<tr>
<td>uc</td>
<td>$U_c$</td>
<td>Body-bias dependence of mobility</td>
<td>0.0465</td>
<td>1/N</td>
</tr>
<tr>
<td>uc0</td>
<td>$U_{c0}$</td>
<td>Mobility coefficient</td>
<td>0.0</td>
<td>V$^2$/m$^2$</td>
</tr>
</tbody>
</table>
### MOSFET Level 47 (BSIM3 Revision 2)

#### Table of Device Models Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>( u_{c1} )</td>
<td>( U_{c1} )</td>
<td>Temperature coefficient of ( u_c )</td>
<td>–0.056</td>
<td>( 1/\text{N} )</td>
</tr>
<tr>
<td>( u_0 )</td>
<td>( \mu_0 )</td>
<td>Low-field mobility at ( t_{\text{nom}} )</td>
<td>( \begin{array}{c} 670.0 \text{ [n]} \ 250.0 \text{ [p]} \end{array} )</td>
<td>( \begin{array}{c} \text{cm}^2/\text{N}\cdot\text{s} \ (\geq 1) \text{m}^2/\text{N}\cdot\text{s} \ (&lt; 1) \text{m}^2/\text{V}\cdot\text{s} \end{array} )</td>
</tr>
<tr>
<td>( u_{te} )</td>
<td>( \mu_{te} )</td>
<td>Temperature coefficient of mobility</td>
<td>–1.5</td>
<td>—</td>
</tr>
<tr>
<td>( v_{off} )</td>
<td>( V_{\text{off}} )</td>
<td>Threshold voltage offset</td>
<td>–0.11</td>
<td>V</td>
</tr>
<tr>
<td>( v_{fb} )</td>
<td>( V_{fb} )</td>
<td>Flat band voltage</td>
<td>–1.0</td>
<td>V</td>
</tr>
<tr>
<td>( d_l</td>
<td>d_t )</td>
<td>( D_l )</td>
<td>Channel length reduction</td>
<td>0.0</td>
</tr>
<tr>
<td>( d_w</td>
<td>d_w )</td>
<td>( D_w )</td>
<td>Channel width reduction</td>
<td>0.0</td>
</tr>
<tr>
<td>( l_{\text{mlt}} )</td>
<td>( L_{\text{mlt}} )</td>
<td>Length shrink factor</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>( w_{\text{mlt}} )</td>
<td>( W_{\text{mlt}} )</td>
<td>Width shrink factor</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>( x_{l}</td>
<td>x_{l}</td>
<td>x_{\text{del}} )</td>
<td>( X_l )</td>
<td>Mask and etching length reduction factor</td>
</tr>
<tr>
<td>( x_{w}</td>
<td>x_{w}</td>
<td>x_{\text{del}} )</td>
<td>( X_w )</td>
<td>Mask and etching width reduction factor</td>
</tr>
<tr>
<td>( t_{\text{nom}}</td>
<td>t_{\text{ref}} )</td>
<td>( T_{\text{nom}} )</td>
<td>Temperature</td>
<td>25.0</td>
</tr>
<tr>
<td>( c_{gso}</td>
<td>c_{gsom} )</td>
<td>( C_{gso} )</td>
<td>Gate/source overlap capacitance per unit channel width</td>
<td>0.0</td>
</tr>
<tr>
<td>( c_{gdo}</td>
<td>c_{gdom} )</td>
<td>( C_{gdo} )</td>
<td>Gate/drain overlap capacitance per unit channel width</td>
<td>0.0</td>
</tr>
<tr>
<td>( c_{gbo}</td>
<td>c_{gbom} )</td>
<td>( C_{gbo} )</td>
<td>Gate/bulk overlap capacitance per unit channel length</td>
<td>0.0</td>
</tr>
<tr>
<td>( x_{\text{part}} )</td>
<td>( X_{\text{part}} )</td>
<td>Flag for channel charge partitioning</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>( r_{\text{ds}} )</td>
<td>( R_{\text{ds}} )</td>
<td>Source/drain resistance per unit width</td>
<td>0.0</td>
<td>( \Omega \cdot \mu\text{m} )</td>
</tr>
<tr>
<td>( r_{\text{ds}} )</td>
<td>( R_{\text{ds}} )</td>
<td>Source/drain contact resistance</td>
<td>0.0</td>
<td>( \Omega )</td>
</tr>
<tr>
<td>( l_{\text{dd}} )</td>
<td>( L_{\text{DD}} )</td>
<td>Total source/drain LDD region length</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>( e_{\eta} )</td>
<td>( E_{\eta} )</td>
<td>Effective drain voltage coefficient</td>
<td>0.3</td>
<td>—</td>
</tr>
<tr>
<td>( e_{\eta 0} )</td>
<td>( E_{\eta 0} )</td>
<td>Subthreshold region DIBL coefficient</td>
<td>0.08</td>
<td>—</td>
</tr>
<tr>
<td>( e_{\eta b} )</td>
<td>( E_{\eta b} )</td>
<td>Subthreshold region DIBL coefficient</td>
<td>–0.07</td>
<td>( 1/\text{N} )</td>
</tr>
<tr>
<td>( p_{c\text{clm}} )</td>
<td>( P_{c\text{clm}} )</td>
<td>Channel-length modulation effect coefficient</td>
<td>1.3</td>
<td>—</td>
</tr>
<tr>
<td>( p_{\text{dibl1}} )</td>
<td>( P_{\text{dibl1}} )</td>
<td>DIBL effect coefficient 1</td>
<td>0.39</td>
<td>—</td>
</tr>
<tr>
<td>( p_{\text{dibl2}} )</td>
<td>( P_{\text{dibl2}} )</td>
<td>DIBL effect coefficient 2</td>
<td>0.0086</td>
<td>—</td>
</tr>
<tr>
<td>( p_{\text{scbe1}} )</td>
<td>( P_{\text{scbe1}} )</td>
<td>Substrate current body effect coefficient 1</td>
<td>( 4.24 \times 10^8 )</td>
<td>V/m</td>
</tr>
<tr>
<td>( p_{\text{scbe2}} )</td>
<td>( P_{\text{scbe2}} )</td>
<td>Substrate current body effect coefficient 2</td>
<td>( 1.0 \times 10^{-5} )</td>
<td>m/V</td>
</tr>
<tr>
<td>( p_{\text{vag}} )</td>
<td>( P_{\text{vag}} )</td>
<td>( V_g ) dependence of ( R_{\text{out}} ) coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
</tbody>
</table>
Equations


Drain Current

In the linear region:

\[ I_{ds} = \frac{I_{dslin0}}{1 + \frac{R_{ds}I_{dslin0}}{V_{ds}}} \]  

(8.392)

where

\[ I_{dslin0} = \mu_{eff} C_{ox} \left( \frac{W}{L} \right) \left( 1 + \frac{1}{V_{ds}} \right) \left( V_{gst} - A_{bulk} \frac{V_{ds}}{2} \right) V_{ds} \]  

(8.393)

In the saturation region:

\[ I_{ds} = I_{dsat} \left( 1 + \frac{V_{ds} - V_{dual}}{V_{A}} \right) \left( 1 + \frac{V_{ds} - V_{dual}}{V_{ASCBE}} \right) \]  

(8.394)

where

\[ I_{dsat} = W_{sat} C_{ox} (V_{gst} - A_{bulk} V_{dsat}) \]  

(8.395)

In the subthreshold region:

If \( \text{subthmod} = 0 \):

\[ I_{ds} = 0 \]  

(8.396)

If \( \text{subthmod} = 1 \):

\[ I_{ds} = \frac{I_{limit} I_{exp}}{I_{limit} + I_{exp}} (1 - e^{-V_{ds}/V_{sat}}) \]  

(8.397)
where

\[
I_{\text{limit}} = \frac{9}{2} \mu_0 C_{\text{dep}} \left( \frac{W}{L} \right) V_{\text{in}}^2 \cdot e^{\left( \frac{E_{\text{sat}} + E_{\text{sh}} \cdot V_{\text{sh}}}{nV_{\text{in}}} \right)}
\]  
(8.398)

\[
I_{\text{exp}} = \mu_0 C_{\text{dep}} \left( \frac{W}{L} \right) V_{\text{in}}^2 \cdot e^{\left( \frac{V_{\text{gs}} - V_{\text{sh}} - V_{\text{sat}} + (E_{\text{sat}} + E_{\text{sh}} \cdot V_{\text{sh}})^2}{nV_{\text{in}}} \right)}
\]  
(8.399)

If \( \text{subthmod} = 2 \):

\[
I_{ds} = I_{s0}(1 - e^{-V_{ds}/V_{\text{in}}}) \cdot e^{\left( \frac{V_{gs} - V_{\text{sh}} + (E_{\text{sat}} + E_{\text{sh}} \cdot V_{\text{sh}})^2}{nV_{\text{in}}} \right)}
\]  
(8.400)

where

\[
I_{s0} = \mu_0 \frac{W}{L} \left( \frac{q e_{\text{q}} \nu_{\text{peak}}}{2 \phi_s} \right) \cdot V_{\text{in}}^2
\]  
(8.401)

In the transition region:

\[
I_{ds} = (1 - t)^2 I_{\text{dslow}} + 2(1 - t)I_p + t^2 I_{\text{dhigh}}
\]  
(8.402)

where

\[
t = \left( \frac{V_p - V_{\text{gslow}}}{V_{\text{gslow}} - 2V_p + V_{\text{ghigh}}} \right) \times \left( \frac{1 + \left( V_{\text{gslow}} - 2V_p + V_{\text{ghigh}} \right)(V_{\text{gst}} - V_{\text{gslow}})}{(V_p - V_{\text{gslow}})^2} - 1 \right)
\]  
(8.403)

Variables for which equations are not given here are as follows:

\begin{align*}
VA & \quad \text{Early voltage} \\
VASCBE & \quad \text{Early voltage due to substrate current-induced body effect} \\
\mu_{\text{eff}} & \quad \text{Effective mobility} \\
Abulk & \quad \text{Bulk charge effect factor} \\
V_{\text{dsat}} & \quad \text{Drain saturation voltage}
\end{align*}
MOSFET Level 57 / 70 (BSIM3SOI and BSIM4SOI)

BSIMSOI is an international standard model for silicon-on-insulator (SOI) circuit design, adjunct to the BSIM3v3 framework.

Parameters

```
.model name nmos | pmos level=57 [parameters]
```

For standard model parameters and equations, please refer to the Berkeley manuals included with the Tanner T-Spice documentation. Specific device instance statements and their parameters are listed in the following section.

Syntax

General MOSFET device parameters (length, width, drain, source, etc.) are described in the device statement chapter under “MOSFET (m)” on page 193. In the case of device values which have corresponding model values, the device settings override the model settings.

Device instance parameters for the BSIMSOI MOSFET model are as follows:

```
mname d g s e [p] [b] [t] model [L=l] [W=w] [P=p] [B=b] [T=t] [AD=ad] [AS=as] [PD=pd] [PS=ps] [NRS=nrs] [NRD=nrd] [NRB=nrb] [M=M] [OFF][BJTOFF=bjtoff] [IC=ds gs bs es ps initial voltages] [RTH0=rth0] [CTH0=cth0] [NBC=nbc] [NSEG=nseg] [PDBCP=pdbcp] [PSBCP=psbcp] [AGBCP=agbcp] [AEBCP=aebcp] [VBSUSR=vbsusr] [TNODEOUT] [RGATEMOD=rgatemod] [SOIMOD=soimod] [FRBODY=frbody]
```

d Drain node
g Front gate node
s Source node
e Back gate or substrate node
p Optional external body contact node
b Optional internal body node
t Optional temperature node
ad Drain diffusion area
aebcp Parasitic gate-to-body overlap area for body contact
agbcp Parasitic perimeter length for body contact at drain side
as Source diffusion area
bjtoff Turn off BJT current if equal to 1
cth0 Thermal capacitance per unit width:
  - if not specified, CTH0 is extracted from model card
  - if specified, it will override the one in model card
frbody Layout-dependent body resistance coefficient
SOI Modes

There are three modes in BSIMSOI, soimod = 0, 1 or 2. BSIMPD (soimod = 0) can be used to model the PD SOI device, where the body potential is independent of DVbi (VBS > DVbi). Therefore the calculation of DVbi is skipped in this mode. On the other hand, the ideal FD model (soimod = 2) is for the FD device with body potential equal to DVbi. Hence the calculation of body current/charge, which is essential to the PD model, is skipped. For the unified SOI model (soimod = 1), however, both DVbi and body current/charge are calculated to capture the floating-body behavior exhibited in FD devices. This unified model covers both BSIMPD and the ideal FD model.
**Body and Temperature Nodes**

There are three optional nodes, P, B and T nodes. Nodes P and B are used for body contact devices. If the TNODEOUT flag is not set, when you specify four nodes, this element is a four terminal device, i.e., floating body. If you specify five nodes, the fifth node represents the external body contact node (P). There is a body resistance between the internal body node and the P node. In both these cases, an internal body node is created, but it is not accessible in the circuit deck. However, if you specify six nodes, the fifth node will represent the P node and the sixth node will represent the internal body node (B). This configuration is useful for distributed body resistance simulation.

If the TNODEOUT flag is set, the last node is interpreted as the temperature node. In this case, when you specify five nodes, it is a floating body. If you specify six nodes, it is a body-contacted case. Finally, if you specify seven nodes, it is a body-contacted case with an accessible internal body node. The temperature node is useful for thermal coupling simulation.
MOSFET Level 100 (Penn State & Philips PSP Model)

The PSP model is a new compact MOSFET model, which has been jointly developed by Philips Research and Penn State University, and is able to accurately model present-day and upcoming deep-submicron bulk CMOS technologies.

The PSP model is a symmetrical, surface-potential-based model, giving an accurate physical description of the transition from weak to strong inversion. The PSP model includes an accurate description of all physical effects important for modern and future CMOS technologies, such as:

- mobility reduction
- bias-dependent series resistance
- velocity saturation
- conductance effects (CLM, DIBL, etc.)
- lateral doping gradient effect
- mechanical stress related to STI
- gate leakage current
- gate-induced drain leakage
- gate depletion
- quantum-mechanical effects
- bias-dependent overlap capacitances

In addition, it gives an accurate description of charges and currents and their first-order derivatives (transconductance, conductance, capacitances), but also of their higher-order derivatives. In other words, it gives an accurate description of MOSFET distortion behaviour, and as such the PSP model is suitable for digital, analog as well as RF circuit design.

The PSP model is fully documented in the technical note PSP 102.0 from Pennsylvania State University and Philips Research. For further detailed information about the PSP model, please refer to the Philips MOS Model PSP web page:


Parameters

The PSP model uses the following syntax.

```
.model name nmos|pmos level=100|1000 | model=modelname [parameters]
```

T-Spice includes support for PSP version 100.1 with both electrical and geometrical model parameterization.

The available modelname values for the PSP model selection are:

<table>
<thead>
<tr>
<th>Modelname</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>psp100 or psp100e (default)</td>
<td>PSP model version 100.1, electrical</td>
</tr>
<tr>
<td>psp1000 or psp100g</td>
<td>PSP model version 100.1, geometrical</td>
</tr>
</tbody>
</table>
## Additional MOSFET Parameters

This section describes additional parameters, including parasitics, used by the equations describing MOSFET levels 1-3 and BSIM (levels 1, 2, 3, 14, 28, 47, 49, 53, and 54).

### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>acm</td>
<td>ACM</td>
<td>Source/drain area calculation method</td>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>cj</td>
<td>Cj</td>
<td>Source/drain bottom junction capacitance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>mj</td>
<td>Mj</td>
<td>Source/drain bottom junction capacitance grading coefficient</td>
<td>0.5</td>
<td>—</td>
</tr>
<tr>
<td>mjsw</td>
<td>Mjsw</td>
<td>Source/drain sidewall junction capacitance grading coefficient</td>
<td>0.33</td>
<td>—</td>
</tr>
<tr>
<td>n</td>
<td>N</td>
<td>Emission coefficient</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>cjgate</td>
<td>Cjgate</td>
<td>Zero-bias gate edge sidewall junction capacitance</td>
<td>0</td>
<td>F/m</td>
</tr>
<tr>
<td>cjsw</td>
<td>Cjsw</td>
<td>Source/drain sidewall junction capacitance</td>
<td>0</td>
<td>F/m</td>
</tr>
<tr>
<td>expli</td>
<td>EXPLI</td>
<td>Current limit</td>
<td>1.0×10¹⁵</td>
<td>A</td>
</tr>
<tr>
<td>hdif</td>
<td>Hdif</td>
<td>Length of heavily doped diffusion from contact to lightly doped region</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>js</td>
<td>Js</td>
<td>Source/drain junction reverse saturation current density</td>
<td>0.0</td>
<td>A/m²</td>
</tr>
<tr>
<td>jsw</td>
<td>Jsw</td>
<td>Source/drain sidewall junction reverse saturation current density</td>
<td>0</td>
<td>A/m</td>
</tr>
<tr>
<td>ld</td>
<td>Ld</td>
<td>Lateral diffusion into channel from source/drain diffusion</td>
<td>0.75 · xj</td>
<td>—</td>
</tr>
<tr>
<td>latd</td>
<td>Ld</td>
<td>Lateral diffusion into channel from source/drain diffusion</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>lmax</td>
<td>Lmax</td>
<td>Maximum channel length</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>lmin</td>
<td>Lmin</td>
<td>Minimum channel length</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>meto</td>
<td>Meto</td>
<td>Fringing field factor for overlap capacitance calculation</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>mj0</td>
<td>Mj</td>
<td>Source/drain bottom junction capacitance grading coefficient</td>
<td>0.5</td>
<td>—</td>
</tr>
<tr>
<td>mjw</td>
<td>Mjw</td>
<td>Source/drain sidewall junction capacitance grading coefficient</td>
<td>0.33</td>
<td>—</td>
</tr>
</tbody>
</table>

---

T-Spice 16 User Guide
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>nds</td>
<td>$N_{ds}$</td>
<td>Reverse bias slope coefficient</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>pb</td>
<td>$P_{b}$</td>
<td>Source/drain junction built-in potential</td>
<td>0.8 [level 49] 1.0 [else]</td>
<td>V</td>
</tr>
<tr>
<td>pbsw</td>
<td>$P_{bsw}$</td>
<td>Source/drain sidewall junction capacitance built-in potential</td>
<td>pb</td>
<td>V</td>
</tr>
<tr>
<td>prdt</td>
<td>$Pr_{dt}$</td>
<td>Drain resistance temperature coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>prst</td>
<td>$Pr_{st}$</td>
<td>Source resistance temperature coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
<tr>
<td>rd</td>
<td>$R_{d}$</td>
<td>Drain resistance (rsh override)</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rdc</td>
<td>$R_{dc}$</td>
<td>Drain contact resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rs</td>
<td>$R_{s}$</td>
<td>Source resistance (rsh override)</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rsc</td>
<td>$R_{sc}$</td>
<td>Source contact resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>rsh</td>
<td>$R_{sh}$</td>
<td>Source/drain sheet resistance</td>
<td>0.0</td>
<td>Ω</td>
</tr>
<tr>
<td>vnds</td>
<td>$V_{n,ds}$</td>
<td>Reverse diode current transition point</td>
<td>−1</td>
<td>V</td>
</tr>
<tr>
<td>wmax</td>
<td>$W_{max}$</td>
<td>Maximum channel width</td>
<td>1.0</td>
<td>m</td>
</tr>
<tr>
<td>wmin</td>
<td>$W_{min}$</td>
<td>Minimum channel width</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>wmlt</td>
<td>$W_{mlt}$</td>
<td>Width diffusion layer shrink reduction factor</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>xl</td>
<td>$X_{l}$</td>
<td>Difference between the drawn and the actual length</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>xlr</td>
<td>$X_{lre}$</td>
<td>Difference between physical and drawn reference channel length</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>xw</td>
<td>$X_{w}$</td>
<td>Difference between the drawn and the actual width</td>
<td>0.0</td>
<td>m</td>
</tr>
<tr>
<td>xwr</td>
<td>$X_{wre}$</td>
<td>Difference between physical and drawn reference channel width</td>
<td>0.0</td>
<td>m</td>
</tr>
</tbody>
</table>
Equations

Parasitic Resistances

MOSFET parasitics are simulated as resistances in series with the source \((R_s)\) and drain \((R_d)\). How these resistances are computed depends on the area calculation method \((\text{acm})\) specified.

<table>
<thead>
<tr>
<th>acm</th>
<th>(r_{s\text{eff}})</th>
<th>(r_{d\text{eff}})</th>
</tr>
</thead>
<tbody>
<tr>
<td>0, 10</td>
<td>(R_{s\text{eff}} = N_{rs} \cdot R_{sh} + R_{sc}) if (N_{rs} \cdot R_{sh} \neq 0) (R_{d\text{eff}} = N_{rd} \cdot R_{sh} + R_{dc}) if (N_{rd} \cdot R_{sh} \neq 0)</td>
<td>(R_{s\text{eff}} = R_s + R_{sc}) otherwise (R_{d\text{eff}} = R_s + R_{dc}) otherwise</td>
</tr>
</tbody>
</table>

\[
R_{s\text{eff}} = \frac{(L_d + L_{dif}) \cdot R_s}{W_{eff}} + N_{rs} \cdot R_{sh} + R_{sc}
\]
\[
R_{d\text{eff}} = \frac{(L_d + L_{dif}) \cdot R_d}{W_{eff}} + N_{rd} \cdot R_{sh} + R_{dc}
\]

1, 11

\[
R_{s\text{eff}} = \frac{(L_d + L_{dif}) \cdot R_s}{W_{eff}} + N_{rs} \cdot R_{sh} + R_{sc}
\]
\[
R_{d\text{eff}} = \frac{(L_d + L_{dif}) \cdot R_d}{W_{eff}} + N_{rd} \cdot R_{sh} + R_{dc}
\]

2, 3, 12, 13 If \(N_{rs}\) is specified: If \(N_{rd}\) is specified:

\[
R_{s\text{eff}} = \frac{(L_d + L_{dif}) \cdot R_s + R_{sh} \cdot H_{dif\text{eff}}}{W_{eff}}
\]
\[
R_{d\text{eff}} = \frac{(L_d + L_{dif}) \cdot R_d + R_{sh} \cdot H_{dif\text{eff}}}{W_{eff}}
\]

Parasitic Diodes

Parasitic diodes are added for each MOSFET. One diode is placed between the MOSFET's bulk and source terminals, and the other between the bulk and drain.
Diode characteristics are determined by the device parameters \( \text{as, ad, pd, ps, and geo} \), as well as the model parameters \( \text{acm, cj, cjsw, cjgate, js, jsw, is, n, nds, vnds, and hdif} \). The quantity \( \text{weff} \) also plays a role in determining default values for source and drain areas and perimeters for some values of \( \text{acm} \).

If the MOSFET bulk-source voltage \( \text{vbs} \) is positive (the bulk-source diode is forward biased), then the bulk-source DC current \( \text{ibs} \) is

\[
\text{ibs} = \text{isatbs} \cdot \exp(\frac{\text{vbs}}{n \cdot \text{vt}} - 1)
\]  
(8.404)

where \( \text{vt} = kT/q \) (the thermal voltage), and \( \text{isatbs} \) is the saturation current:

\[
\text{isatbs} = \text{js} \cdot \text{aseff} + \text{jsw} \cdot \text{pseff}
\]  
(8.405)

\( \text{aseff} \) and \( \text{pseff} \) are described below.

If this computed value of \( \text{isatbs} \) is zero, then \( \text{isatbs} \) will be set to the \( \text{is} \) parameter value.

If the MOSFET bulk-drain voltage \( \text{vds} \) is positive (the bulk-drain diode is forward biased), then the bulk-drain DC current is

\[
\text{ibd} = \text{isatbd} \cdot \exp(\frac{\text{vbd}}{n \cdot \text{vt}} - 1)
\]  
(8.406)

where \( \text{isatbd} \) is the saturation current:

\[
\text{isatbd} = \text{js} \cdot \text{adeff} + \text{jsw} \cdot \text{pdeff}
\]  
(8.407)

\( \text{adeff} \) and \( \text{pdeff} \) are described below.

If this computed value of \( \text{isatbd} \) is zero, then \( \text{isatbd} \) will be set to the \( \text{is} \) parameter value.

The exponential function in both diodes is replaced by a linear extension when the current is larger than the value of \( \text{expli} \). The linear extension is chosen such that the diode current function is continuously differentiable at the transition point where the diode current equals \( \text{expli} \).

When a MOSFET parasitic diode with saturation current \( \text{isat} \) is reverse-biased with a negative voltage \( \text{vdi} \), then its current \( \text{idi} \) behaves as follows.

If \( 0 > \text{vdi} > \text{vnds} \), then

\[
\text{idi} = \text{isat} \cdot \text{vdi}
\]  
(8.408)

If \( \text{vdi} < \text{vnds} \), then

\[
\text{idi} = \text{isat} \cdot (\text{vnds} + (\text{vdi} - \text{vnds})/\text{nds})
\]  
(8.409)

**Effective Areas and Perimeters**

Effective source and drain areas and perimeters depend on the value of parameter \( \text{acm} \) (and, if \( \text{acm} = 3 \), on parameter \( \text{geo} \)). The parameter names are:

- \( \text{aseff} \)  Effective source area
- \( \text{adeff} \)  Effective drain area
- \( \text{pseff} \)  Effective source perimeter
- \( \text{pdeff} \)  Effective drain perimeter
The values of $geo$ are:

0  Drain and source not shared by other devices (default)
1  Drain shared with another device
2  Source shared with another device
3  Drain and source shared with other devices

The area and perimeter values are computed as follows. The first table lists the equations used when $acm=0$, 1, 2, or 3. The second table lists equations for $acm=10$, 11, 12, or 13. The parameter $CALCACM$ can only be invoked when $acm=12$. The values of $defas$, $defad$, and $moscap$ are specified with the .options command.

<table>
<thead>
<tr>
<th>$acm=0$ or 10 with $as$</th>
<th>$acm=0$ without $as$</th>
<th>$acm=10$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$A_{s,eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$A_{d,eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$P_sW_{mlt}$</td>
<td>$P_{s,eff}$</td>
</tr>
<tr>
<td>$P_{d,eff}$</td>
<td>$P_dW_{mlt}$</td>
<td>$P_{d,eff}$</td>
</tr>
</tbody>
</table>

For $acm=1$ or 11:

<table>
<thead>
<tr>
<th>$acm=1$ or 11 with $as$</th>
<th>$acm=1$ without $as$</th>
<th>$acm=11$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$W_{eff}W_{mlt}$</td>
<td>$W_{eff}W_{eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$W_{eff}W_{mlt}$</td>
<td>$W_{eff}W_{eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$W_{eff}$</td>
<td>$W_{eff}$</td>
</tr>
<tr>
<td>$P_{d,eff}$</td>
<td>$W_{eff}$</td>
<td>$W_{eff}$</td>
</tr>
</tbody>
</table>

For $acm=2$ or 12:

<table>
<thead>
<tr>
<th>$acm=2$ or 12 with $as$</th>
<th>$acm=2$ without $as$</th>
<th>$acm=12$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$A_{s,eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$A_{d,eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$P_sW_{mlt}$</td>
<td>$P_{s,eff}$</td>
</tr>
</tbody>
</table>

0 Drain and source not shared by other devices (default)
1 Drain shared with another device
2 Source shared with another device
3 Drain and source shared with other devices

The area and perimeter values are computed as follows. The first table lists the equations used when $acm=0$, 1, 2, or 3. The second table lists equations for $acm=10$, 11, 12, or 13. The parameter $CALCACM$ can only be invoked when $acm=12$. The values of $defas$, $defad$, and $moscap$ are specified with the .options command.

<table>
<thead>
<tr>
<th>$acm=0$ or 10 with $as$</th>
<th>$acm=0$ without $as$</th>
<th>$acm=10$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$A_{s,eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$A_{d,eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$P_sW_{mlt}$</td>
<td>$P_{s,eff}$</td>
</tr>
<tr>
<td>$P_{d,eff}$</td>
<td>$P_dW_{mlt}$</td>
<td>$P_{d,eff}$</td>
</tr>
</tbody>
</table>

For $acm=1$ or 11:

<table>
<thead>
<tr>
<th>$acm=1$ or 11 with $as$</th>
<th>$acm=1$ without $as$</th>
<th>$acm=11$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$W_{eff}W_{mlt}$</td>
<td>$W_{eff}W_{eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$W_{eff}W_{mlt}$</td>
<td>$W_{eff}W_{eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$W_{eff}$</td>
<td>$W_{eff}$</td>
</tr>
<tr>
<td>$P_{d,eff}$</td>
<td>$W_{eff}$</td>
<td>$W_{eff}$</td>
</tr>
</tbody>
</table>

For $acm=2$ or 12:

<table>
<thead>
<tr>
<th>$acm=2$ or 12 with $as$</th>
<th>$acm=2$ without $as$</th>
<th>$acm=12$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$A_{s,eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$A_{d,eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$P_sW_{mlt}$</td>
<td>$P_{s,eff}$</td>
</tr>
</tbody>
</table>

0 Drain and source not shared by other devices (default)
1 Drain shared with another device
2 Source shared with another device
3 Drain and source shared with other devices

The area and perimeter values are computed as follows. The first table lists the equations used when $acm=0$, 1, 2, or 3. The second table lists equations for $acm=10$, 11, 12, or 13. The parameter $CALCACM$ can only be invoked when $acm=12$. The values of $defas$, $defad$, and $moscap$ are specified with the .options command.

<table>
<thead>
<tr>
<th>$acm=0$ or 10 with $as$</th>
<th>$acm=0$ without $as$</th>
<th>$acm=10$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$A_{s,eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$A_{d,eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$P_sW_{mlt}$</td>
<td>$P_{s,eff}$</td>
</tr>
<tr>
<td>$P_{d,eff}$</td>
<td>$P_dW_{mlt}$</td>
<td>$P_{d,eff}$</td>
</tr>
</tbody>
</table>

For $acm=1$ or 11:

<table>
<thead>
<tr>
<th>$acm=1$ or 11 with $as$</th>
<th>$acm=1$ without $as$</th>
<th>$acm=11$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$W_{eff}W_{mlt}$</td>
<td>$W_{eff}W_{eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$W_{eff}W_{mlt}$</td>
<td>$W_{eff}W_{eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$W_{eff}$</td>
<td>$W_{eff}$</td>
</tr>
<tr>
<td>$P_{d,eff}$</td>
<td>$W_{eff}$</td>
<td>$W_{eff}$</td>
</tr>
</tbody>
</table>

For $acm=2$ or 12:

<table>
<thead>
<tr>
<th>$acm=2$ or 12 with $as$</th>
<th>$acm=2$ without $as$</th>
<th>$acm=12$ without $as$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{s,eff}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$A_{s,eff}$</td>
</tr>
<tr>
<td>$A_{d,eff}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$A_{d,eff}$</td>
</tr>
<tr>
<td>$P_{s,eff}$</td>
<td>$P_sW_{mlt}$</td>
<td>$P_{s,eff}$</td>
</tr>
</tbody>
</table>

0 Drain and source not shared by other devices (default)
1 Drain shared with another device
2 Source shared with another device
3 Drain and source shared with other devices
### Chapter 8: Device Models
#### Additional MOSFET Parameters

For `acm=3` or `13`:

<table>
<thead>
<tr>
<th></th>
<th><code>acm=3</code> or <code>13</code></th>
<th><code>acm=3</code></th>
<th><code>acm=13</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>with <code>as</code></td>
<td>without <code>as</code></td>
<td>without <code>as</code></td>
</tr>
<tr>
<td>$P_{d_{eff}}$</td>
<td>$P_d W_{mlt}$</td>
<td>$4 \cdot H_{dif_{eff}} + 2 \cdot W_{eff}$</td>
<td>$4 \cdot H_{dif_{eff}} + 2 \cdot W_{eff}$ if <code>calcacm=1</code> otherwise</td>
</tr>
<tr>
<td>$A_{s_{eff}}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$2 \cdot H_{dif_{eff}} \cdot W_{eff}$</td>
<td>$2 \cdot H_{dif_{eff}} \cdot W_{eff}$ if <code>geo=0</code> or <code>1</code> otherwise</td>
</tr>
<tr>
<td>$A_{d_{eff}}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$2 \cdot H_{dif_{eff}} \cdot W_{eff}$</td>
<td>$2 \cdot H_{dif_{eff}} \cdot W_{eff}$ if <code>geo=0</code> or <code>1</code> otherwise</td>
</tr>
<tr>
<td>$P_{s_{eff}}$</td>
<td>$P_s W_{mlt}$</td>
<td>$4 \cdot H_{dif_{eff}} \cdot W_{eff}$</td>
<td>$2 \cdot H_{dif_{eff}} \cdot W_{eff}$ if <code>geo=0</code> or <code>1</code> otherwise</td>
</tr>
<tr>
<td>$P_{d_{eff}}$</td>
<td>$P_d W_{mlt}$</td>
<td>$4 \cdot H_{dif_{eff}} \cdot W_{eff}$</td>
<td>$2 \cdot H_{dif_{eff}} \cdot W_{eff}$ if <code>geo=0</code> or <code>1</code> otherwise</td>
</tr>
</tbody>
</table>

For `acm=2` or `12`:

<table>
<thead>
<tr>
<th></th>
<th><code>acm=2</code></th>
<th><code>acm=12</code></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>with <code>as</code></td>
<td>without <code>as</code></td>
</tr>
<tr>
<td>$P_{d_{eff}}$</td>
<td>$P_d W_{mlt}$</td>
<td>$0$ if <code>calcacm=1</code> otherwise</td>
</tr>
<tr>
<td>$A_{s_{eff}}$</td>
<td>$A_s(W_{mlt})^2$</td>
<td>$0$ if <code>calcacm=1</code> otherwise</td>
</tr>
<tr>
<td>$A_{d_{eff}}$</td>
<td>$A_d(W_{mlt})^2$</td>
<td>$0$ if <code>calcacm=1</code> otherwise</td>
</tr>
<tr>
<td>$P_{s_{eff}}$</td>
<td>$P_s W_{mlt}$</td>
<td>$0$ if <code>calcacm=1</code> otherwise</td>
</tr>
<tr>
<td>$P_{d_{eff}}$</td>
<td>$P_d W_{mlt}$</td>
<td>$0$ if <code>calcacm=1</code> otherwise</td>
</tr>
</tbody>
</table>
Resistor

Parameters

```
.model name r [parameter=X]
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Default</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>bulk</td>
<td>Name of the node used as the bulk node for capacitance.</td>
<td>Gnd</td>
<td></td>
</tr>
<tr>
<td>cap</td>
<td>Default capacitance.</td>
<td>0</td>
<td>F</td>
</tr>
<tr>
<td>capsw</td>
<td>Sidewall fringing capacitance.</td>
<td>0</td>
<td>F/m</td>
</tr>
<tr>
<td>cratio</td>
<td>Specifies how capacitance is distributed between input and output nodes. The capacitor between node1 and the bulk node has the value ( \text{cratio} \times \text{Ceff} ), while the capacitor between node2 and the bulk node has the value ( (1-\text{cratio}) \times \text{Ceff} ). Ceff is the effective capacitance as described below.</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>cox</td>
<td>Bottomwall capacitance.</td>
<td>0</td>
<td>F/m</td>
</tr>
<tr>
<td>di</td>
<td>Relative dielectric constant.</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>dlr</td>
<td>Difference between the drawn resistor length and its actual length. Multiplied by <code>.options scalm</code>. For further information, see “<code>.options</code>” (page 125).</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>dw</td>
<td>Difference between the drawn resistor width and its actual width. Multiplied by <code>.options scalm</code>. For further information, see “<code>.options</code>” (page 125).</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>l</td>
<td>Default length. Multiplied by shrink and <code>.options scalm</code> to obtain the scaled length. For further information, see “<code>.options</code>” (page 125).</td>
<td>0</td>
<td>m</td>
</tr>
<tr>
<td>level</td>
<td>Model selector—not used.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>noise</td>
<td>Default noise multiplier.</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>rac</td>
<td>Default AC resistance.</td>
<td></td>
<td>DC resistance Ohm</td>
</tr>
<tr>
<td>res</td>
<td>Default resistance.</td>
<td>0</td>
<td>Ohm</td>
</tr>
<tr>
<td>rsh</td>
<td>Sheet resistance per square.</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>shrink</td>
<td>Shrink factor.</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>tc1c</td>
<td>First-order temperature coefficient for capacitance.</td>
<td>0</td>
<td>( \text{deg}^{-1} )</td>
</tr>
<tr>
<td>tc2c</td>
<td>Second-order temperature coefficient for capacitance.</td>
<td>0</td>
<td>( \text{deg}^{-2} )</td>
</tr>
<tr>
<td>tc1r</td>
<td>First-order temperature coefficient for resistance.</td>
<td>0</td>
<td>( \text{deg}^{-1} )</td>
</tr>
</tbody>
</table>
### Large-Signal Model

![Resistor Diagram](image)

### Equations

#### Resistance

Effective length is calculated as

$$L_{\text{eff}} = L_{\text{scaled}} - (2 \cdot dlr \cdot scalm) \tag{8.410}$$

Effective width is calculated as

$$W_{\text{eff}} = W_{\text{scaled}} - (2 \cdot dw \cdot scalm) \tag{8.411}$$

If element resistance is specified, the effective resistance is

$$R_{\text{eff}} = \text{resistance} \times devscale \tag{8.412}$$
Otherwise, if \( W_{\text{eff}} \cdot L_{\text{eff}} \cdot rsh = 0 \), then \( R_{\text{eff}} = \text{res} \cdot \text{devscale} \).

Or if \( W_{\text{eff}} \cdot L_{\text{eff}} \cdot rsh \neq 0 \), then \( R_{\text{eff}} = L_{\text{eff}} \cdot rsh \cdot (\text{devscale}) / W_{\text{eff}} \).

In AC analysis, the device’s resistance is

\[
R_{\text{AC eff}} = \text{acres} \cdot \text{devscale}
\]  

(8.413)

if \text{acres} is given.

\[
R_{\text{AC eff}} = \text{rac} \cdot \text{devscale}
\]  

(8.414)

if \text{acres} is not given and model parameter \text{rac} is given.

\[
R_{\text{AC eff}} = R_{\text{eff}}
\]  

(8.415)

if neither \text{acres} nor \text{rac} is given.

\[\text{Note:}\]
If T-Spice calculates the effective resistance (\( R_{\text{eff}} \) or \( R_{\text{AC eff}} \)) to be less than \( 10^{-5} \ \Omega \), then a warning message is issued and the effective resistance is automatically assigned a value of \( 10^{-5} \ \Omega \).

\[\text{For additional information on \text{devscale}, see the device statement “Resistor (r)” on page 198.}\]

\section*{Capacitance}

Effective length is calculated as

\[
L_{\text{eff}} = L_{\text{scaled}} - (2 \cdot dlr \cdot scalm)
\]  

(8.416)

Effective width is calculated as

\[
W_{\text{eff}} = W_{\text{scaled}} - (2 \cdot dw \cdot scalm)
\]  

(8.417)

If element capacitance is specified, the effective capacitance is

\[
C_{\text{eff}} = c \times \text{devscale}
\]  

(8.418)

If \text{cap} is the only model parameter specified

\[
C_{\text{eff}} = \text{cap} \cdot \text{devscale}
\]  

(8.419)

Otherwise,

\[
C_{\text{eff}} = \text{devscale} \cdot (L_{\text{eff}} \cdot W_{\text{eff}} \cdot cox + 2(L_{\text{eff}} + W_{\text{eff}}) \times \text{capsw})
\]  

(8.420)
If $\text{cox}$ is not specified, it is computed in one of two ways.

If $\text{di}$ is specified: $\text{cox} = \frac{8.8542149 \times 10^{-12}}{\text{thick}}$ (8.421)

If $\text{di}$ is not specified: $\text{cox} = \frac{3.453148 \times 10^{-11}}{\text{thick}}$ (8.422)

**Note:** If $c$, $\text{cap}$, $\text{cox}$, and $\text{thick}$ are not specified, no capacitor is created.
Switch

A current- or voltage-controlled switch.

Syntax

```
.model modelname sw|csw [parameter=value [parameter=value [...]]]
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Model</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>vt</td>
<td>sw</td>
<td>Threshold voltage</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>it</td>
<td>csw</td>
<td>Threshold current</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>vh</td>
<td>sw</td>
<td>Hysteresis voltage</td>
<td>0</td>
<td>V</td>
</tr>
<tr>
<td>ih</td>
<td>csw</td>
<td>Hysteresis current</td>
<td>0</td>
<td>A</td>
</tr>
<tr>
<td>ron</td>
<td>sw, csw</td>
<td>ON resistance</td>
<td>1</td>
<td>Ohm</td>
</tr>
<tr>
<td>roff</td>
<td>sw, csw</td>
<td>OFF resistance</td>
<td>1e12</td>
<td>Ohm</td>
</tr>
<tr>
<td>dv</td>
<td>sw</td>
<td>Threshold transition width</td>
<td>0.01</td>
<td>V</td>
</tr>
<tr>
<td>di</td>
<td>csw</td>
<td>Threshold transition width</td>
<td>1e-8</td>
<td>A</td>
</tr>
<tr>
<td>hdt</td>
<td>sw, csw</td>
<td>Effective only when hysteresis is used. The time to switch 99.9% of the way between vt and vt-vh.</td>
<td>1e-10</td>
<td>s</td>
</tr>
</tbody>
</table>

The T-Spice switch is essentially a controlled resistor. The resistance is `ron` when the switch is on, and `roff` when the switch is off. The switch changes state between on and off when the controlling voltage or current is at its threshold value.

The following figure shows the resistance of the switch as a function of the controlling variable:

![Resistance vs Control Voltage/Current](image)

The T-Spice switch elements can display hysteresis, so that the threshold value is different when the control voltage/current is increasing than when it is decreasing. For a voltage-controlled switch, the threshold voltage is `vt` when `v(control1, control2)` is increasing, and `vt-vh` when `v(control1, control2)` is decreasing. For a current-controlled switch, the threshold current is `it` when `i(source_name)` is...
increasing, and \texttt{it\-ih} when \texttt{i(source\_name)} is decreasing. The switch is on when the control voltage or current is greater than the threshold value.

The \texttt{dv} and \texttt{di} parameters define a small interval around the threshold in which a smooth transition between \texttt{ron} and \texttt{roff} is made.

\section*{Examples}

The following example creates a voltage-controlled switch:

\lstinputlisting{example1.txt}

The following example creates a current-controlled switch:

\lstinputlisting{example2.txt}
Transmission Line

T-Spice supports two transmission line models:

- **Lossless** line, defined by characteristic impedance and delay. This model is described by Branin 1967.
- **Lossy** line, defined by RLCG parameters. This model is described here.

**Equations**

The lossy transmission line is modeled with equivalent circuits consisting of cascaded cells or *lumps*, typically comprising discrete resistors, inductors, and capacitors. Lumps essentially discretize the transmission line wave equations over the length of the line. Distributed RLCG values are converted to non-distributed (lumped element) values:

\[
R_{lump} = R \cdot \frac{l}{n} \tag{8.423}
\]

\[
L_{lump} = L \cdot \frac{l}{n} \tag{8.424}
\]

\[
C_{lump} = C \cdot \frac{l}{n} \tag{8.425}
\]

\[
G_{lump} = G \cdot \frac{l}{n} \tag{8.426}
\]

where \( l \) is the physical length of the transmission line and \( n \) is the number of lumps.

Typically, many lumps are needed to model a transmission line accurately. The number of lumps is specified by the *lumps* parameter on the device statement. A cascade ladder network—the *iterative ladder circuit*—is constructed with the specified number of lumps.

The “gamma” lump type is the most common implementation.
The “tee” lump type is a symmetrical alternative that includes an additional series resistance and inductance per lump.

The “pi” lump type is formed by adding identical shunt elements at the input and output of each lump.

Low-loss transmission lines need not be modeled with RLCG lumps. An alternative approach is to use lossless transmission line sections separated by lumped resistances and conductances. The “hybrid RGT” lump type consists of an ideal transmission line section with series resistances and shunt conductances at input and output. This equivalent circuit defaults to the lossless transmission line when \( R \) and \( G \) values are zero, and is therefore the default \texttt{lumptype} option.
9 Small-Signal and Noise Models

Introduction

Small-Signal Models

The linear small-signal models for diodes, BJTs, JFETs, MESFETs, and MOSFETs described in this chapter are derived from T-Spice’s nonlinear device representations.

Instead of using simplified equations to compute the small-signal model parameters, which can introduce errors at low and high frequencies, T-Spice generates linearized small-signal models directly from the device equations, with the most accurate small-signal results available.

Small-signal parameters are evaluated at the DC operating point. The currents and voltages appearing in the equations in this chapter are all DC values. In addition, the voltages are measured at the “intrinsic” terminals, inside the parasitic resistances connected to the external terminals.

Small-signal data are available with the ".acmodel" (page 79) command.

Noise Models

The following parameters are used by many of T-Spice’s noise models.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Description</th>
<th>Default</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>noiselevel</td>
<td>nlev</td>
<td>Noise equation selector (JFET, MESFET, MOSFET only)</td>
<td>2</td>
<td>—</td>
</tr>
<tr>
<td>gdsnoise</td>
<td>gdsnoi</td>
<td>Channel noise coefficient (JFET, MESFET, MOSFET only)</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>af</td>
<td>Af</td>
<td>Flicker noise exponent</td>
<td>1.0</td>
<td>—</td>
</tr>
<tr>
<td>kf</td>
<td>Kf</td>
<td>Flicker noise coefficient</td>
<td>0.0</td>
<td>—</td>
</tr>
</tbody>
</table>

Noise models add RMS noise current sources to small signal models in order to simulate noise power densities (NPDs) in the device. The NPDs are calculated from model parameter values and DC operating point conditions.

For example, each of the resistors in a circuit, including the parasitic resistances in models, generates thermal noise, whose NPD is inversely proportional to the resistance:

$$\left[\frac{I_{th}}{R}\right] = \frac{4kT}{R} \cdot \Delta f$$

(9.1)
where \( k \) is Boltzmann’s constant, \( T \) is the temperature in Kelvins, \( R \) is the resistance, and \( \Delta f \) is the width of the frequency band over which the noise is measured (1 Hz in T-Spice). The corresponding noise current source has a magnitude of \( \sqrt{|I_{noise}|^2} \) and units of \( A/\sqrt{Hz} \), and is placed in parallel with the resistor.

The following sections show the NPD calculation for intrinsic noise sources in the models. These sources represent shot and/or flicker noise.

*Shot* noise is typically computed as

\[
|I_{shot}|^2 = 2q|I_x| \cdot \Delta f
\]

where \( q \) is the electron charge and \( I_x \) is the DC current into terminal \( x \) at the operating point of the device.

*Flicker* noise is usually modeled by

\[
|I_{flicker}|^2 = \frac{K|I_x|^4}{f} \cdot \Delta f
\]

where \( f \) is the center frequency of the band over which the noise is being measured.

Where appropriate, alternate formulations for shot and flicker noise are presented for individual models.

The noise source corresponding to the NPD, \( \sqrt{|I_{XY}|^2} \), is connected between the intrinsic \( X \) and \( Y \) nodes in the noise model.
Diode

Small-Signal Model

The conductance $g_D$ is the partial derivative of the forward current $I_D$ with respect to the voltage $V_D$ across the intrinsic diode.

$$g_D = \frac{\partial I_D}{\partial V_D}_{op}$$

(9.4)

The small-signal capacitance $C_D$ across the diode is

$$C_D = \frac{\partial Q_D}{\partial V_D}_{op}$$

(9.5)

$r_D$ models the diode’s linear parasitic resistance.

Noise Model

In addition to the thermal noise associated with $r_D$, this model includes a PN junction noise source that includes shot and flicker noise:

$$\left(\frac{1}{V_{AB,i}}\right)^2 = 2qI_A \cdot \Delta f + \frac{K_I I_A^4}{f} \cdot \Delta f$$

(9.6)
BJT Level 1 (Gummel-Poon)

Small-Signal Model

The bipolar conductances are denoted by $g_m$, $g_o$, $g_\pi$, and $g_\mu$.

$$g_m = \frac{\partial I_C}{\partial V_{BE}}_{op}$$  \hspace{1cm} (9.7)

$$g_o = \frac{1}{r_o} = \frac{\partial I_C}{\partial V_{CE}}_{op}$$  \hspace{1cm} (9.8)

$$g_\pi = \frac{1}{r_\pi} = \frac{\partial I_B}{\partial V_{BE}}_{op}$$  \hspace{1cm} (9.9)

$$g_\mu = \frac{1}{r_\mu} = \frac{\partial I_B}{\partial V_{BC}}_{op}$$  \hspace{1cm} (9.10)

The capacitances $C_E$ and $C_C$ are denoted by $C_\pi$ and $C_\mu$ respectively.

$$C_\pi = \frac{\partial Q_B}{\partial V_{BE}}_{op}$$  \hspace{1cm} (9.11)

$$C_\mu = \frac{\partial Q_B}{\partial V_{BC}}_{op}$$  \hspace{1cm} (9.12)
Other parameters computed are the transistor gain at DC operating point $\beta_{DC}$, the AC signal gain $\beta_{AC}$, and the gain-band-width product $f_T$.

\[
\beta_{DC} = \left. \frac{I_C}{I_B} \right|_{\text{op}} \tag{9.13}
\]

\[
\beta_{AC} = \left. \frac{\partial I_C}{\partial I_B} \right|_{\text{op}} = g_m r_\pi \tag{9.14}
\]

\[
f_T = \left. \frac{g_m}{2\pi(C_\pi + C_\mu)} \right|_{\text{op}} \tag{9.15}
\]

**Gummel-Poon Noise Model**

T-Spice uses the equations in this section to simulate thermal, shot, and flicker noise power densities in the BJT Gummel-Poon device model.

**Thermal Noise**

T-Spice uses the following equations to calculate thermal noise power density ($nd_{therm}$) of the base resistor ($r_b$), collector resistor ($r_c$), and emitter resistor ($r_e$) noise currents, respectively:

\[
nd_{therm}(I_b) = \frac{4kT}{r_{bb}} \tag{9.16}
\]

\[
nd_{therm}(I_c) = \frac{4kT}{r_{ceff}} \tag{9.16}
\]

\[
nd_{therm}(I_e) = \frac{4kT}{r_{eeff}} \tag{9.16}
\]

where $k$ is Boltzmann’s constant, $T$ is temperature, and $r_{bb}$, $r_{ceff}$, and $r_{eeff}$ are the effective base, collector, and emitter resistances.

**Shot Noise**

The following equations give the shot noise density ($nd_{shot}$) of the base current ($I_b$) and collector current ($I_c$), respectively:

\[
nd_{shot}(I_b) = 2qI_b \tag{9.17}
\]

\[
nd_{shot}(I_c) = 2qI_c \tag{9.17}
\]

where $q$ is the elementary electron charge.

**Flicker Noise**

The flicker noise density ($nd_{flick}$) of the base current ($I_b$) is:

\[
nd_{flick}(I_b) = \frac{KF \cdot (I_b)^{AF}}{f}, \tag{9.18}
\]
where $f$ represents frequency.
JFET/MESFET

Small-Signal Model

The transconductance is denoted by $g_m$ and the output conductance by $g_{DS}$.

\begin{align}
  g_m &= \left. \frac{\partial I_D}{\partial V_{GS}} \right|_{op} \\
  g_{DS} &= \left. \frac{\partial I_D}{\partial V_{DS}} \right|_{op}
\end{align}

(9.19) \hspace{1cm} (9.20)

The gate junction conductances are denoted by $g_{GS}$ and $g_{GD}$. These are usually very small, since the junctions are reverse-biased in normal operation.

\begin{align}
  g_{GS} &= \left. \frac{\partial I_G}{\partial V_{GS}} \right|_{op} \\
  g_{GD} &= \left. \frac{\partial I_G}{\partial V_{GD}} \right|_{op}
\end{align}

(9.21) \hspace{1cm} (9.22)

The gate junction capacitances are denoted by $C_{GS}$ and $C_{GD}$.

\begin{align}
  C_{GS} &= \left. \frac{\partial Q_G}{\partial V_{GS}} \right|_{op} \\
  C_{GD} &= \left. \frac{\partial Q_G}{\partial V_{GD}} \right|_{op}
\end{align}

(9.23) \hspace{1cm} (9.24)

$r_D$ and $r_S$ are constant linear resistances.
Noise Model

In addition to the thermal noise associated with $r_D$, $r_S$, and $r_G$, this model includes the intrinsic noise source

$$[i_{DS}]^2 = [i_{ch}]^2 + \frac{K_r I_D}{f} \cdot \Delta f$$

(9.25)

where $I_D$ is the DC operating point drain current.

Channel noise is modeled in two ways, depending on the parameter noiselevel. When noiselevel < 3,

$$[i_{ch}]^2 = \frac{8}{3} k T g_m \cdot \Delta f$$

(9.26)

When noiselevel = 3,

$$[i_{ch}]^2 = \frac{8}{3} k T \beta \cdot \left( V_{GS} - V_{th} \right) \cdot G_{DS} \cdot \left( \frac{1 + \alpha + \alpha^2}{1 + \alpha} \right) \cdot \Delta f$$

(9.27)

where $k$ is Boltzmann’s constant, $T$ is the current temperature in Kelvins, $g_m$ is the transconductance, $\beta$ is the gain at the operating point, and

$$\alpha = \begin{cases} 
1 - \frac{V_{DS}}{V_{GS} - V_{th}} & \text{(linear region)} \\
0 & \text{(saturation region)}
\end{cases}$$

(9.28)
The MOSFET conductances are denoted by \( g_m \), \( g_{DS} \), and \( g_{mbs} \).

\[
g_m = \left. \frac{\partial I_D}{\partial V_{GS}} \right|_{op} \tag{9.29}
\]

\[
g_{DS} = \left. \frac{\partial I_D}{\partial V_{DS}} \right|_{op} \tag{9.30}
\]

\[
g_{mbs} = \left. \frac{\partial I_D}{\partial V_{BS}} \right|_{op} \tag{9.31}
\]

The gate junction conductances are denoted by \( g_{BD} \) and \( g_{BS} \). These are usually very small, since the junctions are reverse-biased in normal operation.

\[
g_{BD} = \left. \frac{\partial I_B}{\partial V_{BD}} \right|_{op} \tag{9.32}
\]

\[
g_{BS} = \left. \frac{\partial I_B}{\partial V_{BS}} \right|_{op} \tag{9.33}
\]
The bulk junction capacitances are denoted by $CBS$ and $CBD$.

\[ C_{BD} = \left. \frac{\partial Q_B}{\partial V_{BD}} \right|_{\text{op}} \]  
(9.34)

\[ C_{BS} = \left. \frac{\partial Q_B}{\partial V_{BS}} \right|_{\text{op}} \]  
(9.35)

The gate-to-junction and gate-to-bulk capacitances are denoted by $CGS$, $CGD$, and $CGB$.

\[ C_{GS} = \left. \frac{\partial Q_G}{\partial V_{GS}} \right|_{\text{op}} \]  
(9.36)

\[ C_{GD} = \left. \frac{\partial Q_G}{\partial V_{GD}} \right|_{\text{op}} \]  
(9.37)

\[ C_{GB} = \left. \frac{\partial Q_G}{\partial V_{GB}} \right|_{\text{op}} \]  
(9.38)

The AC small-signal capacitances are denoted by $CD_{tot}$, $CG_{tot}$, and $CS_{tot}$, and $CB_{tot}$.

\[ CD_{tot} = \left. \frac{\partial Q_D}{\partial V_{D}} \right|_{\text{tot}} \]  
(9.39)

\[ CG_{tot} = \left. \frac{\partial Q_G}{\partial V_{G}} \right|_{\text{tot}} \]  
(9.40)

\[ CS_{tot} = \left. \frac{\partial Q_S}{\partial V_{S}} \right|_{\text{tot}} \]  
(9.41)

\[ CB_{tot} = \left. \frac{\partial Q_B}{\partial V_{B}} \right|_{\text{tot}} \]  
(9.42)

Since these parameters are computed directly from the MOSFET equations, the model is valid for low- and high-frequency simulations.

**Noise Model**

In addition to the thermal noise associated with $rD$ and $rS$, this model includes channel and flicker noise sources.

\[ [I_{DS3}]^2 = [I_{ch}]^2 + [I_{flicker}]^2 \]  
(9.43)
Channel noise is modeled in two ways, depending on the parameter \textit{noiselevel}. When \textit{noiselevel} $< 3$,

$$\left| V_{ch} \right|^2 = \frac{8}{3} kT g_m \cdot \Delta f$$ \hfill (9.44)

When \textit{noiselevel} = 3,

$$\left| V_{ch} \right|^2 = \frac{8}{3} kT \beta \cdot (V_{GS} - V_{th}) \cdot G_{DS} \cdot \left( \frac{1 + \alpha + \alpha^2}{1 + \alpha} \right) \cdot \Delta f$$ \hfill (9.45)

where $\beta$ is the gain at the operating point and

$$\alpha = \begin{cases} 1 - \frac{V_{DS}}{V_{Dsat}} & \text{(linear region)} \\ 0 & \text{(saturation region)} \end{cases}$$ \hfill (9.46)

Flicker noise is modeled in three ways, depending on \textit{noiselevel}. When \textit{noiselevel} = 0,

$$\left| i_{flicker} \right|^2 = \frac{K_f \cdot (i_{DS})^{A_f}}{C_{ox} L_{eff}^2} \cdot \Delta f$$ \hfill (9.47)

When \textit{noiselevel} = 1,

$$\left| i_{flicker} \right|^2 = \frac{K_f \cdot (i_{DS})^{A_f}}{f c_{ox} L_{eff} W_{eff}} \cdot \Delta f$$ \hfill (9.48)

When \textit{noiselevel} = 2 or \textit{noiselevel} = 3,

$$\left| i_{flicker} \right|^2 = \frac{K_f \cdot g_m^2}{f^3 c_{ox} L_{eff} W_{eff}} \cdot \Delta f$$ \hfill (9.49)
References


Introduction

T-Spice is often required to study the effects of variations in parameter values on circuit performance. For example, parametric analysis can be used to evaluate multidimensional trends in the output over defined ranges of input values, or the sensitivity of circuit behavior to random fluctuations in fabrication conditions.

A large range of parameters may be systematically and automatically varied:

- External parameters (such as temperature)
- Simulation parameters (such as tolerances)
- Device parameters (such as input voltage level or transistor length)
- Model parameters (such as transistor threshold voltage)

Three types of parametric analysis are supported by T-Spice: parameter sweeping, Monte Carlo analysis, and optimization.

This chapter guides you through several tutorial problems in order to demonstrate some basic concepts of parametric analysis.

The example files for these tutorials can be found in the \examples\input subdirectory of the T-Spice Pro installation path.

Output File Formats

Results produced by parameter sweeps and Monte Carlo analysis are easily read in T-Spice output files. You can open a T-Spice output file from the T-Spice Simulation Manager by clicking on the Show Output button. Alternatively, select File > Open and browse within the Open dialog to select the output filename. T-Spice output files are text files; they are also readable in the text editor of your choice.

Analysis results for parameter sweeps are reported in table format, with section headings that contain the current values of the swept parameters. For example, if a .step command invokes several transient analyses, each analysis produces its own output section, the header of which shows the parameter values for that analysis (e.g., TRANSIENT ANALYSIS – vdd=3).

Results obtained from .measure commands are listed at the end of each output section corresponding to a specific parameter value. These measurements are summarized at the end of the sweep in a table labeled TRANSFER ANALYSIS. The use of a .step command causes .measure results to be plotted in W-Edit, with the swept variable as the x-axis.
Chapter 10: Parametric Analysis

Parameter Sweeps

In a *parameter sweep*, a specified parameter is held or initialized at a given value, all analyses requested by the input file are performed, and the results are recorded. Then the parameter is incremented by a set amount, and the same analyses are repeated. The cycle continues as the parameter is incremented through a defined range of values.

Parameter values may be swept *linearly*—in identical increments, typically through a limited range—or *logarithmically*—in exponential increments, typically through a range spanning multiple orders of magnitude. You can also specify a sweep over a list of values.

Parameter sweeping is performed by using the **sweep** option with one of the following commands:

- `.ac` (see “.ac” on page 76)
- `.dc` (see “.dc” on page 89)
- `.step` (see “.step” on page 158)
- `.tran` (see “.tran” on page 165)

Additionally, it is also possible to perform simultaneous parameter sweeps of several different variables using the command **.data** (see “.connect” on page 86).

Adding the **sweep** option to a `.tran` or `.ac` command instructs T-Spice to perform that analysis for all parameter values of the specified sweep. Using the **sweep** option with an analysis command is similar to using it with **.step**. A single **.step** command causes T-Spice to perform parameter sweeps for *all* analysis commands in the input file. If **sweep** is specified on an analysis command and a **.step** command is present, the **sweep** specified with the analysis command is nested inside the sweep specified with the **.step** command.

All input files listed for this chapter are in the directory `<install_dir>\tutorial\input`.

**Example 1: Parametric Sweep**

This example uses a ring oscillator to demonstrate the basic features of a parametric sweep.

*T-Spice Input*  
ring2.cir

*Output*  
ring2.sp

**T-Spice Input**

* Circuit: ring2

* .SUBCKT inv in out Gnd Vdd
c2 out Gnd cap
mlp out in Vdd Vdd pmos L=5u W=12u
mn1 out in Gnd Gnd nmos L=5u W=8u
.ENDS

* Main circuit: ring2
cinv1 a7 Gnd 400ff
.include ml2_125.md
Xinv1 a1 a2 Gnd Vdd inv
Xinv2 a2 a3 Gnd Vdd inv
In this example, instead of keeping the load capacitor in the inverter subcircuit constant, the capacitor is defined as a variable `cap` and T-Spice sweeps it over a range of 200fF to 1000fF. The `.param` statement sets a nominal value for `cap` and the `.step` command sweeps `cap` linearly from 200fF to 1000fF in increments of 200fF.

The `.measure` statement measures the period, time delay, and pulse width of the ring oscillator at the different values of capacitance. In this example, period is measured at \( v(a2) = 3V \) from its second falling edge to its third falling edge. Time delay is measured from the second falling edge of \( v(a2) \) at 3V to the second falling edge of \( v(a1) \) at \( 3V \). Pulse width is measured from the second rising edge of \( v(a2) \) at 1.5V to the second falling edge of \( v(a2) \) at 1.5V.

**Output**

T-Spice reports the transient analysis results in five sections for `cap=200fF`, `400fF`, `600fF`, `800fF`, and `1000fF`.

Following is part of the output section for `cap=200fF`.

**TRANSIENT ANALYSIS - cap=2e-013**

<table>
<thead>
<tr>
<th>Time &lt;s&gt;</th>
<th>v(a1) &lt;V&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0000e+000</td>
<td>0.0000e+000</td>
</tr>
<tr>
<td>1.0876e-010</td>
<td>1.9456e-002</td>
</tr>
<tr>
<td>4.9424e-010</td>
<td>3.3521e-001</td>
</tr>
<tr>
<td>8.9842e-010</td>
<td>6.0502e-001</td>
</tr>
<tr>
<td>1.2047e-009</td>
<td>7.0183e-001</td>
</tr>
</tbody>
</table>

The measurement results are reported at the end of each section.

**MEASUREMENT RESULTS - cap=2e-013**

- **period = 9.9026e-008**
  - Trigger = 1.7549e-007
  - Target = 2.7452e-007

- **pulsewidth = 4.8288e-008**
  - Trigger = 1.3217e-007
  - Target = 1.8046e-007

- **timedelay = 4.1316e-008**
  - Trigger = 1.7549e-007
  - Target = 2.1681e-007
Measurement results are summarized in a table at the end of the .measure output file in the simulation results folder. Because the .step command was used, measurement results will be plotted against the parameter cap in W-Edit.

**TRANSFER ANALYSIS**

<table>
<thead>
<tr>
<th>cap</th>
<th>period</th>
<th>pulsewidth</th>
<th>timedelay</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0000e-013</td>
<td>9.9026e-008</td>
<td>4.8288e-008</td>
<td>4.1316e-008</td>
</tr>
<tr>
<td>4.0000e-013</td>
<td>1.4248e-007</td>
<td>6.9783e-008</td>
<td>5.9662e-008</td>
</tr>
<tr>
<td>6.0000e-013</td>
<td>1.8579e-007</td>
<td>9.1213e-008</td>
<td>7.8093e-008</td>
</tr>
<tr>
<td>8.0000e-013</td>
<td>2.2913e-007</td>
<td>1.1262e-007</td>
<td>9.6608e-008</td>
</tr>
<tr>
<td>1.0000e-012</td>
<td>2.7235e-007</td>
<td>1.3401e-007</td>
<td>1.1505e-007</td>
</tr>
</tbody>
</table>

**Waveform**

The following two figures show the waveform output for this example. The first displays the output of the ring oscillator vs. time with different capacitance values; the second displays the period, time delay and pulse width in a single chart.

Ring oscillator versus time at different capacitances:
Monte Carlo Analysis

In a Monte Carlo analysis, T-Spice performs simulation runs using randomly chosen parameter values. The parameter values for each run are chosen from probability distributions defined by the user.

T-Spice’s Monte Carlo analysis will generate a summary output of all simulation run measurement results after all runs are completed. T-Spice will also report, for each Monte Carlo iteration, the values of all expressions evaluated using probability distributions.

Monte Carlo analysis can be invoked using the keywords `sweep monte` with one of the following commands:

- `.ac` (page 76)
- `.dc` (page 89)
- `.step` (page 158)
- `.tran` (page 165)

Syntax and options for the keyword `sweep` are defined in the section describing `.step` (page 158).

Probability distributions are assigned to parameters by use of the `.param` command. For a complete description of the syntax of this command, see `.param` (page 131).

Example 2: Monte Carlo Analysis

This example demonstrates Monte Carlo Analysis on a CMOS inverter circuit.

Input

invert5.cir
A Monte Carlo analysis sweeps parameter values that are chosen based on statistical variations. In this example, T-Spice varies the model parameter \( vto \) using random values chosen by probability distribution. In model file `ml2_125mc.md`, the `.model` statement specifies the \( vto \) parameter as two variables: \( vto_n \) for an n-channel MOSFET and \( vto_p \) for a p-channel MOSFET.

The `.param` statement defines the probability distribution, where \( vto_n=\text{unif}(0.622490, 0.5, 1) \) and \( vto_p=\text{unif}(-0.63025, 0.5, 1) \) select uniform distributions centered at \( 0.622490 \) and \( -0.63025 \) with relative variation of \( 50\% \). The keyword `monte=10` in the `.tran` statement invokes Monte Carlo analysis with 10 runs. The `.measure` statement measures the falltime of the output pulse for different values of \( vto \).

Output

T-Spice reports the transient analysis results in ten sections for the ten Monte Carlo runs.

Following is part of the output section for the first run.

```
TRANSIENT ANALYSIS - Monte-Carlo-index=1
Time<s>       v(in)<V>       v(out)<V>
0.0000e+000  0.0000e+000  2.9996e+000
6.0000e-010  0.0000e+000  2.9996e+000
2.6000e-009  0.0000e+000  2.9996e+000
4.6000e-009  0.0000e+000  2.9996e+000
6.6000e-009  0.0000e+000  2.9996e+000
8.5999e-009  0.0000e+000  2.9996e+000
...`
```

Measurement results are reported at the end of each section.

```
MEASUREMENT RESULTS - Monte-Carlo-index=1
falltime = 1.2278e-008
Trigger = 1.0471e-007
Target = 1.1699e-007
```

At the end of the output file, T-Spice reports the Monte Carlo parameter values for each run.
MONTE CARLO PARAMETER VALUES

Index 1

parameter Vto for model nmos =  3.1202e-001
parameter Vto for model pmos =  -6.7032e-001

Index 2

parameter Vto for model nmos =  4.3157e-001
parameter Vto for model pmos =  -8.2483e-001

Index 3

parameter Vto for model nmos =  6.7541e-001
parameter Vto for model pmos =  -6.1756e-001

... Measurement results are summarized along with the following statistical results from the analysis — minimum, maximum, mean, average deviation, variance, and sigma:

MONTE CARLO PARAMETER STATISTICS

<table>
<thead>
<tr>
<th>Variable</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Median</th>
<th>Mean</th>
<th>AvgDev</th>
<th>Variance</th>
<th>Sigma</th>
</tr>
</thead>
<tbody>
<tr>
<td>parameter vth0 for model NMOS</td>
<td>325.8096m</td>
<td>382.7094m</td>
<td>368.9319m</td>
<td>365.2907m</td>
<td>14.6962m</td>
<td>278.8930u</td>
<td>16.7000m</td>
</tr>
<tr>
<td>parameter xl for model NMOS</td>
<td>-22.8550n</td>
<td>22.5583n</td>
<td>7.4712n</td>
<td>4.2441n</td>
<td>10.7063n</td>
<td>149.6252a</td>
<td>12.2321n</td>
</tr>
<tr>
<td>parameter xw for model NMOS</td>
<td>-79.6264n</td>
<td>-12.5077n</td>
<td>-46.2837n</td>
<td>-45.0382n</td>
<td>21.0046n</td>
<td>503.2271a</td>
<td>22.4327n</td>
</tr>
<tr>
<td>parameter vth0 for model PMOS</td>
<td>-513.5953m</td>
<td>-460.3336m</td>
<td>-493.8904m</td>
<td>-493.1504m</td>
<td>12.4712m</td>
<td>211.9433u</td>
<td>14.5582m</td>
</tr>
<tr>
<td>parameter xw for model PMOS</td>
<td>-66.7281n</td>
<td>-4.0675n</td>
<td>-31.4267n</td>
<td>-38.3830n</td>
<td>21.2560n</td>
<td>424.9114a</td>
<td>20.6133n</td>
</tr>
</tbody>
</table>

*WEDIT: XFER cycles Monte-Carlo-index 10
*WEDIT: .step Monte-Carlo-index 1         10 1

TRANSFER ANALYSIS

<table>
<thead>
<tr>
<th>Index&lt;&gt;</th>
<th>AvgDelay&lt;&gt;</th>
<th>RiseTime&lt;&gt;</th>
<th>FallTime&lt;&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000e+000</td>
<td>9.4912e-010</td>
<td>2.0142e-009</td>
<td>1.5805e-009</td>
</tr>
<tr>
<td>2.0000e+000</td>
<td>9.0216e-010</td>
<td>1.8379e-009</td>
<td>1.5957e-009</td>
</tr>
<tr>
<td>3.0000e+000</td>
<td>9.1165e-010</td>
<td>1.9438e-009</td>
<td>1.5161e-009</td>
</tr>
<tr>
<td>4.0000e+000</td>
<td>9.4299e-010</td>
<td>2.0042e-009</td>
<td>1.5635e-009</td>
</tr>
<tr>
<td>5.0000e+000</td>
<td>9.2365e-010</td>
<td>1.9451e-009</td>
<td>1.5745e-009</td>
</tr>
<tr>
<td>6.0000e+000</td>
<td>9.0999e-010</td>
<td>1.8812e-009</td>
<td>1.5810e-009</td>
</tr>
<tr>
<td>7.0000e+000</td>
<td>9.3615e-010</td>
<td>1.9600e-009</td>
<td>1.5934e-009</td>
</tr>
<tr>
<td>8.0000e+000</td>
<td>9.1777e-010</td>
<td>1.9021e-009</td>
<td>1.5770e-009</td>
</tr>
<tr>
<td>9.0000e+000</td>
<td>9.2884e-010</td>
<td>1.9601e-009</td>
<td>1.5707e-009</td>
</tr>
<tr>
<td>1.0000e+001</td>
<td>9.3430e-010</td>
<td>2.0159e-009</td>
<td>1.5245e-009</td>
</tr>
<tr>
<td>Minimum</td>
<td>9.0216e-010</td>
<td>1.8379e-009</td>
<td>1.5161e-009</td>
</tr>
<tr>
<td>Maximum</td>
<td>9.4912e-010</td>
<td>2.0159e-009</td>
<td>1.5957e-009</td>
</tr>
<tr>
<td>Mean</td>
<td>9.2566e-010</td>
<td>1.9464e-009</td>
<td>1.5767e-009</td>
</tr>
<tr>
<td>Avgdev</td>
<td>1.2617e-011</td>
<td>4.4437e-011</td>
<td>1.9792e-011</td>
</tr>
<tr>
<td>Variance</td>
<td>2.3398e-022</td>
<td>3.4592e-021</td>
<td>7.1960e-022</td>
</tr>
<tr>
<td>Sigma</td>
<td>1.5296e-011</td>
<td>5.8815e-011</td>
<td>2.6825e-011</td>
</tr>
</tbody>
</table>
Waveform

Concurrent with the analysis, W-Edit launches and displays the families of output traces from a Monte Carlo run.

Optimization

The T-Spice optimization feature allows circuit parameters to be tuned within given ranges to achieve the best possible circuit performance. T-Spice seeks to minimize the difference between performance measurements and a user-defined optimization goal. In order to specify an optimization, you must supply the following:

[1] A list of parameters which can be adjusted to optimize performance. Each parameter is assigned a nominal, minimum, and maximum value with .paramlimits. The nominal value serves as the initial guess in the optimization process. (For a complete description of the syntax of the .paramlimits command, see "paramlimits" (page 135))

[2] An optimization goal defined in terms of .optgoal commands and .measure results. T-Spice attempts to minimize the deviation of .measure results from their respective goal values defined in .optgoal. If multiple .optgoal commands are used, each .optgoal parameter has a weight value to indicate its relative importance. (For a complete description of the syntaxes, see "macro /eom" (page 107) or "optgoal" (page 122).)

[3] A .optimize command that invokes the optimization run. The optimization model and analysis name are specified using the .optimize command. (For a complete description of the .optimize command syntax, see "optimize" (page 123).)

[4] A .model command which defines the optimization algorithm, as well as optimization algorithm parameters and tolerances. (For a complete description of the .model command syntax, see "model" (page 116).)
Defining Optimization Parameters

Any parameter defined using `.param` can be used as an optimization parameter. Use the `.paramlimits` command to associate such a parameter with an optimization run, so that T-Spice can vary it during the optimization. There is no limit on how many optimization parameters an optimization run can have, but the optimization will be much faster with fewer parameters. The precise syntax for this keyword is described in “.paramlimits” (page 135).

Defining Optimization Goals

Optimization goals are defined using `.optgoal`. The function of the `.optgoal` command is to link a `.measure` result to an optimization run and to specify a goal value for the measurement. During each optimization run, T-Spice calculates an optimization function based on the differences between measurements and their corresponding optimization goals. T-Spice uses this calculation to find parameter values that minimize the deviation of measurement results from goal values.

An optimization run can have multiple `.optgoal` commands, each of which is assigned a weight. The value of the optimization function for each run is obtained by summing the weighted individual goals: each `.optgoal` contributes a term of the form weight × (goal-result)/goal.

Curve-fit optimization can be performed by using the error function measurements available on the `.measure` command.

For more information on the relevant commands, see “.macro /.eom” (page 107) and “.optgoal” (page 122).

Invoking Optimization

A `.optimize` command invokes an optimization run using the parameters and goals specified by `.paramlimits` and `.optgoal` when these commands use a matching optimization run name. Only those parameters for which the run name matches are varied during the optimization run. The model keyword defines an optimization model, which is specified using a `.model` command with optimization algorithm parameters such as iteration count limits and convergence tolerances. The analysisname option identifies the .step, .ac, .dc, or .tran analysis that will be performed to evaluate the measurements for the optimization.

After an optimization, the optimized parameter values are used in all subsequent analyses specified in the same input file. This allows for incremental optimization: some parameters can be optimized while others are held fixed; other parameters can then be optimized based on the results of the first optimization. DC analyses are performed first, followed by AC analyses, and then transient analyses. Multiple analyses of the same type are performed in the order in which they are found in the input file.

The T-Spice wizard-style user interface will guide you in setting up optimization commands.

Example 3: Optimization

```
Input A          opamp_ac.cir
Input B          myopamp.cir
Output           myopamp.sp
```

This tutorial will demonstrate the procedures to be followed in setting up an optimization command. We use the operational amplifier introduced in Example 4 to show how you can optimize output by varying
the output transistor’s lengths and widths. We can use the existing netlist by adding optimization commands to the netlist.

This exercise uses the input file `opamp_ac.cir` as a base file to which we will add optimization commands in the course of the tutorial. The modified input file that you create will be named `myopamp.cir`. For reference, we have included a file called `opamp4.cir`, which represents what `myopamp.cir` should look like at the end of the tutorial.

- Open `opamp.cir` and choose `File > Save As` to rename the file. Name the file `myopamp.cir`.
- For simplicity, remove the statements `.print ac vp(out)` and `.acmodel (*)` in the original netlist.
- Place the cursor at the beginning of any line, such as the line above the `.include` statement. Choose `Edit > Insert Command` or click the Insert Command icon in the Command toolbar ( ).
- In the left-hand pane of the Command Tool, click `Settings` and then its subset `Parameters`.
- With `Parameter type` set to `General`, enter the following values. For each value pair that you enter, click `Add`. T-Spice will add the parameter to the list.

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Parameter value</th>
</tr>
</thead>
<tbody>
<tr>
<td>I3</td>
<td>6u</td>
</tr>
<tr>
<td>w3</td>
<td>20u</td>
</tr>
<tr>
<td>I4</td>
<td>10u</td>
</tr>
<tr>
<td>w4</td>
<td>6u</td>
</tr>
</tbody>
</table>

When you have finished, the Command Tool will look like this:
Click Insert Command. T-Spice will insert the following line into the netlist:

```
.param l3=6u w3=20u l4=10u w4=6u
```

Next, edit the lengths and widths of output transistors \texttt{mp3} and \texttt{mn4} to optimize the results. For \texttt{mp3}, replace the existing length and width values with parameter names \texttt{l3} and \texttt{w3}, respectively. Enclose these parameter names in single quotes ('). For \texttt{mn4}, replace the existing length and width values with parameter names \texttt{l4} and \texttt{w4}, respectively.

Next, add a \texttt{.measure} command. Place the cursor at the beginning of any line and choose Edit > Insert Command or click the Insert Command icon. Double-click Output, then click Measure under Output (or click the Measure button). The Command Tool will appear in a form suitable for entering a \texttt{.measure} command. Type or select the following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis type:</td>
<td>AC</td>
</tr>
<tr>
<td>Measurement result name</td>
<td>gain</td>
</tr>
<tr>
<td>Measurement type</td>
<td>Signal statistics</td>
</tr>
<tr>
<td>Type of:</td>
<td>Maximum</td>
</tr>
<tr>
<td>Measured signal</td>
<td>\texttt{vdb(out)}</td>
</tr>
</tbody>
</table>

Leave all other fields blank. The Command Tool dialog will look like this:

Click Insert Command to insert the command in the netlist. T-Spice inserts the following line in the netlist:

```
.measure ac gain max vdb(out)
```
For the **bandwidth** measurement, repeat the last step, using the following values:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis type</td>
<td>AC</td>
</tr>
<tr>
<td>Measurement result name</td>
<td>bandwidth</td>
</tr>
<tr>
<td>Measurement type</td>
<td>Find-when</td>
</tr>
<tr>
<td>Find</td>
<td>x-value</td>
</tr>
<tr>
<td>When</td>
<td>Signal</td>
</tr>
<tr>
<td>Signal (name)</td>
<td>vdb(out)</td>
</tr>
<tr>
<td>equals value</td>
<td>15</td>
</tr>
</tbody>
</table>

Leave all other fields blank. The **Command Tool** dialog will look like this:

![Command Tool dialog](image)

Click **Insert Command**. T-Spice inserts the following line in the netlist:

```
.measure ac bandwidth when vdb(out)=15 cross=1
```

Now, choose **Simulation > Run Simulation**, press **F5**, or click the Run Simulation icon in the Simulation toolbar.
Under Simulation > Simulation Settings > W-Edit, select Show during simulation, then use Simulation > Run Simulation. T-Spice will simulate the circuit, and invoke W-Edit to display the following waveform:

Note that the amplifier has a gain of 43.1 dB and a bandwidth of 11.4 kHz. In the following procedure, we will use the T-Spice optimization feature to modify this design to achieve a gain of 20 dB while maximizing bandwidth.

Switch back to T-Spice. Choose Edit > Insert Command again. T-Spice displays the T-Spice Command Tool.
In the left-hand tree, click on **Optimization**, then click **Wizard** (or the **Wizard** button) to go to the first dialog, **Optimization setup**. Type `optsiz`e in the **Optimization** field and type or select **First AC Analysis** in the **Analysis name** field.

Click **Continue** to go to the next dialog, **Set optimization goals**. Enter **gain** for the first measurement name, with a target value of 20 (db) and a weight of 5. Click **Add** to add this in the **List of optimization goals**.
Repeat these steps to define the second measurement name **bandwidth**, target value **5e3** and weight **1**, and add them to the **List of optimization goals**. The **T-Spice Command Tool** dialog will look like this:

![T-Spice Command Tool](image)

Click **Continue** to go to the next dialog, **Set parameter limits**. First set an optimization goal for **Parameter name** of **l3**, using the first column of values in the table below. Click **Add** to add these values to the **List of parameters**. The optimization goals for **w3**, **l4**, and **w4** can be set in the same way.

<table>
<thead>
<tr>
<th>Field</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter name</td>
<td>l3</td>
</tr>
<tr>
<td>Minimum value</td>
<td>1u</td>
</tr>
<tr>
<td>Maximum value</td>
<td>20u</td>
</tr>
<tr>
<td>Delta</td>
<td>0.5u</td>
</tr>
<tr>
<td>Guess value</td>
<td>6u</td>
</tr>
</tbody>
</table>
When you finish, the **T-Spice Command Tool** dialog will look like this:

![T-Spice Command Tool dialog](image1.png)

Click **Continue** to go to the next dialog, **Set optimization algorithm**. In the **Name** field, type **optmod**. For all other values, accept the defaults. The **T-Spice Command Tool** dialog will look like this:

![T-Spice Command Tool dialog](image2.png)
Click **Continue** to go to the next dialog. **Insert command.** T-Spice displays your optimization commands in the dialog. Check them to make sure they are correct.

```
.optimize optsize model=optmod analysisname=step
.paramlimits optsize l3 minval=1u maxval=20u delta=0.5u guess=6u
.paramlimits optsize w3 minval=1u maxval=30u delta=0.5u guess=20u
.paramlimits optsize l4 minval=1u maxval=20u delta=0.5u guess=10u
.paramlimits optsize w4 minval=1u maxval=20u delta=0.5u guess=6u
.optgoal optsize gain=20 minval=1e-12 weight=5
.optgoal optsize bandwidth=5e3 minval=1e-12 weight=1
.model optmod opt cendif=1e-9 close=0.001 cut=2 difsz=1e-3 grad=1e-6
+ itropt=20 max=600000 parmin=0.1 relin=0.001 relout=0.001
```

If you need to change a line, click **Back** to make changes.

Click **Insert Command.** T-Spice inserts the optimization commands in the text editor.

Rerun the simulation. This time you will see the following waveform:

![myopamp](image)

**Output**

The list of optimization model parameters is summarized in the output file, followed by the optimization results and the optimized parameter values for *l3, w3, l4*, and *w4*.

```
* BEGIN NON-GRAphICAL DATA

Optimization model parameters:
Level=1
cendif=1e-009
close=0.001
cut=2
difsz=0.001
grad=1e-006
итропт=20
```

T-Spice 16 User Guide 534
max=600000
parmin=0.1
relin=0.001
relout=0.001

* END NON-GRAPHICAL DATA

* BEGIN NON-GRAPHICAL DATA

Optimization results:

- Residual = 0.359805
- Gradient norm = 0.363615
- Marquardt parameter = 16.384
- Function evaluations = 45
- Number of iterations = 16

Optimized parameter values:

13 = 9.5000e-006
w3 = 1.9000e-006
14 = 3.5000e-006
w4 = 1.20000e-006

* END NON-GRAPHICAL DATA

AC analysis results are reported in different sections for different optimization runs. In this example, only one optimization run—optsize—was performed.

AC ANALYSIS - OPTIMIZE=optsize

<table>
<thead>
<tr>
<th>Frequency&lt;Hz&gt;</th>
<th>vdb(out)&lt;dB&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00000e+000</td>
<td>2.0533e+001</td>
</tr>
<tr>
<td>1.58489e+000</td>
<td>2.0533e+001</td>
</tr>
<tr>
<td>2.51189e+000</td>
<td>2.0533e+001</td>
</tr>
<tr>
<td>3.98107e+000</td>
<td>2.0533e+001</td>
</tr>
<tr>
<td>6.30957e+000</td>
<td>2.10533e+001</td>
</tr>
</tbody>
</table>

... 

The measurement results are reported at the end of each section. Because of the difference in relative weight, gain (weight=5) has higher priority in the optimization than bandwidth (weight=1), and gain is much closer to its optimization goal.

gain = 2.0533e+001
At = 1.00000e+000

bandwidth = 1.4642e+003

T-Spice also supports HSPICE-compatible optimization commands, as the following example shows.

**Example 4: Optimization Using HSPICE-Compatible Commands**

* Input: opamp5.cir
* Output: opamp5.sp

* Circuit: opamp5.sp
* Main circuit: opamp5
.ac DEC 5 1 100MEG sweep optimize=optsize
+ results=gain bandwidth model=optmod
.model optmod opt level=1 itropt=20
.param l4=optsize(10u, 1u, 20u,0.5u) l3=optsize(6u, 1u, 20u,0.5u)
+ w4=optsize(6u, 1u, 20u,0.5u) w3=optsize(20u, 1u, 30u, 0.5u)
.measure ac gain max vdb(out) goal=20 weight=5
.measure ac bandwidth when vdb(out)=15 goal=5e3
ccomp vfl out 2pF
.cout out Gnd 2pF
mn1 vn1 vbias Gnd Gnd nmos L=10u W=6u
mn2 vml inl vn1 Gnd nmos L=6u W=6u
mn3 vfl in2 vn1 Gnd nmos L=6u W=6u
mn4 out vbias Gnd Gnd nmos l='l4' w='w4'
.include ml2_125.md
mp1 vml vml Vdd Vdd pmos L=6u W=6u
mp2 vfl vml Vdd Vdd pmos L=6u W=6u
mp3 out vfl Vdd Vdd pmos l='l3' w='w3'
.print ac vdb(out)
.vbias vbias Gnd 0.8
Vdd Vdd Gnd 5.0
vdiff in2 inl -0.0007 AC 1.0 90
vin1 inl Gnd 2.0
* End of main circuit: opamp5

Output

The output of this example is the same as the output “Example 3: Optimization” on page 526.
The following references may be consulted for further information on various aspects of circuit design, modeling, and simulation.


Index

SYMBOLS

, 73, 74, 101, 137, 141, 488
^ (caret character), 45
? (wildcard character), 138
.option password command
    described, 61
* (wildcard character), 138
+121752, 373

A
abbreviations, metric, 69
.abs(x), 71
absolute charge tolerance, 37
absolute value of x, 71
.abs_tol option
    DC analysis, impact on, 33
    numnd, relationship to, 37
    simulation concepts of, 36
    speed vs. accuracy, impact on, 38
    transient analysis, impact on, 34
.ac
    device parameter
        resistor, 198
    keyword, 109, 178
AC analysis
    examples, 77
    multiple simulation runs, 80
    optimization and, 109, 123
    parameters, 77
    performing with .ac, 76
    results, printing, 138, 152
    small-signal, 36
AC analysis
    small-signal output, 60, 61, 79
.ac command, 76–78, 120
accuracy
    Fourier analysis, 95
    vs speed, trading off, 38
    tolerance for overall, 37
    transient analysis, 37
.accurate option, 126
.acm, 495–497
.acmodel command
    .ac command, using with, 60, 61, 79
    described, 79
    examples, 79
    .op command, using with, 60, 61, 79, 121
        in subcircuits, 162
    syntax, 60, 61, 79
.acos(x), 71
active region of operation, 382
active window, 42
.ad device parameter
    MOSFET, 193, 465, 488
addition
    expression for, 70
    line continuation and, 69
    operator precedence of, 71
additive inverse, 70
.agauss keyword, 132
 ako keyword, 116
algebraic expressions
    See expressions
algorithm, optimization, 526
algorithms
    backward differentiation, 35
    Fourier transform example, 96
    Gear’s BDF, 35
    gmindc stepping, 34, 236, 238, 244
    Levenberg-Marquardt, 118
optimization
    described, 39
    parameters for, 116–118
    specifying with .model, 160
    time derivative of charge calculation, 37
    trapezoidal integration, 35
aliasing subcircuit pin names, 67
.alter command
    deleting library sections with, 90
    described, 80
    examples, 80–81
    output, 81
    syntax, 80
.amax keyword, 111
ambient temperature, 198
.amin keyword, 112
ampersand, literal, 48
analysis
    DC transfer, 519
    Monte Carlo, 522
.analysys keyword, 123
.analysisname keyword, 76–77
AND gate, zero-delay, 225
angles, in expressions, 70
anode terminal, notation for, 138
area
calculation method
See acm
effective, MOSFET, 194, 495–497
area device parameter
BJT, 172
diode, 187
JFET, 191
MESFET, 192
areab device parameter
BJT, 172
areac device parameter
BJT, 172
arguments
.print command, 138
defined, 75
interrupting with comments, 68
reporting, 111
arithmetic, finite precision, 36
as device parameter
MOSFET, 193, 465, 488
asin(x), 71
assert command
described, 82
examples, 83
syntax, 82
at keyword, 110, 113
atan(x), 71
atan2(x,y), 71
aunif keyword, 132
autostop option, 109
average power consumption, 136
average signal value, 111
avg keyword, 112

B
backslash
literal, 48
in Unix-style expressions, 47
backward differentiation formulas, 35
Backward Euler method, 35
backward searching, 46
bandwidth, 110
base of natural logarithms, 372
base terminal, notation for, 138
base units, 69
base-width modulation, BJT, 380
batch simulations
controlling, 55
See also Simulation Manager
beginning
of line, specifying, 47
of measurement, specifying, 110
bimodal distribution, 132
bin description parameters, 459
binary
notation, in device statements, 181
output, 152
bipolar junction transistor
See BJT
bit keyword, 181
bit patterns, 181
BJT
device parameters, 172, 172–173
device statement, 172
device terminal names, 138
gmin conductance, in DC analysis, 34
level/model cross reference, 367
noise types, 140
specifying, for .model, 116
temperature effects
energy gap, 389
BJT Level 1 (Gummel-Poon)
capasitance equations, 387
base-collector, 386–387
base-emitter, 385, 386
substrate, 387
temperature-dependent, 391–392
current equations
base charge, 382, 383
collector and base, 383, 384
emitter, 383
substrate, 384
device model, 373, 393, 399
device parameters, 172
geometry considerations, 380
large-signal model, 379
model parameters
base charge parameters, 374–375
DC current parameters, 374
junction capacitance parameters, 375–376
level 2 parameters, 377
optional parameters, 382
parasitic capacitance parameters, 375
parasitic resistor parameters, 375
temperature effect parameters, 377–379
transit time parameters, 376
noise types, 140
regions of operation, 382
scaling equations, 381
temperature effects
capacitance equations, 391–392
saturation current and Beta, 389–391
variable base resistance, 384
BJT Level 10 (Modella)
device model, 399
model documentation, 399
model parameters, 399
BJT Level 6 (Mextram), 393
device model, 393
model documentation, 393
model parameters, 393
BJT Level 9 (VBIC)
device model, 394–398
device parameters, 172
equations, 398
model parameters, 394–398
noise types, 140
temperature, 398
blank lines, in netlist, 68
Boltzmann’s constant, 372
Boolean options, 125
bounding errors
See tolerances
braces, in syntax, 75
brackets, in syntax, 75
branch current tolerances, 36
BSIM1
See MOSFET Level 28
See MOSFET Level 4/13 (BSIM1)
bulk terminal, notation for, 138
bus keyword, 181
bus patterns, 181
buses, defining, 168, 169

c
device parameter
capacitor, 174
resistor, 198
transmission line, 203
key letter, 174
keyword, .model, 116

capacitance
as derivative of charge function, 214
parameters describing
diode model, 404–405
MESFET model, 421
MOSFET Level 49/53 model, 457
resistor, 199
sidewall, MOSFET, 196
transmission line, distributed, 203
capacitor
conductance, adding for DC analysis, 34
device model
described, 400
equations describing, 400–401
.model type, specifying, 116
nonlinear example, 214
as open circuit, 33
parallel-plate example, 214
carry
parameters
device statement, 174
model definition, 400
voltage-controlled, 214
caret (^)
exponentiation with, 71
in Find command, 46
searching for, 45
carrier frequency, current source, 180
case sensitivity
of metric prefixes, 69
of node and device names, 66
cathode terminal, notation for, 138
cdis, 376
ceil(x), 71
cendif keyword, 117
charge
effects, assumed for DC analysis, 121
elementary electron constant, 372
reporting with .print, 138
tolerances, 37–38
.chargetol option
discretization error effects, 34
simulation concepts of, 37–38
.checkpoint command
defined, 85
syntax, 84
.described, 84
cchg device parameter, 210
circuit description
See netlists
circuit equations
linearizing, 36
solving
See algorithms
circuits
defined, 33
open, 33
short, 33
close keyword, 117, 118
coefficients, 35
coefficients, of polynomial, 183
collector terminal, notation for, 138
color coding of T-Spice input files, 44
columns
in .data command, 87
string headings for, 138
Command Tool, 20, 22
commands
inserting, 20
comments
in continued lines, 68
C-style, 44, 68
delimiters for, 44, 67
at end of netlist, 91
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>in expressions, 70</td>
<td></td>
</tr>
<tr>
<td>syntax in T-Spice, 67</td>
<td></td>
</tr>
<tr>
<td>complex voltage, reporting, 139</td>
<td></td>
</tr>
<tr>
<td>computations</td>
<td></td>
</tr>
<tr>
<td>AC magnitude/phase differences, 322</td>
<td></td>
</tr>
<tr>
<td>average, 111</td>
<td></td>
</tr>
<tr>
<td>derivatives, 112, 117</td>
<td></td>
</tr>
<tr>
<td>maximum, 111</td>
<td></td>
</tr>
<tr>
<td>minimum, 111</td>
<td></td>
</tr>
<tr>
<td>noise, 120</td>
<td></td>
</tr>
<tr>
<td>peak-to-peak, 111</td>
<td></td>
</tr>
<tr>
<td>root-mean-square, 111</td>
<td></td>
</tr>
<tr>
<td>conditional commands</td>
<td></td>
</tr>
<tr>
<td>syntax, 101</td>
<td></td>
</tr>
<tr>
<td>conductance</td>
<td></td>
</tr>
<tr>
<td>minimum</td>
<td></td>
</tr>
<tr>
<td>transient analysis, 35</td>
<td></td>
</tr>
<tr>
<td>stepping, in DC analysis, 34</td>
<td></td>
</tr>
<tr>
<td>transmission line, distributed, 203</td>
<td></td>
</tr>
<tr>
<td>.connect command</td>
<td></td>
</tr>
<tr>
<td>examples, 86</td>
<td></td>
</tr>
<tr>
<td>syntax, 86</td>
<td></td>
</tr>
<tr>
<td>described, 86</td>
<td></td>
</tr>
<tr>
<td>connectivity errors, 39, 40</td>
<td></td>
</tr>
<tr>
<td>constants, defined, 372</td>
<td></td>
</tr>
<tr>
<td>control options</td>
<td></td>
</tr>
<tr>
<td>See .options command</td>
<td></td>
</tr>
<tr>
<td>convergence</td>
<td></td>
</tr>
<tr>
<td>enabling with .nodeset, 119</td>
<td></td>
</tr>
<tr>
<td>errors</td>
<td></td>
</tr>
<tr>
<td>See nonconvergence errors</td>
<td></td>
</tr>
<tr>
<td>factors influencing, 37</td>
<td></td>
</tr>
<tr>
<td>gradient test for, 117</td>
<td></td>
</tr>
<tr>
<td>parasitic diode improvements for, 194</td>
<td></td>
</tr>
<tr>
<td>relative parameter variation test for, 118</td>
<td></td>
</tr>
<tr>
<td>tolerances for, 36</td>
<td></td>
</tr>
<tr>
<td>corners, rounding</td>
<td></td>
</tr>
<tr>
<td>pulse waveform, 179</td>
<td></td>
</tr>
<tr>
<td>vectorized current source, 181</td>
<td></td>
</tr>
<tr>
<td>with polynomial, 179</td>
<td></td>
</tr>
<tr>
<td>cos(x), 71</td>
<td></td>
</tr>
<tr>
<td>cosh(x), 71</td>
<td></td>
</tr>
<tr>
<td>cosine functions, 71</td>
<td></td>
</tr>
<tr>
<td>coupled inductors</td>
<td></td>
</tr>
<tr>
<td>See mutual inductor, 197</td>
<td></td>
</tr>
<tr>
<td>coupled transmission line</td>
<td></td>
</tr>
<tr>
<td>device model, 402</td>
<td></td>
</tr>
<tr>
<td>device parameters, 177</td>
<td></td>
</tr>
<tr>
<td>examples, 402</td>
<td></td>
</tr>
<tr>
<td>model parameters, 402</td>
<td></td>
</tr>
<tr>
<td>model type specification, 116</td>
<td></td>
</tr>
<tr>
<td>cpl keyword, .model, 116</td>
<td></td>
</tr>
<tr>
<td>cross keyword, 110, 113</td>
<td></td>
</tr>
<tr>
<td>crossing, defined, 111</td>
<td></td>
</tr>
<tr>
<td>C-style comments, 68</td>
<td></td>
</tr>
<tr>
<td>csw</td>
<td></td>
</tr>
<tr>
<td>keyword, .model, 116</td>
<td></td>
</tr>
<tr>
<td>switch model type, 201</td>
<td></td>
</tr>
<tr>
<td>cur parameter</td>
<td></td>
</tr>
<tr>
<td>voltage-controlled current source, 210</td>
<td></td>
</tr>
<tr>
<td>current</td>
<td></td>
</tr>
<tr>
<td>complex, reporting, 139</td>
<td></td>
</tr>
<tr>
<td>directionality convention for, 138</td>
<td></td>
</tr>
<tr>
<td>gain, reporting, 164</td>
<td></td>
</tr>
<tr>
<td>reporting with .print, 138</td>
<td></td>
</tr>
<tr>
<td>second-order effects, BJT, 37</td>
<td></td>
</tr>
<tr>
<td>subthreshold</td>
<td></td>
</tr>
<tr>
<td>See subthreshold current source</td>
<td></td>
</tr>
<tr>
<td>current source</td>
<td></td>
</tr>
<tr>
<td>current-controlled</td>
<td></td>
</tr>
<tr>
<td>See current-controlled current source</td>
<td></td>
</tr>
<tr>
<td>device statement, 178–182</td>
<td></td>
</tr>
<tr>
<td>nonlinear, 211, 218</td>
<td></td>
</tr>
<tr>
<td>powerup simulation of, 37</td>
<td></td>
</tr>
<tr>
<td>sweeping, 159</td>
<td></td>
</tr>
<tr>
<td>voltage-controlled</td>
<td></td>
</tr>
<tr>
<td>See voltage-controlled current source</td>
<td></td>
</tr>
<tr>
<td>current-controlled</td>
<td></td>
</tr>
<tr>
<td>current source, 183–184</td>
<td></td>
</tr>
<tr>
<td>switch, 116</td>
<td></td>
</tr>
<tr>
<td>voltage source, 185–186</td>
<td></td>
</tr>
<tr>
<td>curve-fit optimization, 526</td>
<td></td>
</tr>
<tr>
<td>cut keyword, 117</td>
<td></td>
</tr>
<tr>
<td>cut-off region of operation, 382</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
</tr>
<tr>
<td>d</td>
<td></td>
</tr>
<tr>
<td>key letter, 187</td>
<td></td>
</tr>
<tr>
<td>keyword, .model, 116</td>
<td></td>
</tr>
<tr>
<td>suffix, decimal notation, 181</td>
<td></td>
</tr>
<tr>
<td>d2dt(f), 73</td>
<td></td>
</tr>
<tr>
<td>d2dx(f,x), 74</td>
<td></td>
</tr>
<tr>
<td>damping factor</td>
<td></td>
</tr>
<tr>
<td>sinusoidal current source, 180</td>
<td></td>
</tr>
<tr>
<td>dashes, in expressions, 71</td>
<td></td>
</tr>
<tr>
<td>.data command</td>
<td></td>
</tr>
<tr>
<td>.alter command with, 80</td>
<td></td>
</tr>
<tr>
<td>in DC sweeps, 89</td>
<td></td>
</tr>
<tr>
<td>ending with .enddata, 92</td>
<td></td>
</tr>
<tr>
<td>in error measurements, 109</td>
<td></td>
</tr>
<tr>
<td>examples, 87–88</td>
<td></td>
</tr>
<tr>
<td>optimization with .measure, 109</td>
<td></td>
</tr>
<tr>
<td>parameter sweeps with, 159, 160</td>
<td></td>
</tr>
<tr>
<td>syntax, 87</td>
<td></td>
</tr>
<tr>
<td>data command</td>
<td></td>
</tr>
<tr>
<td>described, 87</td>
<td></td>
</tr>
<tr>
<td>data command, 519</td>
<td></td>
</tr>
<tr>
<td>data keyword, 158, 159, 160</td>
<td></td>
</tr>
<tr>
<td>db(x), 71</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td></td>
</tr>
</tbody>
</table>
Index

device keyword, 79
device models
nonlinear, 33
parameters, 116

See model parameters
physical constants for, 372
syntax for, 372
device name syntax, 40, 66

See also names
device parameters
BJT
area, 172
areab, 172
areac, 172
M, 172
scale, 173
capacitor
c, 174
dtemp, 174
I, 174
POLY, 174
scale, 174
tc1, 174
tc2, 174
w, 174
coupled transmission line
length, 177
lumps, 177
lumptype, 177
current source
AC, 178
DC, 178
diode
area, 187
L, 187
M, 187	
tables, 187
W, 187
inductor
dtemp, 188
M, 188
r, 188
scale, 188
tc1, 188
tc2, 188
instance
M, 190
JFET
area, 191
M, 191
tables, 191
MESFET
area, 192
I, 192
M, 192
tables, 192
Index

w, 192
MOSFET
  ad, 193, 465, 488
  as, 193, 465, 488
  frbody, 488
  geo, 195
  l, 193, 465
  M, 193, 465, 488
  min, 488
  nbc, 488
  nrd, 193, 465, 488
  nrs, 193, 465, 488
  pd, 193, 465, 488
  ps, 193, 465, 488
  psbcp, 488
  rth0, 488
  tables, 193, 465, 488
  vbsusr, 488
  w, 193, 465
resistor
  ac, 198
  c, 198
  dtemp, 198
  l, 198
  m, 198
  noise, 198
  r, 198
  scale, 198
  tc1, 198
  tc2, 198
  w, 198
transmission line
  c, 203
  f, 203
  g, 203
  l, 203
  length, 203
  lumps, 203
  lumptype, 203
  nl, 203
  r, 203
  td, 203
  z0, 203
voltage-controlled current source
  chg, 210
  cur, 210
  Laplace, 210
  POLY, 210, 217
voltage-controlled voltage source
  Laplace, 220
  POLY, 220
device state printout
  BJT Level 1 (Gummel-Poon), 143
  BJT Level 9 (VBIC), 144
  capacitor, 146
  current-controlled current source, 146
  current-controlled voltage source, 146
described, 143
diode, 146
  inductor, 147
  MOSFET, 147
  resistor, 149
  voltage-controlled current source, 149
  voltage-controlled voltage source, 149
device statement syntax, 171
device values
defaults listed, 366
devices
  active, total number of, 57
  noise contributions of, 139
  passive, total number of, 57
  subcircuit
    accessing, 100
    scope, 162
  total number of, 57
dielectric permittivities, 372
difference measurements, 110
differentiation, 35
diffusion capacitance, 408
difsiz keyword, 117
diode
  added conductance, in DC analysis, 34
  capacitance equations
    diffusion, 408
    junction, 408–409
    polysilicon (contact electrode), 409
  current equations, 406–408
  device model, 403–416
  device statements, 187
  device terminal notation, 138
  equations, 406–415
  Fowler-Nordheim model, 414–415
  capacitance, 415
  current equations, 415
  geometric and scaling effects
    Level 1, 409–410
    Level 3, 410–411
  Juncap2 model documentation, 403
  level/model cross reference, 367
  .model type, specifying, 116
  model parameters
    capacitance, 404–405
    DC, 404
    Fowler-Nordheim model, 406
    geometric, 403–404
    model selectors, 403
    noise, 405
    scaling, 403, 404
    temperature, 405–406
  noise types, 141
  Philips Advanced Diode 500 model documentation, 403
Schottky barrier, 415, 416

temperature effects
  breakdown voltage, 412
  contact potential, 413
  energy gap equation, 411
  grading coefficient, 414
  junction capacitance, 414
  resistance, 414
  saturation current, 412
  transit time, 412
directionality convention, current, 183
directories
  input files, 519
tutorial, 519
discrete optimization, 133, 135
discretization error, 34
  BDF methods and, 35
described, 34
discretization error tolerance
  See also chargetol, relchargetol
discretized differentiation formulas, 35
display area, 18
displaying simulation output, 19, 54
distributions, in parametric analysis, 38
divided difference approximation, 37
division, 70, 71
docking the Simulation Manager, 54
documentation conventions, 14
drain terminal, notation for, 138
dtemp device parameter
  capacitor, 174
  inductor, 188
  resistor, 198

E

e
  key letter, 220
ea, 72, 372
Ebers-Moll model, 373, 382, 393, 399
Edit >
  Find, 45
  Goto Line, 45
  Replace, 46
editing text, 44
effective areas and perimeters, 495–497
electrical specifications, computing, 109
electron charge, elementary, 372
  , 101
emitter terminal, notation for, 138
encrypted files, decrypting, 64
encryption commands
  .option password, 61
  .protect, 59
  .unprotect, 60

.visible, 60
.end command, 91
dead line, 47
dead of measurement, specifying, 110
.enddata command, 87, 92
  , 101
.endl command, 93, 104
.ends command, 94, 162
energy gap
  BJT Level 1 (Gummel-Poon), 389
diode, 411
eom command
defined, 107
equations, solving
  See algorithms
equivalent noise spectral density, 120
err, err1, err2, err3 keywords, 114
error
  measurements, 114–115
  messages
    comments as cause of, 68
    optimization, 122
  error function measurements, 115
errors, 518
  connectivity, 39, 40
  convergence, 40
discretization, 34, 37
  syntax, 39, 40
errx,y), 71
event-triggered measurements
  See find-when measurements
examples
  Monte Carlo Analysis, 522–525
  Optimization, 526–535
  Optimization Using HSPICE-Compatible Commands, 535
  Parametric Sweep, 519–522
exp keyword, 178
exp(x), 72
exponential
  parameter sweeps, 38
  waveform
    current source, 178
    voltage source, 205
exponentiation, in expressions, 71–72
exporting
  schematic files, 42
expressions
  angles in, 70
  evaluating with .measure, 114
  frequency, 139
  multiline, 68, 69
  operator precedence in, 70
  parameters in, 131
  printing values for, 137, 138
  in signal statistics measurements, 112

Index

summarized, 70
in trigger/target measurements, 111
vectorized waveform, 181
in voltage source definition, 205
See also operations

external
data
difference measurements with, 109
incorporating in simulation, 87
models
device instances of, 190
extraction, from L-Edit, 42

F

f
device parameter, transmission line, 203
key letter, 183
metric prefix, 69
fabrication conditions, sensitivity to, 38
fabs(x), 72
Failed, simulation status, 54
fall crossing, 111, 113
fall keyword, 110, 111, 113
fall times, measuring, 110
femto-, 69
flicker noise, 140, 141
file synchronization, 42
files
including in netlist, 103
find keyword, 112
finding text, 45, ??–46
find-when measurements, 112–114, 115
Finished, simulation status, 54
finite precision arithmetic, 36
first line, as comment, 67
fixed-point notation, 69
flicker noise, 140
floating nodes, 39, 100
floating point numbers, 69
floating the Simulation Manager, 54
floor(x), 72
fmod(x,y), 72
font options, 30
Forward Euler predictor, 35
.four command, 95–96
Fourier analysis
described, 95
output from, 96
four-terminal transistor, 192, 193
Fowler-Nordheim model
capacitance equations, 415
current equations, 415
parameters, 406
frequency
circuit dependence on, 76
difference measurements of, 110
fundamental, in FFT, 95
modulating current sources by, 180
modulating voltage sources by, 207
resolution, in FFT, 95
sinusoidal current source, 180
transmission line, 203
‘frequency()’, 139
frequency-modulated waveform, 180
from keyword, 112, 114
ft keyword, 181
functions
built-in, 71
See also user-defined functions
fundamental frequency
FFT analysis, 95
frequency-modulated waveform, 180

G
g
metric prefix, 69
device parameter, transmission line, 203
key letter, 210
gamma-type lumps, 177, 204
gate models
AND, 225
NAND, 225
NOR, 225
OR, 225
gate terminal, notation for, 138
gauss keyword, 132
Gaussian distribution, 132
Gear’s BDF Method, 35
gap parameter, MOSFET, 195
geometry and scaling parameters
BJT Level 1 (Gummel-Poon), 380–382
diode, 403–404
MESFET, 422
giga-, 69
global
minimum, computing, 112
names, 67, 97
parameters
overriding, 131
sweeping, 159
simulation options
See .options command
.global command, 97, 162
gmin option, 35
gmindic option, 34
gmindic stepping, 34
GND, 97
goal keyword, 109–110
goal, for optimization, 525, 526
goals

optimization, 39, 122, 160
goto line number, 45

grad keyword, 117
gradient descent, 39
gradients
See derivatives
grading coefficient
diode, 414
gramp option, 34, 236, 238, 244
graphical user interface
See user interface
ground, in netlists, 66
guessval keyword, 135
Gummel-Poon model, 373, 379, 393, 399

H

h

key letter, 185

suffix, hexadecimal notation, 181
half-interval, rounding, 179, 181
harmonic distortion, in FFT, 95
.hd1 command
examples, 98
syntax, 98
described, 98
hexadecimal notation, 181
hierarchical circuits
See subcircuits
hierarchical notation, 67, 190
high-bias effects
BJT model, 373, 393, 399
high-level injection, BJT, 380
HSPICE, 535
HSPICE compatibility, 310
HSPICE compatible sweep syntax, 160
ht keyword, 181
hybrid RGT lumps, 177, 204
hyperbolic functions, 72
hyphens, in expressions, 71
hysteresis, in switches, 201

I

i

key letter, 178
.ic command

in .alter blocks, 80
for DC operating point, 121
defined, 100
in subcircuits, 162
ideal voltage supply, 205

idt(f), 74
idx(f,x), 74
, 101
if(a,b,c), 72
imaginary components, printing
complex current, 139
complex voltage, 139
impedance, of transmission line, 203
.include command

in .alter blocks, 80
compared with .lib, 104
defined, 103
parameter scope in, 131
including library files, 104
increasing crossing, 113
incremental optimization, 526
independent variables
analysis types and, 109
difference measurements of, 110
reporting, 111
inductance
of inductor, 189
transmission line, distributed, 203
inductors

coupled
See mutual inductor
DC analysis treatment of, 33
device statements, 188–189
initializing currents in, 100
parallel, 188
temperature dependence, 188
initial conditions
DC operating point, 121
powerup from zero, 37
for transient analysis, 165
guesses, for DC operating point, 37, 119
parameter values, optimization, 133, 135
seed value, for random numbers, 159
.inoise keyword, 139
input deck

See netlists
input files, 519
See netlists
input noise spectral density, 139
input signals

previewing, 166
Insert mode (Ins), 44
instance

device statement, 190–??
instantaneous power, 136
int(x), 72
integer expressions, 72
integral keyword, 112
integration
functions, 73
simulation concepts, 34, 35
interpolate keyword, 95
interpolation
  Fourier analysis, 95
  linear, 344
intrinsic carrier concentration at 300K, 372
inverse region of operation, 382
inverse trigonometric functions, 71
inverse, additive, 70
inverter
  zero-delay, 224
inward current positive, 138
iss, 384
italics, in syntax, 75
iterations
  initial conditions, effect on, 37
  limits on
    described, 33, 34, 37
    optimization, 117
    source stepping, 37
iterative ladder circuit (ILC) expansion, 177
itropt keyword, 117

J
j
  key letter, 191
JFET
  charge equations, 419
  current equations, 418–419
  device model, 417–419
  device statements, 191
  device terminal names, 138
  equations, 418–419
  large signal model, 418
  level/model cross reference, 368
  model parameters, 417
  noise types, 141
  specifying, for .model, 116
  junction field effect transistor
    See JFET

K
k
  key letter, 197
  metric prefix, 69
k (Boltzmann’s constant), 372
KCL equations, 36–37
key letter
  c, 174
  d, 187
  e, 220
  f, 183
  g, 210
  h, 185
  i, 178
  j, 191
  k, 197
  l, 188
  m, 193
  q, 172
  r, 198
  s, 201
  t, 203
  u, 177
  v, 205
  x, 190
  z, 192
key letters
  in .print statements, 138
  defined, 67
  keyword groups, 27
  keyword groups in text files, 27
  keywords, 27
    reserved, list of, 66
  kilo-, 69
  Kirchoff’s Current Law, 33, 36
    See also KCL equations
"knee" currents, BTJ, 380
  kth order BDF method, 35

L
l
  device parameter
    capacitor, 174
    MESFET, 192
    MOSFET, 193, 465
    resistor, 198
    transmission line, 203
  key letter, 188
L device parameter
diode, 187
language conventions, 66
Laplace voltage-controlled current source parameter, 210
Laplace voltage-controlled voltage source parameter, 220
large-signal models
  MESFET, 424
  MOSFET Level 1/2/3, 431
  resistor, 499
large-signal time-domain analysis
  See transient analysis
last keyword, 111, 113
lateral geometry
  BJT, 380
ldexp(x,y), 72
L-Edit, 42
  features, 14
length device parameter
  coupled transmission line, 177
  transmission line, 203
length parameters
capacitor, 174
diodes, 187
MESFET device, 192
MOSFET Level 49/53, 457–458
resistor, 199
transmission line, 203
level
  model selector
    .model keyword, 116
  optimization method selector, 118
Levenberg-Marquardt algorithm, 118
.lib command
  in .alter blocks, 80
described, 104
ending with .endl, 93
modifying with .del lib, 90
libraries
definitions
  See .lib command, 93
  file formats, 104–105
omitting sections of, 80
limit keyword, 132
lin keyword, 76, 77
line
  beginning of, as search input, 47
  breaks
    searching for, 45, 47
  continuation
    in expressions, 69, 70
  end of, as search input, 47
  numbers
    in debugging syntax, 40
    using, 45
linear
  region of operation
    MOSFET, 436
    MOSFET Level 2, 436–438
  sweeps
    AC analysis, 76, 77
    parameter, 38
    with .step, 159
linear sweep, 519
linear system solution
  parallel processing, 41
linearized small-signal models, 76
linearizing KCL equations, 36
list keyword, 76, 77, 158, 160
list sweep, 519
.load command
  defined, 106
  syntax, 106
local scope, 190
logarithmic expressions, 72
logarithmic sweep, 519
logarithmic sweeps
  AC analysis, 77
  of parameters, 38
  with .step, 159
lossless transmission line, 203
lossy transmission line, 203
lt keyword, 181
lumps device parameter
  coupled transmission line, 177
  transmission line, 203
lumptype device parameter
  coupled transmission line, 177
  transmission line, 203
M
m
  metric prefix, 69
device parameter, resistor, 198
key letter, 193
M device parameter
  BJT, 172
diode, 187
inductor, 188
instance, 190
JFET, 191
MESFET, 192
MOSFET, 193, 465, 488
.macro command
  defined, 107
magnitude
  calculation methods, AC analysis, 322
  orders of, notation, 69
  printing
    current, 139
    input noise spectral density, 139
    output noise spectral density, 139
    voltage, 139
Maher-Mead
  See MOSFET Level 5
.malias command
  defined, 108
manual line break, in Find command, 46
mathematical operator precedence, 70
matrix-vector operations, 36
max keyword, 112, 118
max(expression1, expression2), 72
maximum
  number of iterations
    DC analysis, 37
  optimization, 117
  source stepping, 37
  optimization parameter value, 133
Index

signal value, 109, 111
of two expressions, evaluating, 72
maxord option
simulation concepts, 34, 35
maxval keyword, 135
.meas command
See .measure command
measname, 160
.measure command
dc, 89
error functions, 109, 114, 115
example with .optimize, 124
examples, 115
expression evaluation, 114
external data, using with, 87
find-when and derivative measurements, 112, 114
optimization and, 122
plotting results from, 158
results
expressions involving, 114
as optimization goal, 160
signal statistics, 111, 112
in subcircuits, 162
suppressing output, 110
syntax, 109–115
trigger/target measurements, 110
, 520, 525, 526
output, 518
plotting, 518, 521
measurement types, 110, 111
meg metric prefix, 69
mega-, 69
memory
updating to most recent file, 27
MEMS, 214
menu bar, 16, 17
See also user interface
MESFET
capacitance equations, 425–426
conductance, in DC analysis, 34
current equations, 424–425
device model, 420–429
device statements, 192
device terminal names, 138
equations, 424–429
large-signal model, 424
level/model cross reference, 368
model parameters, 420, 423
ACM parameters, 422–423
capacitance, 421
DC, 420–421
geometry, 422
noise, 421
submodel selectors, 420
temperature, 423
N model type specification, 116
P model type specification, 116
temperature dependence, 426–429
messages, output
See Simulation Output window
metal oxide semiconductor field effect transistor
See MOSFET
metal semiconductor field effect transistor
See MESFET
metric prefixes, 40, 69
micro-, 69
microelectromechanical systems (MEMS), 214
mil metric prefix, 69
milli-, 69
min keyword, 112
min(expression1, expression2), 72
minima
multiple occurrences of, 112
minima, finding, 72
minimum
conductance, 34, 236, 238, 244
global, 112
optimization parameter value, 133
signal value, 109, 111
source step size, 37
minsrefstep option, 33, 37
minval keyword, 109, 110, 122, 135
mode, 137, 152, 166
model
names
See names parameters
extracting, 87
specifying, 116
selectors
diode, 403
MOSFET Level 49/53, 453
temperature, nominal, 372
types, for .model, 116–??
.model command, 525
in .alter block, 80
defined, 116, 371
examples, 118
in library files, 105
optimization algorithm parameters, 160
type field, 116
model equations
listing all supported, 365
model evaluation
parallel processing, 41
model keyword, 123, 158, 160, 526
model parameters
BJT Level 1 (Gummel-Poon)
.jbf | ikf | ik, 375
af, 377
bex, 377
bexv, 377
T-Spice 16 User Guide 550
```plaintext
Index

T-tr2, 378
tvaft1, 379
tvaft2, 379
tvar1, 379
tvar2, 379
tvc, 379
tve, 379
tvs, 379
vaf | vbf, 375
var | vb | vbb, 375
vb | vbb | var, 375
vbf | vaf, 375
vjc | pc, 376
vje | pe, 376
vjs | psb, 376
vtf, 376
xcjc | cdis, 376
xtb | tb | tcb, 379
xtf, 376
xti, 379

BJT Level 9 (VBIC)
afn, 394
ajc, 394
aje, 394
ajs, 394
art, 394
avc1, 394
avc2, 394
bfn, 394
cbco, 394
cbeo, 394
coso, 394
cjc, 394
cjcp, 394
cjo, 394
cjep, 394
cht, 394
dear, 394
dtemp, 395
ea, 395
eaic, 395
eaie, 395
eais, 395
eanc, 395
eane, 395
eans, 395
eap, 395
fc, 395
gamm, 395
hrcf, 395
ibbe, 395
ibci, 395
ibcip, 395
ibcn, 395
ibcnp, 395

ibei, 395
ibep, 395
iben, 395
ibenp, 395
ikf, 395
ikp, 395
ikr, 395
is, 395
ismin, 398
isp, 395
ispmin, 398
isrr, 395
itf, 395
kfn, 395
level, 398
me, 395
mcmin, 398
me, 395
memin, 398
ms, 396
msmin, 398
nbbe, 396
nci, 396
ncip, 396
ncn, 396
ncnp, 396
nei, 396
nen, 396
nf, 396
nfp, 396
nkf, 396
nr, 396
pc, 396
pe, 396
ps, 396
qbm, 396
qco, 396
qtf, 396
rbi, 396
rbc, 396
rbpm, 398
rbx, 396
rci, 396
rcx, 396
re, 396
rs, 396
rth, 396
tavc, 396
td, 396
tf, 396
tnbe, 396
tnf, 396	nom, 397
tr, 397
tref, 397
xvbe1, 397```
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>tvbbe2</td>
<td>397</td>
</tr>
<tr>
<td>vbbe</td>
<td>397</td>
</tr>
<tr>
<td>vef</td>
<td>397</td>
</tr>
<tr>
<td>ver</td>
<td>397</td>
</tr>
<tr>
<td>vers</td>
<td>397</td>
</tr>
<tr>
<td>version</td>
<td>397</td>
</tr>
<tr>
<td>vo</td>
<td>397</td>
</tr>
<tr>
<td>vrev</td>
<td>397</td>
</tr>
<tr>
<td>vrt</td>
<td>397</td>
</tr>
<tr>
<td>vtf</td>
<td>397</td>
</tr>
<tr>
<td>wbe</td>
<td>397</td>
</tr>
<tr>
<td>wsp</td>
<td>397</td>
</tr>
<tr>
<td>xii</td>
<td>397</td>
</tr>
<tr>
<td>xikf</td>
<td>397</td>
</tr>
<tr>
<td>xin</td>
<td>397</td>
</tr>
<tr>
<td>xis</td>
<td>397</td>
</tr>
<tr>
<td>xisr</td>
<td>397</td>
</tr>
<tr>
<td>xrb</td>
<td>397</td>
</tr>
<tr>
<td>xribi</td>
<td>397</td>
</tr>
<tr>
<td>xrbp</td>
<td>397</td>
</tr>
<tr>
<td>xrbx</td>
<td>397</td>
</tr>
<tr>
<td>xrc</td>
<td>397</td>
</tr>
<tr>
<td>xrci</td>
<td>397</td>
</tr>
<tr>
<td>xrcx</td>
<td>397</td>
</tr>
<tr>
<td>xre</td>
<td>397</td>
</tr>
<tr>
<td>xrs</td>
<td>397</td>
</tr>
<tr>
<td>xtf</td>
<td>397</td>
</tr>
<tr>
<td>xvo</td>
<td>397</td>
</tr>
<tr>
<td>capacitor</td>
<td></td>
</tr>
<tr>
<td>Cap</td>
<td>400</td>
</tr>
<tr>
<td>Capsw</td>
<td>400</td>
</tr>
<tr>
<td>Cox</td>
<td>400</td>
</tr>
<tr>
<td>Del</td>
<td>400</td>
</tr>
<tr>
<td>Di</td>
<td>400</td>
</tr>
<tr>
<td>L</td>
<td>400</td>
</tr>
<tr>
<td>Shrink</td>
<td>400</td>
</tr>
<tr>
<td>Tc1</td>
<td>400</td>
</tr>
<tr>
<td>Tc2</td>
<td>400</td>
</tr>
<tr>
<td>Thick</td>
<td>400</td>
</tr>
<tr>
<td>Tref</td>
<td>400</td>
</tr>
<tr>
<td>W</td>
<td>400</td>
</tr>
<tr>
<td>coupled transmission line</td>
<td></td>
</tr>
<tr>
<td>[c]</td>
<td>402</td>
</tr>
<tr>
<td>[g]</td>
<td>402</td>
</tr>
<tr>
<td>[I]</td>
<td>402</td>
</tr>
<tr>
<td>[r]</td>
<td>402</td>
</tr>
<tr>
<td>diode</td>
<td></td>
</tr>
<tr>
<td>AF</td>
<td>405</td>
</tr>
<tr>
<td>area</td>
<td>403</td>
</tr>
<tr>
<td>BV</td>
<td>404</td>
</tr>
<tr>
<td>VAR</td>
<td>404</td>
</tr>
<tr>
<td>VRB</td>
<td>404</td>
</tr>
<tr>
<td>CJ</td>
<td>404</td>
</tr>
<tr>
<td>CJA</td>
<td>404</td>
</tr>
<tr>
<td>CJ0</td>
<td>404</td>
</tr>
<tr>
<td>CJ</td>
<td>404</td>
</tr>
<tr>
<td>CJA</td>
<td>404</td>
</tr>
<tr>
<td>CJSW</td>
<td>405</td>
</tr>
<tr>
<td>CJSW</td>
<td>405</td>
</tr>
<tr>
<td>CJSW</td>
<td>405</td>
</tr>
<tr>
<td>CTA</td>
<td>406</td>
</tr>
<tr>
<td>CTC</td>
<td>406</td>
</tr>
</tbody>
</table>
TT, 405
TTT1, 406
TTT2, 406
VAR | VRB | BV | VB, 404
VB | VAR | VRB | BV, 404
VJ | PHA | PB | PHI, 405
VJSW | PHP, 405
VRB | BV | VB | VAR, 404
VSB, 405
W, 404, 406
WM, 187, 404
WP, 187, 404
XM, 404
XOM, 405
XOI, 405
XTI, 406
XW, 404, 406
JFET
beta, 417
cgd, 417
cgs, 417
fc, 417
is, 417
lambda, 417
pb, 417
rd, 417
rs, 417
vto, 417
listing all supported, 366
MESFET
acm, 422
af, 421
align, 422
alph, 420
AREAeff, 422
b, 420
beta, 420
BETAeff, 423
bex, 423
capop, 420
cgd, 421
CGDeff, 423
cgs, 421
CGSeff, 423
crat, 421
ctd, 423
cuts, 423
d, 420
dcap, 420
eg, 423
fc, 421
gamds | gamma, 420
gamma | gamds, 420
gap1, 423
gap2, 423
gcap, 421
gdsnoi, 421
hdif, 422
interr, 421
is, 420
ISeff, 423
k1, 420
kf, 421
l, 422
lambda, 420
Idel, 422
Idif, 422
level, 420
m, 421
n, 423
nchan, 420
niev, 420
pb, 421
rd, 420
RDeff, 422
rg, 420
RGeff, 422
rs, 421
RSeff, 422
rsh, 421
rshg, 421
rshl, 421
sat, 420
satexp, 421
tcv, 423
tlev, 420
tlevc, 420
tpb, 423
trd, 423
trl, 423
trs, 423
tt, 421
ucrit, 421
vbi, 421
vdel, 421
vgexp, 421
vmax, 421
vp, 421
vto, 421
w, 422
wdel, 422
xti, 423
MOSFET (all levels)
acm, 492
cj | cjg, 492
cjgate, 492
cjm | cj, 492
cjsw | cjw, 492
cjw | cjsw, 492
dlat | latd | ld, 492
expli, 492
geo, 195
hdif, 492
ijs | js, 492
is, 492
js | ijs, 492
jssw | jsw, 492
jsw | jssw, 492
latd | ld | diat, 492
ld | diat | latd, 492
ldif, 492
lmax, 492
lmin, 492
meto, 492
mj | mj0, 492
mj0 | mj, 492
mjsw | mjw, 492
mjw | mjsw, 492
n, 492
nds, 493
pb | pj, 493
pbsw | pjw | php, 493
php | pbsw | pjw, 493
pj | pb, 493
pjw | php | pbsw, 493
prdt, 493
prst, 493
rd, 493
rdc, 493
rs, 493
rsc, 493
rsh | rshm, 493
rshm | rsh, 493
vnds, 493
wmax, 493
wmin, 493
wmilt, 493
xl, 493
xle, 493
xref, 493
xwref, 493
mosfet levels 1/2/3

del, 431
delta, 430
dfs | nfs | dnf | nf, 430
dl | idel | xl, 430
dnb | nb | nsb, 430
dnf | nf | dfs | nfs, 430
dw | wdel | xw, 430
eta, 431
gamma, 430
kp, 430
lambda, 430
ld, 430
idel | xl | dl, 430
level, 430

mosfet level 4 (bsim1)
alpha, 476
cgbo, 450
cgdo, 449
cgso, 449
dl, 448
dl | idel | xl, 450
dw, 448
dw | wdel | xw, 450
eta, 447
gamma, 476
iratio, 476
k1, 447
k2, 447
kt1, 475
lalpha, 476
lbeta, 476
lomega, 476
lomega, 476
lalpha, 476
lbeta, 476
lomega, 476
lbeta, 476
lx2mz, 448
lx2u0, 448
lx2u1, 449
lx3e, 448
lx3ms, 449
lx3u1, 449
mus, 449
muz, 447
n0, 450
nb, 450
nd, 450
palph, 476
peta, 447
pgamma, 476
phi, 447
piratio, 476
pk1, 447
pk2, 447
pkt1, 475
pmus, 449
pmuz, 448
pn0, 450
pnb, 450
pnd, 450
pphi, 447
pu0, 448
pu1, 448
pute, 475
pvfb, 447
pwoffset, 476
px2e, 448
px2ms, 449
px2mz, 448
px2u0, 448
px2u1, 449
px3e, 448
px3ms, 449
px3u1, 449
satmod, 476
submod, 476
temp, 449
tempmod, 475
tox, 449
u0, 448
u1, 448
ute, 475
vdd, 449
vfb, 447
voffset, 476
walph, 476
wdel | xw | dw, 450
weta, 447
wgamma, 476
wiratio, 476
wk1, 447
wk2, 447
wk1, 475
wmlt, 476
wmus, 449
wmuz, 447
wn0, 450
wnb, 450
wnd, 450
wp, 447
wu0, 448
wu1, 448
wute, 475
wvfb, 447
wwoffset, 476
wx2e, 448
wx2ms, 449
wx2mz, 448
wx2u0, 448
wx2u1, 449
wx3e, 448
wx3ms, 449
wx3u1, 449
x2e, 448
x2ms, 449
x2mz, 448
x2u0, 448
x2u1, 448
x3e, 448
x3ms, 449
x3u1, 449
xj, 476
xl | ldel | dl, 450
xpart, 450
xw | dw | wdel, 450
MOSFET Level 5
del, 451
dl | ldel | xl, 451
dw | wdel | xw, 451
eghalf, 451
ld, 451
ldel | xl | dl, 451
mu0, 451
nsub, 451
qdtol, 451
qstol, 451
solver, 451
tox, 451
tref, 451
vfb, 451
vmax, 451
wd, 451
wdel | xw | dw, 451
xj, 451
xl | dl | ldel, 451
xw | dw | wdel, 451
MOSFET Level 14/54
acnqsmod, 465
Index

ad, 488
aebcp, 488
agbcp, 488
bjtoff, 488
cth0, 488
frbody, 488
geomod, 465
ic, 489
l, 489
min, 465
nbc, 489
nf, 465
nrb, 489
nrs, 489
nseg, 489
off, 489
pd, 489
pdbcp, 489
ps, 489
psbcp, 489
RBDB, 465
rbbodymod, 465
RBPD, 465
RBPS, 465
RBSB, 465
rganomod, 489
rggeomod, 465
rsh, 489
rth0, 489
SA, 465
sb, 466
sd, 466
soimod, 489
tradeout, 489
trnsgmod, 465
vbsusr, 489
w, 489

MOSFET Level 15 (RPI a-Si TFT)
ALPHASAT, 467
CGDO, 467
CGSO, 468

MOSFET Level 47
a0, 483
a1, 483
a2, 483
at, 483
bulkmod, 483
cdsc, 483
cdscb, 483
cgbo | cgbo, 485
cgbom | cgbo, 485
cgdo | cgdo, 485
cgdom | cgdo, 485
cgso | cgso, 485
cgsom | cgso, 485
cit, 483
dl | ld, 485
dl | ldlel | xl, 485
drout, 484
dsub, 484
dvt0, 484
dvt1, 484
dvt2, 484
dw | wd, 485
dw | wdel | xw, 485
em, 484
eta, 485
eta0, 485
etab, 485
gamma1, 484
gamma2, 484
k1, 484
k2, 484
k3, 484
k3b, 484
keta, 483
kt1, 484
kt1l, 484
kt2, 484
ld | dl, 485
ldd, 485
ldel | xl | dl, 485
litt, 484
lmit, 485
mobmod, 483
nch | npeak, 484
nfactor, 483
ngate, 484
nlx, 484
npeak | nch, 484
nsub, 483
pclm, 485
pdibl1, 485
pdibl2, 485
phi, 484
pscbe1, 485
pscbe2, 485
pvag, 485
rdso, 485
rdsw, 485
satmod, 483
subthmod, 483
tnom | tref, 485
tox, 483
tref | tnom, 485
u0, 485
ua, 484
ua1, 484
ub, 484
ub1, 484
uc, 484
uc0, 484
dvt0, 454
dvt0w, 454
dvt1, 454
dvt1w, 454
dvt2, 454
dvt2w, 454
dw | wdel | xw, 458
dwb, 457
dwc, 457
dwg, 457
EG, 458
elm, 460
em, 459
eta0, 454
etabl, 454
gamma1, 460
gamma2, 460
GAP1, 458
GAP2, 458
is, 454
k1, 454
k2, 454
k3, 454
k3b, 454
keta, 454
kt1, 459
kt1l, 459
kt2, 459
ldel | xl | dl, 458
lint, 457
ll, 457
lle, 457
lln, 457
lw, 457
lwc, 458
lw1, 458
lwlc, 458
lwn, 458
mobmod, 453
nch | npeak, 455
nfactor, 455
ngate, 455
nlx, 455
npeak | nch, 455
nqsmod, 453
nsub, 455
pclm, 455
pdiblc1, 455
pdiblec2, 455
pdiblecb, 455
prt, 459
prwb, 455
prwg, 455
pscbe1, 455
pscbe2, 455
PTA, 459
PTP, 459
pvag, 455
rdsw, 455
tlev, 459
tlevc, 459
tnom | tref, 459
tox, 455
tref | tnom, 459
TRS, 459
u0, 455
ua, 456
ua1, 459
ub, 456
ub1, 459
uc, 456
uc1, 459
ute, 459
vbm, 456
vbx, 460
version, 453
vfb, 456
vfbcv, 456
vfbsflag, 456
voff, 456
voffcv, 456
vsat, 456
vth0 | vtho, 456
vtho | vth0, 456
w0, 456
wdel | xw | dw, 458
wint, 458
wl, 458
wlw, 458
wlw, 458
wlw, 458
ww, 458
wwc, 458
wwl, 458
wwl, 458
wwl, 458
wwn, 458
xj, 456
xl | dl | Idel, 458
xpart, 457
xt, 460
XTI, 459
xw | dw | wdel, 458
resistor
bulk, 498
cap, 498
capsw, 498
cox, 498
cratio, 498
di, 498
dlr, 498
dw, 498
l, 498
level, 498
noise, 498
rac, 498
res, 498
rsh, 498
shrink, 498
tc1c, 498
tc1r, 498
tc2c, 498
tc2r, 499
thick, 499
tref, 499
vce, 499
vce, 499
w, 499
switch
di, 502
dv, 502
dht, 502
ih, 502
it, 502
roff, 502
ron, 502
vh, 502
vt, 502
unspecifed, 118
model selectors
BJT, 382
models
small signal, 36
models supported by T-Spice, 365
modes
See options
modparam keyword, 159
modulation index, 180
monitoring simulations, 55
See also Simulation Manager
Monte Carlo Analysis, 522
output, 523
statistics, 524
Monte Carlo analysis, 131
defined, 38
example, 167
number of runs, specifying, 159
parameters, 132, 133
specifying with .step, 160
monte keyword, 158, 159, 160, 522, 523
MOSFET
cconductance, in DC analysis, 34
device parameters, 193
device statements, 193–196
effective areas and perimeters, 495–497
equations, additional, 494–497
level/model cross reference, 368–369
noise types, 141
N-type model specification, 116
parasitic diode equations, 494–495
parasitic resistance equations, 494
P-type model specification, 116
stacked devices, 195
switch-level example, 215
terminals, notation for, 138

MOSFET Level 1/2/3
capacitance, 432–433
charge, 432
current equations
  Level 1, 432
device model, 430–446
large signal model, 431
Level 1 equations, 431–434
Level 2 equations, 434–443
Level 3 equations, 443–446
linear and saturation regions, 436–438
model parameters, 430
second-order effects
  Level 2, 439
subthreshold region
  Level 2, 434–436
temperature dependence, 445–446
threshold voltage, 433–434
Ward-Dutton charge model, 439–443
MOSFET Level 4/13 (BSIM1)
device models, 447–450
model parameters, 447–450
MOSFET Level 5
characteristics, 451–452
device models, 451–452
model parameters, 451
MOSFET Level 100 (Penn State & Philips PSP Model)
device model, 491
model documentation, 491
model parameters, 491
MOSFET Level 11/63 (Philips MOS 11)
device model, 463–464
model documentation, 463
model parameters, 463–464
MOSFET Level 14/54 (BSIM4)
device models, 465–466
model parameters, 465–??
MOSFET Level 15
equivalent circuit, 469
MOSFET Level 15/61 (RPI a-SI)
model parameters, 467
MOSFET Level 16
equivalent circuit, 473
MOSFET Level 16/62 (RPI Poly-Si)
model parameters, 470
MOSFET Level 16/62 (RPI Poly-Si) v3.0
model parameters, 472
MOSFET Level 20 (Philips MOS 20)
device model, 474
model documentation, 474
model parameters, 474
MOSFET Level 28 (Extended BSIM1)
device models, 475–478
equations, 476–478
model parameters, 475–476
process parameters, equations for, 476–478
subthreshold current, 478
MOSFET Level 30 (Philips MOS 30)
device model, 479
model documentation, 479
model parameters, 479
MOSFET Level 31 (Philips MOS 31)
device model, 480
model documentation, 480
model parameters, 480
MOSFET Level 40 (Philips MOS 40)
device model, 481
model documentation, 481
model parameters, 481
MOSFET Level 44/55
model parameters, 482
MOSFET Level 44/55 (EKV)
device models, 482
model documentation, 482
model parameters, 482
MOSFET Level 47, 483
device models, 483–487
drain current, 486–487
equations, 486–487
model parameters, 483–485
MOSFET Level 49/53
device models, 453–461
drain current, 460–461
equations, 460–461
gate charge, 461
model parameters, 453–460
  AC and capacitance, 457
  basic, 453–456
  bin description, 459
  length and width, 457–458
  model selectors, 453
  nonquasi static, 460
  process parameters, 459
  temperature, 458–459
MOSFET Level 57 (BSIM3SOI)
device models, 488
model documentation, 488
model parameters, 488
MOSFET Level 70 (BSIM4SOI)
device models, 488
model documentation, 488
MOSFET Level 70 (BSIM4SOI) model parameters, 488
MOSFET Level 9/50 (Philips MOS 9)
device model, 462
model documentation, 462
model parameters, 462
MOSFETs (levels 1-3 and BSIM)
parameters, additional, 492–??, 493, ??–493
multidimensional parameter space, 39
multidimensional trends, 38
multiline expressions, 69, 70
multiple
  convergence solutions
    use of .nodeset for, 119
document interface, 18
minima, 112
optimization parameters, assigning, 133
optimizations, sequential, 135
parameters, assigning, 133
simulation runs, with .alter, 80
  simulations
    See batch simulations
subcircuits, 162
multiplication, 70, 71
multiplicity
  CCVS device statement, 185
diode device statement, 187
  inductors, 188
JFETs, 191
MESFETs, 192
MOSFETs, 194
  parameter, 172, 174
parasitic diodes, 195
resistors, 199
subcircuits, 190
VCCS device statement, 211, 217
multiplier
  in bit pattern, 181
  factors
    example using, 182
  noise source (resistor), 199
multi-threaded processing, 40
multi-threaded processing, 41
mutual inductor
  device statement, 197

N

n
  metric prefix, 69
  argument, .print, 138
names, 66–67
  device names, 67
global, 67, 97
  hierarchical, 67
inconsistencies in, 39
internal, 67
library sections, 104
node, 66
  of parameters, 70
  reserved, 66
in subcircuits, 162
NAND gate, 225
nano-, 69
natural logarithm, 72
negation, 70, 71
negative measurement results, 111
  nested
    .include commands, 103
    subcircuits, 67, 162
    sweeps
      Monte Carlo analysis in, 160
      optimization and, 160
netlists
  color-coding in, ??–44
  editing mode, 44
  ending, 91
  exporting and extracting, 42
  inserting commands in, 20
  opening, 55
  searching, 45
text editors for, 42
  updating, 27
Newton iterations, 34
nfreqs keyword, 95
njf keyword, 116
nl device parameter
  transmission line, 203
nmf keyword, 116
nmos keyword, 116
nodes
  floating, 100
global, 97
  initial guesses, for DC operating point, 37
  naming syntax
    See names, 66
  order of, in subcircuit calls, 190
  referencing, in subcircuits, 67, 100, 162
  reserved names, 66
  residual current tolerances, 36
  total in simulation, displayed, 57
  in vectors, 181
  voltages, setting for DC operating point calculation, 100
  .nodeset command
    .op, using with, 121
    in .alter blocks, 80
    convergence, improving with, 37
    defined, 119
    in subcircuits, 162
noise, 38
  analysis, 120
  results, printing, 138, 152
models, 120
noise types
  FN, 140, 141
IB, 140
IBE, 140
IBEFN, 140
IBEP, 140
IBEPFN, 140
IC, 140
ICCP, 140
ID, 141
ITZF, 140
RB, 140
RBI, 140
RBP, 140
RBX, 140
RC, 140
RCI, 140
RCX, 141
RD, 141
RE, 140, 141
RG, 141
RS, 141
RX, 140, 141
TOT, 140, 141

parameters
  diode, 405
  MESFET, 421
  spectral density, printing, 139
types of, 140
noise
device parameter
  resistor, 198
  keyword, 138
.noise command
  defined, 120
  in subcircuits, 162
noise spectral density, 120, 139
nominal temperature, 198
nonconvergence
  avoiding with .nodeset, 37
  errors
    transient analysis, 35
    powerup simulation option after, 37
    triggering gmindc stepping, 34, 236, 238, 244
nonconvergences, 33, 37
  DC operating point calculation, 37
  DC transfer analysis, 37
nonlinear
  capacitor, 214
  device models
    solving iteratively, 33
  systems
    discretization error in, 35
nonlinear functions
  See polynomials
noprint keyword, 121
NOR gate, 225
normal distribution, 132

normal region of operation, 382
notation
  numbers and units, 69
  subcircuits (hierarchical), 100
nnp keyword, 116
npnpoints keyword, 95
nrd device parameter, MOSFET, 193, 465, 488
nrs device parameter, MOSFET, 193, 465, 488
N-type MOSFET, specifying for .model, 116
  number of iterations
    DC analysis, 33, 37
    source stepping, 37
    time step size and, 34
  numbers, T-Spice syntax for, 69
  numerical derivatives
    increment for, 117
  numerical format, 109
numnd option
  compared with numnt, 34
  DC analysis and, 33
  simulation concepts of, 37
numnt option
  simulation concepts of, 37
  transient analysis and, 34
numntreduce option
  simulation concepts of, 37
  transient analysis and, 34

O
o suffix, octal notation, 181
oct keyword, 76–77
octal notation, 181
octave sweeps
  .step command, 160
  AC analysis, 76–77
off keyword
  .measure command, 109–110
  vectorized waveform, 181
on keyword, 181
onnoise keyword, 139
.op command
  .acmodel command with, 60, 61, 79
    defined, 121
op keyword, 166
open circuits, 33
opening
  simulation netlist, 55
  simulation output file, 55
operational amplifier, 222
operator precedence, 70
opt keyword
  .model command, 116
  .param command, 133
.step command, 158
.optgoal command
  defined, 122
  example with .optimize, 124
  examples, 122
  vs. .measure, 109
  optimization and, 123
  syntax, 122

optimization
  AC analysis, 123
  algorithm, 116, 526
  algorithm, specifying, 116
  described, 525
  example, 161
  formulae, 122
  goal, 525, 526
  goal, defining
    in .measure command, 110
    in .step command, 160
    globally, with .optgoal command, 122
  HSPICE-compatible, 535
  incremental, 526
  invoking, 123, 526
  iterations, maximum number of, 117
  .measure command for, 109
  method, selecting (.model command), 118
  for model parameter extraction, 88
  multiple (sequential) runs of, 135
  .optgoal vs. .measure, 109
  parameters, 116, 526
    initializing with .paramlimits command, 135
    .param command, 131, 133
    ranges, defining with .paramlimits command, 135
    step sizes for, .model, 117
    variation of, for convergence, 118
  reference data for, incorporating, 87
  sweep, specifying with .step, 160
  transient analysis, 124

Optimization Wizard, ??–534
.optimize command, 123–124, 525, 526
.optimize keyword, 158, 160
.options command, 125
  absi | abstol, 36
  fields
    accurate, 126
    autostop, 109
    nummd | itl1, 37
  options
    general definition of, 33
    T-Spice syntax for, 75
    waveform display during simulation, 52

Options >
  Font, 30
.options command
  absdv, 263
  kcltest, 246
  absi, 229
  absq, 264
  abstol, 229
  absv, 230
  absvar, 263
  acct, 321
  accuracy and convergence, 227
  accurate, 231
  acout, 322
  aldecdir, 356
  amscfg, 357, 358
  autostop, 308
  autoconverge, 232
  brief, 323
  bypass, 233
  bytol, 234
  casesensitive, 309
  compat, 310
  conncheck, 311
  csdf, 325
  cshunt, 235
dcap, 283
dccap, 284
dchomotopy, 236
dcmethod, 238
dcstep, 239
defad, 285
defas, 286
defl, 287
defard, 288
defrs, 289
defpd, 290
defps, 291
defw, 292
deriv, 293
dnout, 326
echo, 327
expert, 328
extraiter, 240
fast, 241
ft, 266
general options, 306
gmin, 242
gminde, 243
gramp, 244
gshunt, 245
imax, 274
ingold, 329
itl3, 275
itl4, 274
kcltest, 246
kvltest, 247
linear solver options, 302
linearsolver, 304
<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>rmax</td>
<td>279</td>
</tr>
<tr>
<td>rmin</td>
<td>272</td>
</tr>
<tr>
<td>scale</td>
<td>299</td>
</tr>
<tr>
<td>scalm</td>
<td>300</td>
</tr>
<tr>
<td>search</td>
<td>316</td>
</tr>
<tr>
<td>spice</td>
<td>317</td>
</tr>
<tr>
<td>statdelay</td>
<td>345</td>
</tr>
<tr>
<td>str</td>
<td>318</td>
</tr>
<tr>
<td>threads</td>
<td>41, 319</td>
</tr>
<tr>
<td>tnom</td>
<td>301</td>
</tr>
<tr>
<td>trextraiter</td>
<td>280</td>
</tr>
<tr>
<td>trnewtol</td>
<td>280</td>
</tr>
<tr>
<td>trtol</td>
<td>281</td>
</tr>
<tr>
<td>vacasemode</td>
<td>359</td>
</tr>
<tr>
<td>vadatabase</td>
<td>360</td>
</tr>
<tr>
<td>vaexprtol</td>
<td>361</td>
</tr>
<tr>
<td>vasearch</td>
<td>362</td>
</tr>
<tr>
<td>vatimetol</td>
<td>363</td>
</tr>
<tr>
<td>vaverbose</td>
<td>364</td>
</tr>
<tr>
<td>verbose</td>
<td>346</td>
</tr>
<tr>
<td>vmax</td>
<td>261</td>
</tr>
<tr>
<td>vntol</td>
<td>230</td>
</tr>
<tr>
<td>wl</td>
<td>302</td>
</tr>
<tr>
<td>xmu</td>
<td>273</td>
</tr>
<tr>
<td>xref</td>
<td>347</td>
</tr>
<tr>
<td>zpivtol</td>
<td>306</td>
</tr>
<tr>
<td>OR gate</td>
<td>225</td>
</tr>
<tr>
<td>order</td>
<td></td>
</tr>
<tr>
<td>of integration</td>
<td>34, 35</td>
</tr>
<tr>
<td>of magnitude</td>
<td>69</td>
</tr>
<tr>
<td>oscillator, voltage-controlled</td>
<td>226</td>
</tr>
<tr>
<td>output</td>
<td></td>
</tr>
<tr>
<td>.alter</td>
<td>81</td>
</tr>
<tr>
<td>files</td>
<td></td>
</tr>
<tr>
<td>.ac command</td>
<td>76</td>
</tr>
<tr>
<td>displaying</td>
<td>19, 54</td>
</tr>
<tr>
<td>opening</td>
<td>55</td>
</tr>
<tr>
<td>parameter sweeps</td>
<td></td>
</tr>
<tr>
<td>format of</td>
<td>158</td>
</tr>
<tr>
<td>of Fourier analysis</td>
<td>96</td>
</tr>
<tr>
<td>messages</td>
<td></td>
</tr>
<tr>
<td>See Simulation Output window</td>
<td></td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>522, 523–525</td>
</tr>
<tr>
<td>noise</td>
<td></td>
</tr>
<tr>
<td>total, reporting</td>
<td>140</td>
</tr>
<tr>
<td>operating points</td>
<td>79</td>
</tr>
<tr>
<td>signal</td>
<td></td>
</tr>
<tr>
<td>statistics measurements</td>
<td>112</td>
</tr>
<tr>
<td>small-signal data</td>
<td>121</td>
</tr>
<tr>
<td>small-signal parameters</td>
<td>79</td>
</tr>
<tr>
<td>suppressing from</td>
<td></td>
</tr>
<tr>
<td>.measure</td>
<td>110</td>
</tr>
<tr>
<td>sweep</td>
<td>520–521</td>
</tr>
<tr>
<td>trigger/target</td>
<td>111</td>
</tr>
<tr>
<td>measurements</td>
<td></td>
</tr>
<tr>
<td>comparing</td>
<td>114</td>
</tr>
<tr>
<td>Fourier analysis of</td>
<td>95</td>
</tr>
<tr>
<td>viewing</td>
<td>55</td>
</tr>
<tr>
<td>list</td>
<td>330</td>
</tr>
<tr>
<td>lvltim</td>
<td>267</td>
</tr>
<tr>
<td>macmod</td>
<td>312</td>
</tr>
<tr>
<td>maxdefailures</td>
<td>248</td>
</tr>
<tr>
<td>maxmsg</td>
<td>331</td>
</tr>
<tr>
<td>maxord</td>
<td>270</td>
</tr>
<tr>
<td>measform</td>
<td>332</td>
</tr>
<tr>
<td>measinfo</td>
<td>333</td>
</tr>
<tr>
<td>method</td>
<td>271</td>
</tr>
<tr>
<td>mindcratio</td>
<td>249</td>
</tr>
<tr>
<td>minresistance</td>
<td>294</td>
</tr>
<tr>
<td>minsrcestep</td>
<td>250</td>
</tr>
<tr>
<td>mintimeratio</td>
<td>272</td>
</tr>
<tr>
<td>model evaluation</td>
<td>281</td>
</tr>
<tr>
<td>options</td>
<td></td>
</tr>
<tr>
<td>modmonte</td>
<td>295</td>
</tr>
<tr>
<td>monteinfo</td>
<td>334</td>
</tr>
<tr>
<td>moscap</td>
<td>296</td>
</tr>
<tr>
<td>mout</td>
<td>297, 298</td>
</tr>
<tr>
<td>mu</td>
<td>273</td>
</tr>
<tr>
<td>newtol</td>
<td>240</td>
</tr>
<tr>
<td>node</td>
<td>335</td>
</tr>
<tr>
<td>nomod</td>
<td>336</td>
</tr>
<tr>
<td>numdgt</td>
<td>337</td>
</tr>
<tr>
<td>numnd</td>
<td>251, 252</td>
</tr>
<tr>
<td>numns</td>
<td>253</td>
</tr>
<tr>
<td>numnt</td>
<td>274</td>
</tr>
<tr>
<td>numntreduce</td>
<td>275</td>
</tr>
<tr>
<td>numnx</td>
<td>254</td>
</tr>
<tr>
<td>numnxramp</td>
<td>255</td>
</tr>
<tr>
<td>opts</td>
<td>338</td>
</tr>
<tr>
<td>output options</td>
<td>317</td>
</tr>
<tr>
<td>outputall</td>
<td>339</td>
</tr>
<tr>
<td>parhier</td>
<td>313</td>
</tr>
<tr>
<td>pathnum</td>
<td>340</td>
</tr>
<tr>
<td>persist</td>
<td>315</td>
</tr>
<tr>
<td>pivtol</td>
<td>305</td>
</tr>
<tr>
<td>poweruplen</td>
<td>276</td>
</tr>
<tr>
<td>precise</td>
<td>256</td>
</tr>
<tr>
<td>probei</td>
<td>349</td>
</tr>
<tr>
<td>probej</td>
<td>350</td>
</tr>
<tr>
<td>probevl</td>
<td>341</td>
</tr>
<tr>
<td>probemonte</td>
<td>351, 353</td>
</tr>
<tr>
<td>probeq</td>
<td>352</td>
</tr>
<tr>
<td>probetop</td>
<td>342</td>
</tr>
<tr>
<td>probev</td>
<td>354</td>
</tr>
<tr>
<td>prtdel</td>
<td>343</td>
</tr>
<tr>
<td>prtinterp</td>
<td>344</td>
</tr>
<tr>
<td>recycleOP</td>
<td>258</td>
</tr>
<tr>
<td>relchgtol</td>
<td>278</td>
</tr>
<tr>
<td>reldv</td>
<td>277</td>
</tr>
<tr>
<td>reli</td>
<td>259</td>
</tr>
<tr>
<td>relq</td>
<td>278</td>
</tr>
<tr>
<td>reltol</td>
<td>259</td>
</tr>
<tr>
<td>relv</td>
<td>260</td>
</tr>
<tr>
<td>relvar</td>
<td>277</td>
</tr>
<tr>
<td>resmin</td>
<td>294</td>
</tr>
<tr>
<td>order</td>
<td></td>
</tr>
<tr>
<td>of integration</td>
<td>34, 35</td>
</tr>
<tr>
<td>of magnitude</td>
<td>69</td>
</tr>
</tbody>
</table>
Index

viewing in W-Edit, 55
output curves, comparing with external data, 87
output files
  opening, 518
  size of, 34
output keyword, 112
output noise
  spectral density, reporting, 139
output signal
  trigger/target measurements, 111
outvar, 112
outvar2, 113
overriding operator precedence, 70
overwrite mode, 44

P
p metric prefix, 69
p(d), 138
parallel devices
  BJT, 172
  capacitors, 174
  CCVS, 185
  diodes, 187
  JFETs, 191
  MESFETs, 192
  MOSFETs, 194
  resistors, 199
  VCCS, 211, 217
  inductors, 188
  plate capacitor, 214
  subcircuits, 190
parallel processing, 40
.param command, 520, 522, 523, 526
  examples, 133
  expression evaluation and, 114
  in library files, 105
.model and, 117
  in subcircuits, 162
  sweeps and, 159
  syntax, 131–133
param keyword, 114, 159
parameters
  sweeping
    with .measure command, 109
parameter
  sweeping
    for Monte Carlo analysis, 159
parameters
  declaring with .param command, 131–133
  device models, specifying, 116
  expressions involving, 114
  extracting from models, 87
  Monte Carlo, 132–133
  naming, 70
  optimization, 116–118, 133, 525, 526
  overriding subcircuit/external model values, 190
  relative variation, 118
  signal statistics measurements, 111
  small-signal, 76, 121
  subcircuit, specifying, 162
  sweeping, 519
    concepts of, 38
    with .data command, 87
    for optimization, 160
    with .step command, 158
    temperature, 159
  trigger/target measurements, 110
  types of, 518
  unspecified, 118
  See also keyword
parameters supported for each model, 366
parametric analysis, 38
parametric sweeps ., 158
.paramlimits command, 526
  defined, 135
  example with .optimize, 124
  examples, 135
  optimization and, 123
  syntax, 135
parasitic
  capacitance, BJT Level 1 (Gummel-Poon), 375
  diodes, 194–196
    MOSFET, 494–495
  resistance, 188–189
  resistance equations, MOSFET, 494
  resistors, BJT, 375
parentheses, 70, 71
parmin keyword, 118
parsing SPICE versions, 66
Paused, simulation status, 54
.pd device parameter, MOSFET, 193, 465, 488
peak-to-peak values, 111
performance measures, optimizing, 39, 160
Performs, 158
perimeter
  effective, 194, 495–497
  MOSFET
    effective, 495
permittivity
  See dielectric permittivity
phase
  complex current, 139
  discontinuities, VCO model, 226
  voltage, 139
phase advance, sinusoidal waveform, 181
phase differences, calculation of, 322
Philips models
model cross reference, 369–371
physical constants, 372
pi, 372
pico-, 69
piecewise linear waveform
  current source, 179
  voltage source, 206
pin names
  aliasing, 67
  pi-type lumps, 177, 204
pjf keyword, 116
pkt1, 451
plots
  temperature vs. voltage, 163
  voltage vs. temperature, 163
plus signs, in multiline expressions, 70
pmf keyword, 116
pmos keyword, 116
P-N diode
  model type specification, 116
pnf keyword, 116
poi keyword, 76–77, 158–160
POLY capacitor parameter, 174
POLY voltage-controlled current source parameter, 210, 217
POLY voltage-controlled voltage source parameter, 220
polynomials
  coefficients for, 185
  formulas for, 184
  keyword for, 183
  replacing corners with, 179
polysilicon capacitance, 409
positive current, defined
  mutual inductor, 197
  voltage or current source, 183
pow(x,y), 72
.power command, 136
power consumption
  reporting, 138
power dissipation, computing, 136
powers of variables, 72
powerup keyword, 166
powerup simulation, defined, 37
poweruplen option, 37
pp keyword, 112
precision
  of computations, 36
prefixes, metric, 69
preview keyword, 166
.print command, 137–142
  AC magnitude/phase differences, 322
  defined, 137
  device states, 137
  error function measurements, 114
  expression evaluation and, 114
  find-when measurements, 113
noise mode
  arguments, 139
  error function measurements, 114
  signal statistics measurements, 112
  in subcircuits, 162
  trigger/target measurements, 111
  wildcards, 138
.print noise command
  signal statistics measurements, 112
  probabilistic parameter definitions, 38
  probability distributions, 131, 522, 523
  Monte Carlo analysis, 160
.probe command
  AC magnitude/phase differences, 322
  defined, 152–153
  in subcircuits, 162
  progress tracking, 56, 57
  progress, of simulation, 57
  progress, simulation
display options, 52
.protect command, 154
described, 59
.visible command
described, 60
ps device parameter
  MOSFET, 193, 465, 488
PSPICE compatibility, 310
P-Scpe compatible syntax, list sweeps, 160
P-type MOSFET, specifying for .model, 116
pulse "off" time
terminated current source, 181
pulse "on" time
terminated current source, 181
pulse keyword
  current source, 179
pulse period
  current source, 179
pulse waveform
  current source, 179
  voltage source, 205
pulse width
  current source, 179
  terminated current source, 181
pw keyword, 181
pwr(x,y), 72
Queued, simulation status, 54
quotation marks
  in expressions, 70
  filenames in, 90, 104
qz(d), 138
r
  device parameter
    inductor, 188
    resistor, 198
    transmission line, 203
  key letter, 198
r keyword, 116
random
generator, 160
  limit distributions, 132
number sequence, initializing, 159
  parameter variation, 38
random(x), 72
ranges, of optimization parameters, 135
real components, printing
  complex current, 139
  complex voltage, 139
real-time waveform viewing, 52
Redo, 45
reference data, 87
regions of operation
  BJT Level 1 (Gummel-Poon), 382
regular expressions
  rules, 47–48
  searching, 46, 47
  Unix-style, 46, 47
relative
  error, in optimization measurement, 110
  simulation accuracy, 37
  tolerances
    charge, 37
    numerical approximations, 36
  voltages, reporting, 138
relative input parameter variation, 118
relchargetol option
  discretization error and, 34
  simulation concepts of, 37–38
relin keyword, 117, 118
reloading files, 27
relout keyword, 117, 118
reltol option
  DC analysis, 33
  vs. relchargetol, 37
  simulation concepts of, 36
  time step size and, 38
  transient analysis, 34
  remainder expression, 72
removing files from simulation queue, 55
REPEAT keyword, 179
repeating a simulation, 80
repeating bit input
  example using, 182
replacing statements with .alter, 80
replacing text, 45, 46
reserved names, 66
residual currents, 36, 38
resistance
  during AC analysis, 199
  equations, resistor device model, 499
  parasitic
    See parasitic resistance
    scaling, 199
    specifying, 198
    switch, 201
temperature effects on
  diode, 414
temperature influence on, 198
  transmission line, distributed, 203
  variable base resistance equations, 384
resistor
  AC analysis, 199
  capacitance equations, 500–501
  device model, 498–501
  device statements, 198–200
  equations, 499–501
  large-signal model, 499
  model parameters, 498–499
  model type specification, 116
  noise, 199
  resistance equations, 499–500
  as switch implementation, 201
resolution
  Fourier analysis, 95
results keyword, 158, 160
results, measurement, 110
Resume simulation processing, 55
rise keyword, 110, 111, 113
rise time
  exponential waveform, 178
  pulse waveform, 179
  trigger/target example, 115
rms keyword, 112
root-mean-square computation, 111
ROUND keyword, 179
rund half-interval
  pulse waveform, 179
PWL waveform, 179
vectorized waveform, 181
rt keyword, 181
Run Simulation dialog, opening, 53
running simulations, 24, 53
Running, simulation status, 54
run-time update, W-Edit, 52

S

s
key letter, 201
saturation current
diode, 194
saturation current
BJT Level 1 (Gummel-Poon), 389–391
diode
temperature effects, 412
saturation current, diode
equation for, 194
saturation region, 382, 436–438
.save command
defined, 155
.savebias command
defined, 156
scale device parameter
BJT, 173
capacitor, 174
inductor, 188
resistor, 198
scaling factor
BJT area, 172
BJT scale, 173
capacitors, 174
inductor, 188
JFET area, 191
MESFET area, 192
resistor, 199
See also geometric and scaling parameters
schematic export tool, 40
Schottky barrier diodes, 415–416
scientific notation, 69
scope
global, 97
subcircuit parameters, 162, 190
search and replace, 45
searching
library files, 105
text, 45
second-order effects
current, 379
MOSFET Level 2, 439
S-Edit, 42
seed keyword, 158, 159
selecting text, 44
selectors, model
See model selectors
sensitive analog designs, 36
sequential optimization, 135
sets of characters, searching for, 47
simulation commands
.ac, 76–78
.acmodel, 79
.assert, 82
.checkpoint, 84
.connect, 86
.data, 87–88, 519
.dc, 89
.del lib, 90
.else, 101
.elseif, 101
.end, 91
.enddata, 92
.endif, 101
.endl, 93
.ends, 94
.eom, 107
.four, 95–96
.global, 97
.hdl, 98
.ic, 100
setup options, 25
keyword groups, 27
.sffm keyword, 180
.sgn(x), 72
short circuits, 33
shot noise, 140
sigma, 132
sign convention, current, 183
sign of x, 72
.sign(x,y), 72
signal statistics measurements, 111–112, 115
silicon
relative dielectric permittivity of, 372
silicon dioxide
relative dielectric permittivity of, 372
Simulate >
Batch Simulations, 19
Simulate >
Run Simulation, 53
simulaton
accuracy, 38
algorithms
See algorithms
batch, 19
commands, entering, 22
output
See output files
queueing, 19
results, reporting, 137
speed, 38
status, display of, 54
time
Fourier analysis, 95
reporting, 139
setup options, 25
keyword groups, 27
.sffm keyword, 180
.sgn(x), 72
short circuits, 33
shot noise, 140
sigma, 132
sign convention, current, 183
sign of x, 72
.sign(x,y), 72
signal statistics measurements, 111–112, 115
silicon
relative dielectric permittivity of, 372
silicon dioxide
relative dielectric permittivity of, 372
Simulate >
Batch Simulations, 19
Simulate >
Run Simulation, 53
simulaton
accuracy, 38
algorithms
See algorithms
batch, 19
commands, entering, 22
output
See output files
queueing, 19
results, reporting, 137
speed, 38
status, display of, 54
time
Fourier analysis, 95
reporting, 139
simulation commands
.ac, 76–78
.acmodel, 79
.assert, 82
.checkpoint, 84
.connect, 86
.data, 87–88, 519
.dc, 89
.del lib, 90
.else, 101
.elseif, 101
.end, 91
.enddata, 92
.endif, 101
.endl, 93
.ends, 94
.eom, 107
.four, 95–96
.global, 97
.hdl, 98
.ic, 100
setup options, 25
keyword groups, 27
.sffm keyword, 180
.sgn(x), 72
short circuits, 33
shot noise, 140
sigma, 132
sign convention, current, 183
sign of x, 72
.sign(x,y), 72
signal statistics measurements, 111–112, 115
silicon
relative dielectric permittivity of, 372
silicon dioxide
relative dielectric permittivity of, 372
Simulate >
Batch Simulations, 19
Simulate >
Run Simulation, 53
simulaton
accuracy, 38
algorithms
See algorithms
batch, 19
commands, entering, 22
output
See output files
queueing, 19
results, reporting, 137
speed, 38
status, display of, 54
time
Fourier analysis, 95
reporting, 139
simulation commands
.ac, 76–78
.acmodel, 79
.assert, 82
.checkpoint, 84
.connect, 86
.data, 87–88, 519
.dc, 89
.del lib, 90
.else, 101
.elseif, 101
.end, 91
.enddata, 92
.endif, 101
.endl, 93
.ends, 94
.eom, 107
.four, 95–96
.global, 97
.hdl, 98
.ic, 100
Index

.sim, 101
.include, 103
.lib, 104–105
.load, 106
.macro, 107
.malias, 108
.measure, 109–115, 518, 520, 525, 526
.model, 116–118, 525
.nodeset, 119
.noise, 120
.op, 121
.optgoal, 122, 525, 526
.optimize, 123–124, 525, 526
.options, 125
.param, 131–133, 520, 522, 523, 525
.paramlimits, 135, 526
.power, 136
.print, 137–142
.probe, 152–153
.protect, 154
.save, 155
.savebias, 156–157
.step, 158–161, 518, 521
.subckt, 162
.temp, 163
.tf, 164
.tran, 165–167
.unprotect, 154
.vector, 168, ??–169, ??–169
.warn, 169–170

Simulation Manager
See also batch simulations
Simulation Manager, 18, 54, ??–54, ??–54
simulation manager, 16
simulation options
See options command

Simulation Output window
See also output file
Simulation Output window, 19, 54, 55
Simulation Window, 24
simulations
repeated, 80
sin keyword, 180
sin(x), 72
sine functions, 72
sinh(x), 72
sinusoidal waveform, 180
size, of output files, 34
small-signal
analysis, 36
DC transfer function, 164
models, 76
parameters, 76
reporting, 121
small-signal parameters
reporting of, 79
solving circuit equations
See algorithms
source
MOSFET
effective area and perimeter, 194
stepping, 33, 37
step size, 37
sweeping, 159
terminal, notation for, 138
source keyword, 159
sources, voltage
adding for DC simulation, 100
spaces, in filenames, 90
special character codes, 46
specifying device type, 66
specifying subcircuit nodes, 67
speed vs. accuracy, trading off, 38
SPICE
Berkeley implementation, 373, 393, 399
compatibility with, 67, 91
SPICE compatibility, 310
sqrt(x), 72
square root function, 72
.st command, 158
stability, in transient analysis, 35
stacked MOSFET devices, 195
standard deviation, 132
start keyword, 165
statistics, Monte Carlo analysis, 524
status
simulation, 54
Status Bar, 18
status bar, 16
steady state, 33
See also DC analysis
.step command, 158–161, 518, 521
in .alter blocks, 80
defined, 158
optimization and, 123
optimization with .measure, 109
vs. sweep, 77, 519
vs. sweep keyword, 166
step function, 72
step size
parameters, 117
source stepping, 33, 37
time steps, 34
transient analysis, 34, 35, 165
variable, 35
Stop simulation processing, 55
stp() function, 72, 215
strings
as column headings, 138
in vectorized waveforms, 181
subcircuits
accessing nodes and devices, 100
accessing nodes/devices within, 119, 137
defining, 107, 162
ending with .ends, 94
.ic commands in, 100
instantiating, 190–192
nodes, specifying, 67
parallel instances of, 190
parameter values, expressions involving, 114
parameters, scope of, 131
pin names, aliasing, 67
.subckt command
defined, 162
ending with .ends, 94
in library files, 105
.measure commands in, 109
submodel selectors
MESFET, 420
subs, 374
subs, 380, 384
substrate
BJT, 172
terminal, notation for, 138
subthreshold
current, 478
region of operation, 434–436
subtraction, 70, 71
subwaveform
repeating, 179
suffix
d, decimal notation, 181
h, hexadecimal notation, 181
o, octal notation, 181
sw keyword, 116
sw model type, 201
sweep keyword, 38, 77, 87, 158, 159, 166, 519
monte, 522, 523
multiple variables, 519
output, 520–521
sweeps
.data command, 89
linear, 77, 519
list, 519
listed frequencies, 77
logarithmic, 519
logarithmic by decades, 77
logarithmic by octaves, 77
Monte Carlo analysis, 159
multiple variables, 87
optimization, 160
output, 518
parameters, 158
parametric, 158
syntax, 158–160
temperature, 159
swept parameters
difference measurements, 110
switch
current-controlled, 201, 503
device models, 502–503
examples, 503
model parameters, 502–503
table function for, 213
temperature-controlled, 116, 202, 503
switch, voltage- or current-controlled
device statements, 202
switch, voltage-controlled, 201
switch-level MOSFET, 215
symmetric lumps, 204
syntax
color-coding, 44
device names, 66
device statements, 171
documentation conventions, 75
errors, 39, 40
expressions, 70
names, 66
quotation marks, 90, 104
T
t
key letter, 203
metric prefix, 69
tab breaks, searching for, 45, 46
tab character, 47
table() function, 215
tables device parameter
diode, 187
JFET, 191
MESFET, 192
MOSFET, 193, 465, 488
tan(x), 73
tangent function, 73
tanh(x), 73
Tanner Extended BSIM1 MOSFET, 475–476
targ keyword, 110
target string, searching for, 45
tc1 device parameter
capacitor, 174
inductor, 188
resistor, 198
tc2 device parameter
capacitor, 174
inductor, 188
resistor, 198
td device parameter
transmission line, 203
TD keyword, 180
td keyword, 110–111, 113
.temp command
in .alter blocks, 80
defined, 163
**temp** keyword, 159
temperature
  nominal, 198
  specifying, 163
  sweeps, 159, 167
temperature effects
  BJT Level 1 (Gummel-Poon)
    capacitance, 391–392
    energy gap, 389
    saturation current and Beta, 389–391
diode, 411–414
  MESFET, 426–429
  MOSFET Level 1/2/3, 445–446
temperature parameters
  BJT Level 1 (Gummel-Poon), 377
diode, 405, 406
  MESFET, 423
  MOSFET Level 49/53, 458, 459
tera-, 69
terminal
  charge, reporting, 138
  current, reporting, 139
terminals
  BJT, 172
  defined, 33
  JFET, 191
text
  files
    keyboard shortcuts, 44
    searching, 45
    strings, searching for, 45
    wrapping, 68
text editor, 42
  color coding, ??–44
  editing text, 44
  keyword groups, 27
text
  files
    See also input files, netlist, output files
text files
  keyword groups, 27
**.tf** command
  DC operating point and, 121
  defined, 164
  in subcircuits, 162
  The, 158, 373, 393
  thermal noise, 140, 141
  thermal voltage (vt), 194
  three-terminal transistors, 191, 192
  threshold voltage/current, in switch, 201
time
  delay
    in measurements, 111
    measurement example, 115
  in piece-wise linear waveform, 180
  rise time, exponential waveform, 178
  sinusoidal waveform, 180
  vectorized waveform, 181
difference measurement, 110
  resolution
    Fourier analysis, 95
  step size, 34, 35, 165
  excessively small, 38
time(), 205
time-dependent analysis
  See transient analysis
title bar, 16
to keyword, 112, 114
toggled options, 125
tolerances, 36, 40
  **abstol**, 36, 37
  **abstol** option, 36
  charge, 38
  charge tolerance, 37
  **chargetol**, 37
  convergence tolerance for gradient, 117
  DC analysis, 33
  discretization error, 34, 37
  node, 36
  **numnd**, 37
  **numnt**, 37
  **numntreduce**, 37
  **rechargetol**, 37
  **reltol**, 36, 37
toolbars
  Command, 17
  simulation, 18, 56, 57
  total output noise, 140, 141
**.tran** command
  in .alter blocks, 80
  defined, 165–167
  with **.four**, 95
  optimization and, 123
  optimization with **.measure**, 109
**tran** keyword, 109
transconductance, 164, 210, 217
transfer analysis
  See DC analysis
  transfer function, 121, 164
**transfer** keyword, 140
transient analysis, 165
  accuracy of, 37
  command invoking, 165
  power dissipation and, 136
  power usage, 136
  results, 138
  results, printing, 152
  simulation concepts, 34–35
  terminating with **autostop**, 109
transistors, 33
MESFET, 192
MOSFET, 193
three-terminal, 191
transit time
diode, 412
modulation effect in BJT, 380
parameters, 376
transmission delay, 203
transmission line
device models, 504–505
device statements, 203–204
equations, 504, 505
frequency, 203
model with resistors, 198
wavelengths in, 203
transresistance, 140, 141, 164
Trapezoidal Integration Method, 35
trig keyword, 110
trigger/target measurements, 110, 115
trigonometric functions, 72
T-Spice
current conventions, 138
T-type lumps, 177, 204
tutorials
  Optimization, 526–534
type, model, 116

U
u
trig key letter, 177
u metric prefix, 69
UIC keyword, 165
unary negation, 70, 71
unif keyword, 132, 523
uniform distribution, 132, 523
units, specifying in output, 138
Unix-style regular expressions, 46, 47
.unprotect command, 154
described, 60
update model selector, 382–383
updating input files, 27
user interface, 15–20
user-defined functions
declaring with .param, 131–132
examples, 134
syntax, 131

V
v
trig key letter, 205
vacuum
dielectric permittivity of, 372
val keyword, 110, 111
variables
defining, 520
variation in distribution functions, 132
VCO
  See voltage-controlled oscillator
.vector command, 168, 181, 182
vectorized waveform
current source, 181
voltage source, 207
vertical geometry
  BJT, 380
View>
  Simulation Manager, 54
viewing
  simulation netlist, 55
  simulation output file, 55
  simulation output in W-Edit, 55
voltage
  complex, 139
divider, example, 21
gain, 164
magnitude, reporting, 139
reporting, 138
setting for DC operating point calculations, 100
source
  adding for DC simulation, 100
current-controlled, 185
currents, calculating, 121
device statements, 205, 226
initializing, 37
sweeping with .step, 159
supply, 205
voltage vs. temperature
  plotting, 163
voltage-controlled
capacitor, 214
current source, 210
oscillator, 226
resistor, 202
switch, 116
vt (thermal voltage), 194

W
W device parameter
diode, 187
w device parameter
capacitor, 174
MESFET, 192
MOSFET, 193, 465
resistor, 198
Ward-Dutton charge model, 439–443
.warn command
defined, 169–170
warnings, 56, 57
waveforms
  current source, 178–181
  frequency-modulation formula, 180
  sinusoidal formula, 181
  current-controlled voltage source
  polynomial formulas, 186
  polynomial formulas, 184
  repeating, 179
wavelengths, number of
  transmission line, 203
W-Edit
  launching automatically, 54
  W-Edit waveform viewer
    invoking for simulation output, 55
weight keyword, 109, 110, 122
weighted measurements, in optimization, 39, 109, 110
when keyword, 113
white space, searching for, 45, 46
width parameters
  capacitor, 174
  diodes, 187
  MESFET device, 192
  MOSFET Level 49/53, 457–458
  resistor, 199
  See also length and width parameters
width, pulse
  vectorized current source, 181
wildcards
  in .print statements, 138
words, selecting, 44
wrapping text, 68

X
X, 100, 162
x
  key letter, 162, 190
  metric prefix, 69

Y
-y command-line flag, 42

Z
Z, 136
z
  key letter, 192
z0 device parameter
    transmission line, 203
zero initial values, 37
Credits

Software Development

Ken Van de Houten  Dan’l Leviton

Quality Assurance

Luba Gromova  Lena Neo
Ken Van de Houten

Documentation

Judy Bergstresser  Ken Van de Houten

Additional Credits

T-Spice is based on version 5.0 of CAzM (Circuit AnalyZer with Macromodeling) written by Donald J. Erdman and Donald J. Rose Gary B. Nifong, Bill Richards, Stephen Kenkel and Ravi Subrahmanyan MCNC Center for Microelectronics and Duke University.


T-Spice uses the Sparse1.3 matrix library, copyright © 1985-1988 by Kenneth S. Kundert, University of California, Berkeley.

T-Spice uses the SuperLU sparse linear solver library, copyright © 2003, The Regents of the University of California.

BSIM models BSIM4, BSIM3v322 and BSIMSOI are based upon software developed by the BSIM Research Group, University of California, Berkeley, ©1990-2004, Regents of the University of California.

EKV models are based on the EPFL-EKV MOSFET Report by the Electronics Laboratories, Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland, ©1998.

Philips models are based upon software developed by Philips Electronics N.V. ©2005

SiMKit is a trademark of Philips Electronics N.V.
The following copyright notice is applicable to Sparse, SuperLU, and BSIM:
THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO,
THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The following copyright notice is applicable to the KLU sparse linear solver, together with it’s component libraries,
AMD, COLAMD, and BTF, which are distributed herein as klu.dll:
KLU Version 1.0, Copyright © 2007 University of Florida. All Rights Reserved.
AMD Version 2.2, Copyright © 2007 by Timothy A. Davis, Patrick R. Amestoy, and Iain S. Duff. All Rights
Reserved.
This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General
Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option)
any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY;
without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See
the GNU Lesser General Public License for more details.
You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to
the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Permission is hereby granted to use or copy this program under the terms of the GNU LGPL, provided that the
Copyright, this License, and the Availability of the original version is retained on all copies. User documentation of
any code that uses this code or any modified version of this code must cite the Copyright, this License, the
Availability note, and "Used by permission." Permission to modify the code and to distribute modified code is
granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was
modified is included.
Availability:
http://www.cise.ufl.edu/research/sparse/klu
http://www.cise.ufl.edu/research/sparse/btf
http://www.cise.ufl.edu/research/sparse/amd
Acknowledgments:
This work was supported by Sandia National Laboratories, and the National Science Foundation under grants
0203270 and 0620286.
END-USER LICENSE AGREEMENT ("Agreement")

This is a legal agreement concerning the use of Software (as defined in Section 2) and hardware (collectively "Products") between the company acquiring the Products ("Customer"), and the Mentor Graphics entity that issued the corresponding quotation or, if no quotation was issued, the applicable local Mentor Graphics entity ("Mentor Graphics"). Except for license agreements related to the subject matter of this license agreement which are physically signed by Customer and an authorized representative of Mentor Graphics, this Agreement and the applicable quotation contain the parties' entire understanding relating to the subject matter and supersede all prior or contemporaneous agreements. If Customer does not agree to these terms and conditions, promptly return or, in the case of Software received electronically, certify destruction of Software and all accompanying items within five days after receipt of Software and receive a full refund of any license fee paid.

1. ORDERS, FEES AND PAYMENT.

1.1. To the extent Customer (or if agreed by Mentor Graphics, Customer’s appointed third party buying agent) places and Mentor Graphics accepts purchase orders pursuant to this Agreement (each an "Order"), each Order will constitute a contract between Customer and Mentor Graphics, which shall be governed solely and exclusively by the terms and conditions of this Agreement, any applicable addenda and the applicable quotation, whether or not those documents are referenced on the Order. Any additional or conflicting terms and conditions appearing on an Order or presented in any electronic portal or automated order management system, whether or not required to be electronically accepted, will not be effective unless agreed in writing and physically signed by an authorized representative of Customer and Mentor Graphics.

1.2. Amounts invoiced will be paid, in the currency specified on the applicable invoice, within 30 days from the date of such invoice. Any past due invoices will be subject to the imposition of interest charges in the amount of one and one-half percent per month or the applicable legal rate currently in effect, whichever is lower. Prices do not include freight, insurance, customs duties, taxes or other similar charges, which Mentor Graphics will state separately in the applicable invoice. Unless timely provided with a valid certificate of exemption or other evidence that items are not taxable, Mentor Graphics will invoice Customer for all applicable taxes including, but not limited to, VAT, GST, sales tax, consumption tax and service tax. Customer will make all payments free and clear of, and without reduction for, any withholding or other taxes; any such taxes imposed on payments by Customer hereunder will be Customer’s sole responsibility. If Customer appoints a third party to place purchase orders and/or makes payments on Customer’s behalf, Customer shall be liable for payment under Orders placed by such third party in the event of default.

1.3. All Products are delivered FCA factory (Incoterms 2010), freight prepaid and invoiced to Customer, except Software delivered electronically, which shall be deemed delivered when made available to Customer for download. Mentor Graphics retains a security interest in all Products delivered under this Agreement, to secure payment of the purchase price of such Products, and Customer agrees to sign any documents that Mentor Graphics determines to be necessary or convenient for use in filing or perfecting such security interest. Mentor Graphics’ delivery of Software by electronic means is subject to Customer’s provision of both a primary and an alternate e-mail address.

2. GRANT OF LICENSE. The software installed, downloaded, or otherwise acquired by Customer under this Agreement, including any updates, modifications, revisions, copies, documentation and design data ("Software") are copyrighted, trade secret and confidential information of Mentor Graphics or its licensors, who maintain exclusive title to all Software and retain all rights not expressly granted by this Agreement. Mentor Graphics grants to Customer, subject to payment of applicable license fees, a nontransferable, nonexclusive license to use Software solely: (a) in machine-readable, object-code form (except as provided in Subsection 5.2); (b) for Customer’s internal business purposes; (c) for the term of the license; and (d) on the computer hardware and at the site authorized by Mentor Graphics. A site is restricted to a one-half mile (800 meter) radius. Customer may have Software temporarily used by an employee for telecommuting purposes from locations other than a Customer office, such as the employee’s residence, an airport or hotel, provided that such employee’s primary place of employment is the site where the Software is authorized for use. Mentor Graphics’ standard policies and programs, which vary depending on Software, license fees paid or services purchased, apply to the following: (a) relocation of Software; (b) use of Software, which may be limited, for example, to execution of a single session by a single user on the authorized hardware or for a restricted period of time (such limitations may be technically implemented through the use of authorization codes or similar devices); and (c) support services provided, including eligibility to receive telephone support, updates, modifications, and revisions. For the avoidance of doubt, if Customer provides any feedback or requests any change or enhancement to Products, whether in the course of receiving support or consulting services, evaluating Products, performing beta testing or otherwise, any inventions, product improvements, modifications or developments made by Mentor Graphics (at Mentor Graphics’ sole discretion) will be the exclusive property of Mentor Graphics.

3. ESC SOFTWARE. If Customer purchases a license to use development or prototyping tools of Mentor Graphics’ Embedded Software Channel ("ESC"), Mentor Graphics grants to Customer a nontransferable, nonexclusive license to reproduce and distribute executable
files created using ESC compilers, including the ESC run-time libraries distributed with ESC C and C++ compiler Software that are linked into a composite program as an integral part of Customer’s compiled computer program, provided that Customer distributes these files only in conjunction with Customer’s compiled computer program. Mentor Graphics does NOT grant Customer any right to duplicate, incorporate or embed copies of Mentor Graphics’ real-time operating systems or other embedded software products into Customer’s products or applications without first signing or otherwise agreeing to a separate agreement with Mentor Graphics for such purpose.

4. BETA CODE.

4.1. Portions or all of certain Software may contain code for experimental testing and evaluation (which may be either alpha or beta, collectively “Beta Code”), which may not be used without Mentor Graphics’ explicit authorization. Upon Mentor Graphics’ authorization, Mentor Graphics grants to Customer a temporary, nontransferable, nonexclusive license for experimental use to test and evaluate the Beta Code without charge for a limited period of time specified by Mentor Graphics. Mentor Graphics may choose, at its sole discretion, not to release Beta Code commercially in any form.

4.2. If Mentor Graphics authorizes Customer to use the Beta Code, Customer agrees to evaluate and test the Beta Code under normal conditions as directed by Mentor Graphics. Customer will contact Mentor Graphics periodically during Customer’s use of the Beta Code to discuss any malfunctions or suggested improvements. Upon completion of Customer’s evaluation and testing, Customer will send to Mentor Graphics a written evaluation of the Beta Code, including its strengths, weaknesses and recommended improvements.

4.3. Customer agrees to maintain Beta Code in confidence and shall restrict access to the Beta Code, including the methods and concepts utilized therein, solely to those employees and Customer location(s) authorized by Mentor Graphics to perform beta testing. Customer agrees that any written evaluations and all inventions, product improvements, modifications or developments that Mentor Graphics conceived or made during or subsequent to this Agreement, including those based partly or wholly on Customer’s feedback, will be the exclusive property of Mentor Graphics. Mentor Graphics will have exclusive rights, title and interest in all such property. The provisions of this Subsection 4.3 shall survive termination of this Agreement.

5. RESTRICTIONS ON USE.

5.1. Customer may copy Software only as reasonably necessary to support the authorized use. Each copy must include all notices and legends embedded in Software and affixed to its medium and container as received from Mentor Graphics. All copies shall remain the property of Mentor Graphics or its licensors. Customer shall maintain a record of the number and primary location of all copies of Software, including copies merged with other software, and shall make those records available to Mentor Graphics upon request. Customer shall not make Products available in any form to any person other than Customer’s employees and on-site contractors, excluding Mentor Graphics competitors, whose job performance requires access and who are under obligations of confidentiality. Customer shall take appropriate action to protect the confidentiality of Products and ensure that any person permitted access does not disclose or use Products except as permitted by this Agreement. Customer shall give Mentor Graphics written notice of any unauthorized disclosure or use of the Products as soon as Customer becomes aware of such unauthorized disclosure or use. Except as otherwise permitted for purposes of interoperability as specified by applicable and mandatory local law, Customer shall not reverse-assemble, reverse-compile, reverse-engineer or in any way derive any source code from Software. Log files, data files, rule files and script files generated by or for the Software (collectively “Files”), including without limitation files containing Standard Verification Rule Format (“SVRF”) and Tcl Verification Format (“TVF”) which are Mentor Graphics’ trade secret and proprietary syntaxes for expressing process rules, constitute or include confidential information of Mentor Graphics. Customer may share Files with third parties, excluding Mentor Graphics competitors, provided that the confidentiality of such Files is protected by written agreement at least as well as Customer protects other information of a similar nature or importance, but in any case with at least reasonable care. Customer may use Files containing SVRF or TVF only with Mentor Graphics products. Under no circumstances shall Customer use Products or Files or allow their use for the purpose of developing, enhancing or marketing any product that is in any way competitive with Products, or disclose to any third party the results of, or information pertaining to, any benchmark.

5.2. If any Software or portions thereof are provided in source code form, Customer will use the source code only to correct software errors and enhance or modify the Software for the authorized use. Customer shall not disclose or permit disclosure of source code, in whole or in part, including any of its methods or concepts, to anyone except Customer’s employees or on-site contractors, excluding Mentor Graphics competitors, with a need to know. Customer shall not copy or compile source code in any manner except to support this authorized use.

5.3. Customer may not assign this Agreement or the rights and duties under it, or relocate, sublicense, or otherwise transfer the Products, whether by operation of law or otherwise (“Attempted Transfer”), without Mentor Graphics’ prior written consent and payment of Mentor Graphics’ then-current applicable relocation and/or transfer fees. Any Attempted Transfer without Mentor Graphics’ prior written consent shall be a material breach of this Agreement and may, at Mentor Graphics’ option, result in the immediate termination of the Agreement and/or the licenses granted under this Agreement. The terms of this Agreement, including without limitation the licensing and assignment provisions, shall be binding upon Customer’s permitted successors in interest and assigns.

5.4. The provisions of this Section 5 shall survive the termination of this Agreement.

6. SUPPORT SERVICES. To the extent Customer purchases support services, Mentor Graphics will provide Customer with updates and technical support for the Products, at the Customer site(s) for which support is purchased, in accordance with Mentor Graphics’ then current End-User Support Terms located at http://supportnet.mentor.com/supportterms.

7. LIMITED WARRANTY.

7.1. Mentor Graphics warrants that during the warranty period its standard, generally supported Products, when properly installed, will substantially conform to the functional specifications set forth in the applicable user manual. Mentor Graphics does not warrant that Products will meet Customer’s requirements or that operation of Products will be uninterrupted or error free. The
warranty period is 90 days starting on the 15th day after delivery or upon installation, whichever first occurs. Customer must notify Mentor Graphics in writing of any nonconformity within the warranty period. For the avoidance of doubt, this warranty applies only to the initial shipment of Software under an Order and does not renew or reset, for example, with the delivery of (a) Software updates or (b) authorization codes or alternate Software under a transaction involving Software re-mix. This warranty shall not be valid if Products have been subject to misuse, unauthorized modification, improper installation or Customer is not in compliance with this Agreement. MENTOR GRAPHICS’ ENTIRE LIABILITY AND CUSTOMER’S EXCLUSIVE REMEDY SHALL BE, AT MENTOR GRAPHICS’ OPTION, EITHER (A) REFUND OF THE PRICE PAID UPON RETURN OF THE PRODUCTS TO MENTOR GRAPHICS OR (B) MODIFICATION OR REPLACEMENT OF THE PRODUCTS THAT DO NOT MEET THIS LIMITED WARRANTY. MENTOR GRAPHICS MAKES NO WARRANTIES WITH RESPECT TO: (A) SERVIC ES; (B) PRODUCTS PROVIDED AT NO CHARGE; OR (C) BETA CODE; ALL OF WHICH ARE PROVIDED “AS IS.”

7.2. THE WARRANTIES SET FORTH IN THIS SECTION 7 ARE EXCLUSIVE. NEITHER MENTOR GRAPHICS NOR ITS LICENSORS MAKE ANY OTHER WARRANTIES EXPRESS, IMPLIED OR STATUTORY, WITH RESPECT TO PRODUCTS PROVIDED UNDER THIS AGREEMENT. MENTOR GRAPHICS AND ITS LICENSORS SPECIFICALLY DISCLAIM ALL IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT OF INTELLECTUAL PROPERTY.

8. LIMITATION OF LIABILITY. EXCEPT WHERE THIS EXCLUSION OR RESTRICTION OF LIABILITY WOULD BE VOID OR INEFFECTIVE UNDER APPLICABLE LAW, IN NO EVENT SHALL MENTOR GRAPHICS OR ITS LICENSORS BE LIABLE FOR INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES (INCLUDING LOST PROFITS OR SAVINGS) WHETHER BASED ON CONTRACT, TORT OR ANY OTHER LEGAL THEORY. EVEN IF MENTOR GRAPHICS OR ITS LICENSORS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL MENTOR GRAPHICS’ OR ITS LICENSORS’ LIABILITY UNDER THIS AGREEMENT EXCEED THE AMOUNT RECEIVED FROM CUSTOMER FOR THE HARDWARE, SOFTWARE LICENSE OR SERVICE GIVING RISE TO THE CLAIM. IN THE CASE WHERE NO AMOUNT WAS PAID, MENTOR GRAPHICS AND ITS LICENSORS SHALL HAVE NO LIABILITY FOR ANY DAMAGES WHATSOEVER. THE PROVISIONS OF THIS SECTION 8 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.

9. HAZARDOUS APPLICATIONS. CUSTOMER ACKNOWLEDGES IT IS SOLELY RESPONSIBLE FOR TESTING ITS PRODUCTS USED IN APPLICATIONS WHERE THE FAILURE OR INACCURACY OF ITS PRODUCTS MIGHT RESULT IN DEATH OR PERSONAL INJURY (“HAZARDOUS APPLICATIONS”). EXCEPT TO THE EXTENT THIS EXCLUSION OR RESTRICTION OF LIABILITY WOULD BE VOID OR INEFFECTIVE UNDER APPLICABLE LAW, IN NO EVENT SHALL MENTOR GRAPHICS OR ITS LICENSORS BE LIABLE FOR ANY DAMAGES RESULTING FROM OR IN CONNECTION WITH THE USE OF MENTOR GRAPHICS PRODUCTS IN OR FOR HAZARDOUS APPLICATIONS. THE PROVISIONS OF THIS SECTION 9 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.

10. INDEMNIFICATION. CUSTOMER AGREES TO INDEMNIFY AND HOLD HARMLESS MENTOR GRAPHICS AND ITS LICENSORS FROM ANY CLAIMS, LOSS, COST, DAMAGE, EXPENSE OR LIABILITY, INCLUDING ATTORNEYS’ FEES, ARISING OUT OF OR IN CONNECTION WITH THE USE OF MENTOR GRAPHICS PRODUCTS IN OR FOR HAZARDOUS APPLICATIONS. THE PROVISIONS OF THIS SECTION 10 SHALL SURVIVE THE TERMINATION OF THIS AGREEMENT.

11. INFRINGEMENT.

11.1. Mentor Graphics will defend or settle, at its option and expense, any action brought against Customer in the United States, Canada, Japan, or member state of the European Union which alleges that any standard, generally supported Product acquired by Customer hereunder infringes a patent or copyright or misappropriates a trade secret in such jurisdiction. Mentor Graphics will pay all costs and damages finally awarded against Customer that are attributable to such action. Customer understands and agrees that as conditions to Mentor Graphics’ obligations under this section Customer must: (a) notify Mentor Graphics promptly in writing of the action; (b) provide Mentor Graphics all reasonable information and assistance to settle or defend the action; and (c) grant Mentor Graphics sole authority and control of the defense or settlement of the action.

11.2. If a claim is made under Subsection 11.1 Mentor Graphics may, at its option and expense: (a) replace or modify the Product so that it becomes noninfringing; (b) procure for Customer the right to continue using the Product; or (c) require the return of the Product and refund to Customer any purchase price or license fee paid, less a reasonable allowance for use.

11.3. Mentor Graphics has no liability to Customer if the action is based upon: (a) the combination of Software or hardware with any product not furnished by Mentor Graphics; (b) the modification of the Product other than by Mentor Graphics; (c) the use of other than a current unaltered release of Software; (d) the use of the Product as part of an infringing process; (e) a product that Customer makes, uses, or sells; (f) any Beta Code or Product provided at no charge; (g) any software provided by Mentor Graphics’ licensors who do not provide such indemnification to Mentor Graphics’ customers; or (h) infringement by Customer that is deemed willful. In the case of (b), Customer shall reimburse Mentor Graphics for its reasonable attorney fees and other costs related to the action.

11.4. THIS SECTION 11 IS SUBJECT TO SECTION 8 ABOVE AND STATES THE ENTIRE LIABILITY OF MENTOR GRAPHICS AND ITS LICENSORS, AND CUSTOMER’S SOLE AND EXCLUSIVE REMEDY, FOR DEFENSE, SETTLEMENT AND DAMAGES, WITH RESPECT TO ANY ALLEGED PATENT OR COPYRIGHT INFRINGEMENT OR TRADE SECRET MISAPPROPRIATION BY ANY PRODUCT PROVIDED UNDER THIS AGREEMENT.

12. TERMINATION AND EFFECT OF TERMINATION.

12.1. If a Software license was provided for limited term use, such license will automatically terminate at the end of the authorized term. Mentor Graphics may terminate this Agreement and/or any license granted under this Agreement immediately upon written notice if Customer: (a) exceeds the scope of the license or otherwise fails to comply with the licensing or confidentiality provisions of this Agreement, or (b) becomes insolvent, files a bankruptcy petition, institutes proceedings for liquidation or winding up or enters into an agreement to assign its assets for the benefit of creditors. For any other material breach of any
19. MISCELLANEOUS.

13. EXPORT. The Products provided hereunder are subject to regulation by local laws and United States ("U.S.") government agencies, which prohibit export, re-export or diversion of certain products, information about the products, and direct or indirect products thereof, to certain countries and certain persons. Customer agrees that it will not export or re-export Products in any manner without first obtaining all necessary approval from appropriate local and U.S. government agencies. If Customer wishes to disclose any information to Mentor Graphics that is subject to any U.S. or other applicable export restrictions, including without limitation the U.S. International Traffic in Arms Regulations (ITAR) or special controls under the Export Administration Regulations (EAR), Customer will notify Mentor Graphics personnel, in advance of each instance of disclosure, that such information is subject to such export restrictions.

14. U.S. GOVERNMENT LICENSE RIGHTS. Software was developed entirely at private expense. The parties agree that all Software is commercial computer software within the meaning of the applicable acquisition regulations. Accordingly, pursuant to U.S. FAR 48 CFR 12.212 and DFAR 48 CFR 227.7202, use, duplication and disclosure of the Software by or for the U.S. government or a U.S. government subcontractor is subject solely to the terms and conditions set forth in this Agreement, which shall supersede any conflicting terms or conditions in any government order document, except for provisions which are contrary to applicable mandatory federal laws.

15. THIRD PARTY BENEFICIARY. Mentor Graphics Corporation, Mentor Graphics (Ireland) Limited, Microsoft Corporation and other licensors may be third party beneficiaries of this Agreement with the right to enforce the obligations set forth herein.

16. REVIEW OF LICENSE USAGE. Customer will monitor the access to and use of Software. With prior written notice and during Customer’s normal business hours, Mentor Graphics may engage an internationally recognized accounting firm to review Customer’s software monitoring system and records deemed relevant by the internationally recognized accounting firm to confirm Customer’s compliance with the terms of this Agreement or U.S. or other local export laws. Such review may include FlexNet (or successor product) report log files that Customer shall capture and provide at Mentor Graphics’ request. Customer shall make records available in electronic format and shall fully cooperate with data gathering to support the license review. Mentor Graphics shall bear the expense of any such review unless a material non-compliance is revealed. Mentor Graphics shall treat as confidential information all information gained as a result of any request or review and shall only use or disclose such information as required by law or to enforce its rights under this Agreement. The provisions of this Section 16 shall survive the termination of this Agreement.

17. CONTROLLING LAW, JURISDICTION AND DISPUTE RESOLUTION. The owners of certain Mentor Graphics intellectual property licensed under this Agreement are located in Ireland and the U.S. To promote consistency around the world, disputes shall be resolved as follows: excluding conflict of laws rules, this Agreement shall be governed by and construed under the laws of the State of Oregon, U.S., if Customer is located in North or South America, and the laws of Ireland if Customer is located outside of North or South America. All disputes arising out of or in relation to this Agreement shall be submitted to the exclusive jurisdiction of the courts of Portland, Oregon when the laws of Oregon apply, or Dublin, Ireland when the laws of Ireland apply. Notwithstanding the foregoing, all disputes in Asia arising out of or in relation to this Agreement shall be resolved by arbitration in Singapore before a single arbitrator to be appointed by the chairman of the Singapore International Arbitration Centre (“SIAC”) to be conducted in the English language, in accordance with the Arbitration Rules of the SIAC in effect at the time of the dispute, which rules are deemed to be incorporated by reference in this section. Nothing in this section shall restrict Mentor Graphics’ right to bring an action (including for example a motion for injunctive relief) against Customer in the jurisdiction where Customer’s place of business is located. The United Nations Convention on Contracts for the International Sale of Goods does not apply to this Agreement.

18. SEVERABILITY. If any provision of this Agreement is held by a court of competent jurisdiction to be void, invalid, unenforceable or illegal, such provision shall be severed from this Agreement and the remaining provisions will remain in full force and effect.

19. MISCELLANEOUS. This Agreement contains the parties’ entire understanding relating to its subject matter and supersedes all prior or contemporaneous agreements. Some Software may contain code distributed under a third party license agreement that may provide additional rights to Customer. Please see the applicable Software documentation for details. This Agreement may only be modified in writing, signed by an authorized representative of each party. Waiver of terms or excuse of breach must be in writing and shall not constitute subsequent consent, waiver or excuse.

Rev. 140201, Part No. 258976