## Contents

### Before you begin

Overview ................................................................. 11
Typographical conventions ........................................ 12
Command syntax formats ........................................... 12
Numeric value conventions .......................................... 13
Numeric expression conventions ................................. 13
Command line options .............................................. 18
Command files ......................................................... 18
Log files ............................................................... 20
Simulation command line specification format ............... 23

### 1 Commands

In this Chapter ......................................................... 27
Standard analyses ..................................................... 27
Output control ......................................................... 27
Simple multi-run analyses ......................................... 27
Circuit file processing .............................................. 27
Statistical analyses .................................................. 27
Device modeling ...................................................... 27
Initial conditions ..................................................... 28
Miscellaneous ......................................................... 28

Command reference for PSpice and PSpice A/D ............. 29
.AC (AC analysis) ...................................................... 32
.AIASES, .ENDAIASES (aliases and endaliases) ............... 34
.AUTOCONVERGE (Autoconvergence of simulations) ......... 35
.CHECKPT (Generate CheckPoints) ............................... 36
.DC (DC analysis) ..................................................... 37
  Linear sweep ......................................................... 37
  Logarithmic sweep ............................................... 38
# PSpice Reference Guide

- **Nested sweep** ................................................................. 39
- **DISTRIBUTION** (user-defined distribution) ........................................ 41
  - Deriving updated parameter values ........................................ 42
- **DMFACTOR** ........................................................................ 43
- **END** (end of circuit) .............................................................. 44
- **ENDS** (end subcircuit) ............................................................ 45
- **EXTERNAL** (external port) ....................................................... 46
- **FOUR** (Fourier analysis) ......................................................... 47
- **FUNC** (function) ................................................................. 48
- **IC** (initial bias point condition) ............................................. 49
- **INC** (include file) ............................................................... 50
- **LIB** (library file) .................................................................. 51
- **LOADBIAS** (load bias point file) .............................................. 53
- **MC** (Monte Carlo analysis) ...................................................... 54
- **MODEL** (model definition) ...................................................... 58
  - Parameters for setting temperature ......................................... 63
- **NODESET** (set approximate node voltage for bias point) ................ 66
- **NOISE** (noise analysis) .......................................................... 67
- **RESTART** (Restart simulation from a CheckPoint) ......................... 69
- **OP** (bias point) ..................................................................... 70
- **OPTIONS** (analysis options) .................................................... 71
  - PSpice A/D digital simulation condition messages ......................... 78
- **PARAM** (parameter) .............................................................. 80
- **PLOT** (plot) ........................................................................... 82
- **PRINT** (print) ........................................................................ 84
- **PROBE** (Probe) ...................................................................... 86
  - DC Sweep and transient analysis output variables ......................... 87
  - AC analysis ............................................................................ 91
- **SAVEBIAS** (save bias point to file) ........................................... 94
- **SENS** (sensitivity analysis) ....................................................... 98
- **STEP** (parametric analysis) ...................................................... 99
  - Usage examples ....................................................................... 101
- **STIMLIB** (stimulus library file) ............................................... 103
- **STIMULUS** (stimulus) ............................................................. 104
- **SUBCKT** (subcircuit) .............................................................. 105
  - Usage examples ....................................................................... 107
<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEMP (temperature)</td>
<td>109</td>
</tr>
<tr>
<td>TEXT (text parameter)</td>
<td>110</td>
</tr>
<tr>
<td>TF (transfer)</td>
<td>112</td>
</tr>
<tr>
<td>TRAN (transient analysis)</td>
<td>113</td>
</tr>
<tr>
<td>VECTOR (digital output)</td>
<td>116</td>
</tr>
<tr>
<td>WATCH (watch analysis results)</td>
<td>119</td>
</tr>
<tr>
<td>.WCASE (sensitivity/worst-case analysis)</td>
<td>121</td>
</tr>
<tr>
<td>* (comment)</td>
<td>125</td>
</tr>
<tr>
<td>; (in-line comment)</td>
<td>126</td>
</tr>
<tr>
<td>+ (line continuation)</td>
<td>127</td>
</tr>
<tr>
<td>Differences between PSpice and Berkeley SPICE2</td>
<td>128</td>
</tr>
</tbody>
</table>

## 2 Analog devices

### Analog devices

#### Analog devices

<table>
<thead>
<tr>
<th>Device types</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analog device summary</td>
<td>134</td>
</tr>
</tbody>
</table>

#### GaAsFET

<table>
<thead>
<tr>
<th>Capture parts</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model Parameters</td>
<td>140</td>
</tr>
<tr>
<td>GaAsFET equations</td>
<td>146</td>
</tr>
<tr>
<td>References</td>
<td>155</td>
</tr>
</tbody>
</table>

#### Capacitor

<table>
<thead>
<tr>
<th>Capture parts</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacitor model parameters</td>
<td>158</td>
</tr>
<tr>
<td>Capacitor equations</td>
<td>159</td>
</tr>
</tbody>
</table>

#### Diode

<table>
<thead>
<tr>
<th>Capture parts</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diode model parameters</td>
<td>161</td>
</tr>
<tr>
<td>Diode equations</td>
<td>162</td>
</tr>
<tr>
<td>References</td>
<td>164</td>
</tr>
</tbody>
</table>

#### Voltage-controlled voltage source

#### Voltage-controlled current source

<table>
<thead>
<tr>
<th>Basic SPICE polynomial expressions (POLY)</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current-controlled current source</td>
<td>165</td>
</tr>
<tr>
<td>Current-controlled voltage source</td>
<td>170</td>
</tr>
</tbody>
</table>

### References

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>131</td>
</tr>
<tr>
<td>133</td>
</tr>
<tr>
<td>134</td>
</tr>
<tr>
<td>137</td>
</tr>
<tr>
<td>138</td>
</tr>
<tr>
<td>140</td>
</tr>
<tr>
<td>146</td>
</tr>
<tr>
<td>155</td>
</tr>
<tr>
<td>156</td>
</tr>
<tr>
<td>157</td>
</tr>
<tr>
<td>158</td>
</tr>
<tr>
<td>159</td>
</tr>
<tr>
<td>160</td>
</tr>
<tr>
<td>161</td>
</tr>
<tr>
<td>161</td>
</tr>
<tr>
<td>162</td>
</tr>
<tr>
<td>164</td>
</tr>
<tr>
<td>165</td>
</tr>
<tr>
<td>170</td>
</tr>
<tr>
<td>174</td>
</tr>
</tbody>
</table>
# PSpice Reference Guide

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capture parts</td>
<td>302</td>
</tr>
<tr>
<td>Resistor model parameters</td>
<td>305</td>
</tr>
<tr>
<td>Resistor equations</td>
<td>306</td>
</tr>
<tr>
<td>Voltage-Controlled switch</td>
<td>307</td>
</tr>
<tr>
<td>Capture parts</td>
<td>308</td>
</tr>
<tr>
<td>Switch equations</td>
<td>310</td>
</tr>
<tr>
<td>Transmission line</td>
<td>312</td>
</tr>
<tr>
<td>Ideal line</td>
<td>313</td>
</tr>
<tr>
<td>Lossy line</td>
<td>314</td>
</tr>
<tr>
<td>Capture parts</td>
<td>316</td>
</tr>
<tr>
<td>Transmission line model parameters</td>
<td>320</td>
</tr>
<tr>
<td>References</td>
<td>321</td>
</tr>
<tr>
<td>Independent voltage source &amp; stimulus</td>
<td>322</td>
</tr>
<tr>
<td>Current-Controlled switch</td>
<td>322</td>
</tr>
<tr>
<td>Capture parts</td>
<td>324</td>
</tr>
<tr>
<td>Variable-Resistance switch model parameters</td>
<td>325</td>
</tr>
<tr>
<td>Short-Transition switch model parameters</td>
<td>325</td>
</tr>
<tr>
<td>Switch equations</td>
<td>326</td>
</tr>
<tr>
<td>Subcircuit instantiation</td>
<td>329</td>
</tr>
<tr>
<td>Operational Amplifiers (OpAmp) Model Parameters</td>
<td>330</td>
</tr>
<tr>
<td>IGBT</td>
<td>332</td>
</tr>
<tr>
<td>Capture parts</td>
<td>333</td>
</tr>
<tr>
<td>IGBT device parameters</td>
<td>334</td>
</tr>
<tr>
<td>IGBT model parameters</td>
<td>334</td>
</tr>
<tr>
<td>IGBT equations</td>
<td>336</td>
</tr>
<tr>
<td>References</td>
<td>339</td>
</tr>
<tr>
<td>Battery Model</td>
<td>340</td>
</tr>
</tbody>
</table>

## 3 Digital devices

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital device summary</td>
<td>343</td>
</tr>
<tr>
<td>Digital primitive summary</td>
<td>344</td>
</tr>
<tr>
<td>General digital primitive format</td>
<td>345</td>
</tr>
<tr>
<td>Timing models</td>
<td>349</td>
</tr>
<tr>
<td>Gates</td>
<td>353</td>
</tr>
</tbody>
</table>
# Behavioral Simulation Functions

4

**Behavioral Simulation Functions**  
447

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZERO()</td>
<td>448</td>
</tr>
<tr>
<td>ONE(expression)</td>
<td>448</td>
</tr>
<tr>
<td>CEIL(arg)</td>
<td>448</td>
</tr>
<tr>
<td>FLOOR(arg)</td>
<td>449</td>
</tr>
<tr>
<td>INTQ(arg)</td>
<td>449</td>
</tr>
<tr>
<td>DELTA()</td>
<td>450</td>
</tr>
<tr>
<td>TIME()</td>
<td>450</td>
</tr>
<tr>
<td>STATE()</td>
<td>451</td>
</tr>
<tr>
<td>BREAK()</td>
<td>452</td>
</tr>
<tr>
<td>System Variables</td>
<td>453</td>
</tr>
</tbody>
</table>
# Behavioral Simulation Models

Function library

<table>
<thead>
<tr>
<th>Model</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASW</td>
<td>467</td>
</tr>
<tr>
<td>ASW1</td>
<td>468</td>
</tr>
<tr>
<td>BEHAV_FREQ</td>
<td>469</td>
</tr>
<tr>
<td>BEHAV_GEN</td>
<td>470</td>
</tr>
<tr>
<td>Coil Spring</td>
<td>471</td>
</tr>
<tr>
<td>CURRENT_FREQ</td>
<td>472</td>
</tr>
<tr>
<td>DC Motor</td>
<td>473</td>
</tr>
<tr>
<td>Tachometer</td>
<td>476</td>
</tr>
<tr>
<td>COMPLEX_FZ</td>
<td>477</td>
</tr>
<tr>
<td>COMPLEX_RI</td>
<td>480</td>
</tr>
<tr>
<td>Flywheel</td>
<td>482</td>
</tr>
<tr>
<td>Gearbox</td>
<td>483</td>
</tr>
<tr>
<td>IN</td>
<td>484</td>
</tr>
<tr>
<td>OUT</td>
<td>485</td>
</tr>
<tr>
<td>ONE_SHOT</td>
<td>486</td>
</tr>
<tr>
<td>VCO</td>
<td>487</td>
</tr>
<tr>
<td>ILIM</td>
<td>489</td>
</tr>
<tr>
<td>VLIM</td>
<td>490</td>
</tr>
<tr>
<td>INTEGRATOR</td>
<td>491</td>
</tr>
<tr>
<td>DIFFERENTIATOR</td>
<td>492</td>
</tr>
<tr>
<td>Transfer Functions</td>
<td>493</td>
</tr>
<tr>
<td>REALPOLE2</td>
<td>495</td>
</tr>
<tr>
<td>REALZERO2</td>
<td>496</td>
</tr>
<tr>
<td>SLEW_LIMIT</td>
<td>497</td>
</tr>
</tbody>
</table>

Spice_elem library

<table>
<thead>
<tr>
<th>Model</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC_SWITCH</td>
<td>500</td>
</tr>
<tr>
<td>VV_SWITCH</td>
<td>501</td>
</tr>
<tr>
<td>CVS10</td>
<td>502</td>
</tr>
<tr>
<td>CCS10</td>
<td>505</td>
</tr>
<tr>
<td>CVS23</td>
<td>507</td>
</tr>
<tr>
<td>CCS23</td>
<td>508</td>
</tr>
<tr>
<td>Currentsense</td>
<td>509</td>
</tr>
</tbody>
</table>
Voltagesense ................................................................. 510
CCCS ........................................................................... 511
VCCS ........................................................................ 512
CCVS ........................................................................ 513
VCVS .......................................................................... 514
Delay lines ................................................................. 515
DC transformer ......................................................... 516
VC CAP ....................................................................... 517
VC CON ....................................................................... 518
VC IND ........................................................................ 519
VC RES ........................................................................ 520

Glossary ....................................................................... 521

A
Appendix A: Battery Models ............................................. 529

Index ............................................................................. 533
Before you begin

Overview

This manual contains the reference material needed when working with special circuit analyses in PSpice.

Included in this manual are detailed command descriptions, start-up option definitions, and a list of supported devices in the digital and analog device libraries.

This manual has comprehensive reference material for all of the PSpice circuit analysis applications, which include:

- PSpice A/D
- PSpice

This manual assumes that you are familiar with Microsoft Windows, including how to use icons, menus and dialog boxes. It also assumes you have a basic understanding about how Windows manages applications and files to perform routine tasks, such as starting applications and opening and saving your work. If you are new to Windows, please review your Microsoft Windows User's Guide.
Typographical conventions

This manual generally follows the conventions used in the Microsoft Windows User’s Guide. Procedures for performing an operation are generally numbered with the following typographical conventions.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>monospace font</td>
<td>mydiodes.slb</td>
<td>Library files and file names.</td>
</tr>
<tr>
<td>key cap or letter</td>
<td>Press J ...</td>
<td>A specific key or key stroke on the keyboard.</td>
</tr>
<tr>
<td>monospace font</td>
<td>Type vac...</td>
<td>Output produced by a printer and commands/text entered from the keyboard.</td>
</tr>
</tbody>
</table>

Command syntax formats

The following table provides the command syntax formats.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Examples</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>monospace font</td>
<td>abcd</td>
<td>User input including keypad symbols, numerals, and alphabetic characters as shown; alphabetic characters are not case sensitive.</td>
</tr>
<tr>
<td>&lt; &gt;</td>
<td>&lt;model name&gt;</td>
<td>A required item in a command line. For example, &lt;model name&gt; in a command line means that the model name parameter is required.</td>
</tr>
<tr>
<td>&lt; &gt;*</td>
<td>&lt;value&gt;*</td>
<td>The asterisk indicates that the item shown in italics must occur one or more times in the command line.</td>
</tr>
<tr>
<td>[ ]</td>
<td>[AC]</td>
<td>Optional item.</td>
</tr>
<tr>
<td>[ ]*</td>
<td>[value]*</td>
<td>The asterisk indicates that there is zero or more occurrences of the specified subject.</td>
</tr>
<tr>
<td>&lt;</td>
<td>&gt;</td>
<td>&lt;YES</td>
</tr>
<tr>
<td>[</td>
<td>]</td>
<td>[ON</td>
</tr>
</tbody>
</table>
Numeric value conventions

The numeric value and expression conventions in the following table not only apply to the PSpice Commands on page 27, but also to the device declarations and interactive numeric entries described in subsequent chapters.

Literal numeric values are written in standard floating point notation. PSpice applies the default units for the numbers describing the component values and electrical quantities. However, these values can be scaled by following the number using the appropriate scale suffix as shown in the following table.

<table>
<thead>
<tr>
<th>Scale</th>
<th>Symbol</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-15}$</td>
<td>F</td>
<td>femto-</td>
</tr>
<tr>
<td>$10^{-12}$</td>
<td>P</td>
<td>pico-</td>
</tr>
<tr>
<td>$10^{-9}$</td>
<td>N</td>
<td>nano-</td>
</tr>
<tr>
<td>$10^{-6}$</td>
<td>U</td>
<td>micro-</td>
</tr>
<tr>
<td>$25.4 \times 10^{-6}$</td>
<td>MIL</td>
<td>--</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>M</td>
<td>milli-</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>clock cycle(^1)</td>
</tr>
<tr>
<td>$10^{+3}$</td>
<td>K</td>
<td>kilo-</td>
</tr>
<tr>
<td>$10^{+6}$</td>
<td>MEG</td>
<td>mega-</td>
</tr>
<tr>
<td>$10^{+9}$</td>
<td>G</td>
<td>giga-</td>
</tr>
<tr>
<td>$10^{+12}$</td>
<td>T</td>
<td>tera-</td>
</tr>
</tbody>
</table>

1. Clock cycle varies and must be set where applicable.

Numeric expression conventions

Numeric values can also be indirectly represented by parameters; see the ".PARAM (parameter)" on page 80 command. Numeric values and parameters can be used together to form arithmetic expressions. PSpice expressions can incorporate the intrinsic functions shown in the following table.

The Function column lists expressions that PSpice and PSpice A/D recognize. The Meaning column lists the mathematical definition of the function. There are also some differences
between the intrinsic functions available for simulation and those available for waveform analysis. Refer to your *PSpice User’s Guide* for more information about waveform analysis.

<table>
<thead>
<tr>
<th>Function 1</th>
<th>Meaning</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS(x)</td>
<td></td>
<td>[x]</td>
</tr>
<tr>
<td>ACOS(x)</td>
<td>arccosine of x</td>
<td>-1.0 [x] 1.0</td>
</tr>
<tr>
<td>ACOSH(x)</td>
<td>inverse hyperbolic cosine of x</td>
<td>x is an expression</td>
</tr>
<tr>
<td>ARCTAN(x)</td>
<td>[tan^{-1}(x)]</td>
<td>result in radians</td>
</tr>
<tr>
<td>ASIN(x)</td>
<td>arcsine of x</td>
<td>-1.0 [x] 1.0</td>
</tr>
<tr>
<td>ASINH(x)</td>
<td>inverse hyperbolic sine of x</td>
<td>x is an expression</td>
</tr>
<tr>
<td>ATAN(x)</td>
<td>[tan^{-1}(x)]</td>
<td>result in radians</td>
</tr>
<tr>
<td>ATAN2(y,x)</td>
<td>arctan of (y/x)</td>
<td>result in radians</td>
</tr>
<tr>
<td>ATANH(x)</td>
<td>inverse hyperbolic tan of x</td>
<td>x is an expression</td>
</tr>
<tr>
<td>COS(x)</td>
<td>cos(x)</td>
<td>x in radians</td>
</tr>
<tr>
<td>COSH(x)</td>
<td>hyperbolic cosine of x</td>
<td>x in radians</td>
</tr>
<tr>
<td>DDT(x)</td>
<td>time derivative of x</td>
<td>transient analysis only</td>
</tr>
<tr>
<td>EXP(x)</td>
<td>[e^x]</td>
<td></td>
</tr>
</tbody>
</table>
| IF(t, x, y)| x if t=TRUE  
y if t=FALSE | t is a Boolean expression that evaluates to TRUE or FALSE and can include logical and relational operators (see Command line options on page 18). X and Y are either numeric values or expressions. For example,  
{IF ( v(1)<THL, v(1), v(1)*v(1)/THL )} |
<p>|           |         | Care should be taken in modeling the discontinuity between the IF and ELSE parts, or convergence problems can result. |</p>
<table>
<thead>
<tr>
<th>Function</th>
<th>Meaning</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>IMG(x)</td>
<td>imaginary part of x</td>
<td>returns 0.0 for real numbers</td>
</tr>
<tr>
<td>LIMIT(x, min, max)</td>
<td>result is min if x &lt; min, max if x &gt; max, and x otherwise</td>
<td></td>
</tr>
<tr>
<td>LOG(x)</td>
<td>( \ln(x) )</td>
<td>log base e</td>
</tr>
<tr>
<td>LOG10(x)</td>
<td>( \log(x) )</td>
<td>log base 10</td>
</tr>
<tr>
<td>M(x)</td>
<td>magnitude of x</td>
<td>this produces the same result as ABS(x)</td>
</tr>
<tr>
<td>MAX(x, y)</td>
<td>maximum of x and y</td>
<td></td>
</tr>
<tr>
<td>MIN(x, y)</td>
<td>minimum of x and y</td>
<td></td>
</tr>
<tr>
<td>P(x)</td>
<td>phase of x</td>
<td>returns 0.0 for real numbers</td>
</tr>
<tr>
<td>PWR(x, y)</td>
<td>(</td>
<td>x</td>
</tr>
<tr>
<td>PWRS(x, y)</td>
<td>(</td>
<td>x</td>
</tr>
<tr>
<td>R(x)</td>
<td>real part of x</td>
<td></td>
</tr>
<tr>
<td>SCHEDULE</td>
<td>piecewise constant function: from time x forward use y</td>
<td>Must include an entry for TIME=0. All y values must be legal for the associated parameter. Time (x) values must be ( \geq 0 ).</td>
</tr>
<tr>
<td>SDT(x)</td>
<td>time integral of x</td>
<td>For transient analysis only.</td>
</tr>
<tr>
<td>SGN(x)</td>
<td>signum function</td>
<td></td>
</tr>
<tr>
<td>SIN(x)</td>
<td>( \sin(x) )</td>
<td>x in radians</td>
</tr>
<tr>
<td>SINH(x)</td>
<td>hyperbolic sine of x</td>
<td>x in radians</td>
</tr>
<tr>
<td>STP(x)</td>
<td>1 if x( \geq 0.0 ), 0 if x( &lt;0.0 )</td>
<td>The unit step function can be used to suppress a value until a given amount of time has passed. For instance, {v(1)*STP(TIME-10ns)} gives a value of 0.0 until 10ns has elapsed, then gives ( v(1) ).</td>
</tr>
<tr>
<td>SQRT(x)</td>
<td>( x^{1/2} )</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>Meaning</td>
<td>Comments</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>TAN(x)</td>
<td>tan(x)</td>
<td>x in radians</td>
</tr>
<tr>
<td>TANH(x)</td>
<td>hyperbolic tangent of x</td>
<td>x in radians</td>
</tr>
<tr>
<td>TABLE</td>
<td>(x, x1, y1, x2, y2, ..., x_n, y_n)</td>
<td>Result is the y value corresponding to x, when all of the x_n,y_n points are plotted and connected by straight lines. If x is greater than the max x_n, then the value is the y_n associated with the largest x_n. If x is less than the smallest x_n, then the value is the y_n associated with the smallest x_n.</td>
</tr>
<tr>
<td>zero(expression)</td>
<td></td>
<td>The expression is evaluated and the result is ignored. Function returns the value 0.</td>
</tr>
<tr>
<td>one(expression)</td>
<td></td>
<td>The expression is evaluated and the result is ignored. Function returns the value 1.</td>
</tr>
<tr>
<td>ceil(arg)</td>
<td></td>
<td>Returns an integer value. The argument for this function should be a numeric value or an expression that evaluates to a numeric value. If arg is an integer, the return value is equal to the argument value. If arg is a non-integer value, the return value is the nearest integer greater than the argument value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Example:</td>
</tr>
<tr>
<td></td>
<td>EE15 12 0 VALUE=+</td>
<td>{ZERO(BREAK(time+10e-6))}</td>
</tr>
<tr>
<td></td>
<td>ceil(PI) = 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ceil(5) = 5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ceil(5.4) = 6</td>
<td></td>
</tr>
</tbody>
</table>
### Expressions

Expressions can contain the standard operators as shown in the following table.

#### Operators

<table>
<thead>
<tr>
<th>Operators</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>arithmetic</strong></td>
<td></td>
</tr>
<tr>
<td>+</td>
<td>addition (or string concatenation)</td>
</tr>
<tr>
<td>-</td>
<td>subtraction</td>
</tr>
<tr>
<td>*</td>
<td>multiplication</td>
</tr>
<tr>
<td>/</td>
<td>division</td>
</tr>
<tr>
<td>**</td>
<td>exponentiation</td>
</tr>
<tr>
<td>PWR()</td>
<td></td>
</tr>
<tr>
<td>PWRS()</td>
<td></td>
</tr>
</tbody>
</table>

### Floor Function

The `floor` function returns an integer value. The argument for this function should be a numeric value or an expression that evaluates to a numeric value. If `arg` is an integer, the return value is equal to the argument value. If `arg` is a non-integer value, the return value is the nearest integer smaller than the argument value.

**Example:**
- `floor(PI) = 3`
- `floor(5) = 5`
- `floor(5.4) = 5`

### Integer Function

The `intq` function returns 1 if the argument is an integer. Returns 0 if the argument is not an integer.

The argument passed to this function can be a numeric value or an expression that evaluates to a numeric value.

### Notes

1. Most numeric specifications in PSpice allow for arithmetic expressions. Some exceptions do exist and are summarized in your PSpice User’s Guide. There are also some differences between the intrinsic functions available for simulation and those available for waveform analysis. Refer to your PSpice User’s Guide for more information about waveform analysis.
Command line options

Command files

A command file is an ASCII text file which contains a list of commands to be executed. A command file can be specified in multiple ways:

- at the command line when starting PSpice, Stimulus Editor, or the Model Editor,
- by choosing Run Commands from the File menu and entering a command file name (for PSpice and Stimulus Editor only)

The command file is read by the program and all of the commands contained within the file are performed. When the end of the command file is reached, commands are taken from the keyboard and the mouse. If no command file is specified, all of the commands are received from the keyboard and mouse.

The ability to record a set of commands can be useful when using PSpice, the Model Editor, and Stimulus Editor. This is especially useful in PSpice, if you are repeatedly doing the same simulation and looking at the same waveform with only slight changes to the circuit before

---

### Operators and Meaning

<table>
<thead>
<tr>
<th>Operators</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>logical</td>
<td></td>
</tr>
<tr>
<td>~</td>
<td>unary NOT</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>^</td>
<td>boolean XOR</td>
</tr>
<tr>
<td>&amp;</td>
<td>boolean AND</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>relational (within IF( ) functions)</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>equality test</td>
</tr>
<tr>
<td>!=</td>
<td>non-equality test</td>
</tr>
<tr>
<td>&gt;</td>
<td>greater than test</td>
</tr>
<tr>
<td>&gt;=</td>
<td>greater than or equal to test</td>
</tr>
<tr>
<td>&lt;</td>
<td>less than test</td>
</tr>
<tr>
<td>&lt;=</td>
<td>less than or equal to test</td>
</tr>
</tbody>
</table>
each run. It can also be used to automatically create hardcopy output at the end of very long (such as overnight) simulation runs.

Creating and editing command files

You can create your own command file using a text editor (such as Notepad). In PSpice and Stimulus Editor, you can choose Log Commands from the File menu (see Log files on page 20 for an example) to record a list of transactions in a log file, then choose Run Commands from the File menu to run the logged file.

Note: After you activate cursors (from the Tools menu, choose Cursor), any mouse or keyboard movements that you make for moving the cursor will not be recorded in the command file.

If you choose to create a command file using a text editor, note that the commands in the command file are the same as those available from the keyboard with these differences:

- The name of the command or its first capitalized letter can be used.
- Any line that begins with an * is a comment.
- Blank lines are ignored, therefore, they can be added to improve the readability of the command file.
- The commands @CR, @UP, @DWN, @LEFT, @RIGHT, and @ESC are used to represent the <Enter>, <↑>, <↓>, <←>, <→>, and <Esc> keys, respectively.
- The command PAUSE causes PSpice, the Model Editor, or Stimulus Editor to wait until any key on the keyboard is pressed. In the case of PSpice, this can be useful to examine a waveform before the command file draws the next one.

The commands are one to a line in the file, but comment and blank lines can be used to make the file easier to read.

Assuming that a waveform data file has been created by simulating the circuit example.dsn, you can manually create a command file (using a text editor) called example.cmd which contains the commands listed below. This set of commands draws a waveform, allows you to look at it, and then exits PSpice.

* Display trace v(out2) and wait
  Trace Add
  V(out2)
  Pause
  * Exit Probe environment
  File Exit

See Simulation command line specification format on page 23 and Specifying simulation command line options on page 25 for specifying command files on the simulation command
log line. See Simulation command line specification format on page 23 and Specifying simulation command line options on page 25 for details on specifying the /C or -c option for PSpice.

Note: The Search Commands feature is a Cursor option for positioning the cursor at a particular point. You can learn more about Search Commands by consulting PSpice Help.

Log files

Instead of creating command files by hand, using a text editor, you can generate them automatically by creating a log file while running PSpice, the Model Editor, or Stimulus Editor. While executing the particular package, all of the commands given are saved in the log file. The format of the log file is correct for use as a command file.

To create a file in PSpice or Stimulus Editor, from the File menu, choose Log Commands and enter a log file name. This turns logging on. Any action taken after starting Log Commands is logged in the named file and can be run in another session by choosing Run Commands.

Note: Some commands might not be added to the log file.

You can also create a log file for PSpice, Stimulus Editor, or the Model Editor by using the /l or -l option at the command line. For example:

PSPICE /L EXAMPLE.LOG

Of course, you can use a name for the log file that is more recognizable, such as acplots.cmd (to PSpice and Stimulus Editor, the file name is any valid file name for your computer).

Note: You can use either (/) or (-) as separators, and file names can be in upper or lower case.

Editing log files

After PSpice, the Model Editor, or Stimulus Editor is finished, the log file is available for editing to customize it for use as a command file. You can edit the following items:

- Add blank lines and comments to improve readability (perhaps a title and short discussion of what the file does).
- Add the Pause command for viewing waveforms before proceeding.
- Remove the Exit command from the end of the file, so that PSpice, the Model Editor, and Stimulus Editor do not automatically exit when the end of the command file is reached.

You can add or delete other commands from the file or even change the file name to be more recognizable. It is possible to build onto log files, either by using your text editor to combine
files or by running PSpice, the Model Editor, and Stimulus Editor with both a command and log file:

```
PROBE /C IN.CMD /L OUT.LOG
```

The file `in.cmd` gives the command to PSpice, and PSpice saves the (same) commands into the `out.log` file. When `in.cmd` runs out of commands, and PSpice is taking commands from the keyboard, these commands also go into the `out.log` file.

**To log commands in PSpice**

Use command logging in PSpice to record and save frequently used actions to a command file. Command files are useful when you need to remember the steps taken in order to display a set of waveforms for any given data file.

1. From the File menu, choose Log Commands.

2. In the Log File Name text box, type `2traces`, then click OK.

   A check mark appears next to Log Command to indicate that logging is turned on.

3. From the File menu, choose Open.

4. Select `example.dat` (located in the examples directory), then click OK.

5. From the Trace menu, choose Add.

6. Select V(OUT1) and V(OUT2), then click OK.

7. From the File menu, choose Log Commands to turn command logging off.

   The check mark next to the command disappears. Subsequent actions performed are not logged in the command file.

You can view the command file using an ASCII text editor, such as Notepad. Command files can be edited or appended, depending on the types of commands you want to store for future use. The file `2traces.cmd` should look as shown below (with the exception of a different file path).

```
*Command file created by Probe - Wed Apr 17 10:33:55
File Open
    /Cadence/probe/example.dat
OK
Trace Add
V(OUT1) V(OUT2)
OK
```
To run the command log

1. From the File menu, choose Run Command.
2. Select 2traces.cmd, then click OK.
   
   The two traces appear.
Simulation command line specification format

The format for specifying command line options for PSpice and PSpice A/D are as follows.

```plaintext
pspice [options] [input file(s)]
```

Where:

**options**

One or more of the options listed in Simulation command line options on page 23. Options can be entered using the dash (-) or slash (/) separator.

**input file**

Specifies the name of a circuit file for PSpice or PSpice A/D to simulate after it starts. The input file can be a simulation file (.sim, .cir, .net), data files (.dat), output files (.out), or any files (*.). PSpice opens any files whose extension PSpice does not recognize as a text file.

You can specify multiple input files, but if the output file or data file options are specified, they apply only to the first specified input file.

The input file name can include wildcard characters (* and ?), in which case all file names matching the specification are simulated.

### Table 1-1  Simulation command line options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-bf&lt;flush interval&gt;</code></td>
<td>Determines how often (in minutes) the simulator will flush the buffers of the waveform data file to disk. This is useful when a long simulation is left running and the machine crashes or is restarted. In this case, the data file will be readable up to the last flush. The default is to flush every 10 minutes. The <code>&lt;flush interval&gt;</code> can be set between 0 and 1440 minutes. A value of zero means not to write unless necessary.</td>
</tr>
</tbody>
</table>
### Table 1-1  Simulation command line options

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-bn&lt;number of buffers&gt;</td>
<td>Determines the number of buffers to potentially allocate for the waveform data file. Zero buffers means to do all writing directly to disk. Allocating a large number of buffers can speed up a large simulation, but will increase memory requirements. Exceeding physical memory will either slow down the simulation, or will make it fail. The default number of buffers is 4 (1 buffer if you are using the CSDF option).</td>
</tr>
<tr>
<td>-bs&lt;buffer size factor&gt;</td>
<td>Determines the size of the individual buffers for writing the waveform data file. Using a larger buffer size can reduce execution time, but at the expense of increasing the memory requirements. The values for the buffer files work as follows: option:-bs0  -bs1  -bs2  -bs3  -bs4  -bs5  -bs6 value: 256  512  1024  2048  4096  8192  16384 The default is 4K (8K if you are using CSDF).</td>
</tr>
<tr>
<td>-c&lt;file name&gt;</td>
<td>Specifies the command file, which runs the session until the command file ends or PSpice stops.</td>
</tr>
<tr>
<td>-d &lt;data file&gt;</td>
<td>Specifies the name of the waveform data file to which PSpice saves the waveform data from the simulation. By default, the name of the waveform data file is the name of the input file with a .dat extension.</td>
</tr>
<tr>
<td>-e</td>
<td>Exits PSpice after all specified files have been simulated. This option replaces the -wONLY option.</td>
</tr>
<tr>
<td>-i &lt;ini file name&gt;</td>
<td>Specifies the name of an alternate initialization file. If not specified, the simulator uses: \PSPice\PSPice.ini</td>
</tr>
<tr>
<td>-l &lt;file name&gt;</td>
<td>Creates a log file, which saves the commands from this session. This log file can later be used as an input command file for PSpice.</td>
</tr>
<tr>
<td>-o &lt;output file&gt;</td>
<td>Specifies the output file to which PSpice saves the simulation output. By default, the name of the output file name defaults to the name of the input file with an .out extension.</td>
</tr>
</tbody>
</table>
Specifying simulation command line options

The command line options can be separated by spaces or in a continuous string, therefore:

- `c makeplot.cmd -p newamp.prb`
- `cmakeplot.cmd -p newamp.prb`

are equivalent. The order of the options does not matter.

The command line options that use `<file name>` assume default extensions. These command line options can be used without specifying the extension to `<file name>`. For example:

- `c makeplot -p newamp`
- `c makeplot.cmd -p newamp.prb`

are equivalent. However, PSpice searches first for the exact `<file name>` specified for these command line options, and if that `<file name>` exists, PSpice uses it. If the exact `<file name>` does not exist, PSpice adds default extensions to `<file name>` and searches for those. The following default extensions are used:

- `<file name[.dat]>` waveform data file
- `-c<file name[.cmd]>` command file
Note: You can learn more about PSpice macros by consulting PSpice Help.

-1<file name[.log]>  log file
-p<file name[.prb]>  displays, goal functions, and macros file
## Commands

### In this Chapter

<table>
<thead>
<tr>
<th>Standard analyses</th>
<th>Output control</th>
<th>Simple multi-run analyses</th>
<th>Circuit file processing</th>
<th>Statistical analyses</th>
<th>Device modeling</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>.AC</code> (AC analysis) on page 32</td>
<td><code>.PLOT</code> (plot) on page 82</td>
<td><code>.STEP</code> (parametric analysis) on page 99</td>
<td><code>.END</code> (end of circuit) on page 44</td>
<td><code>.MC</code> (Monte Carlo analysis) on page 54</td>
<td><code>.SUBCKT</code> (subcircuit) on page 105</td>
</tr>
<tr>
<td><code>.DC</code> (DC analysis) on page 37</td>
<td><code>.PRINT</code> (print) on page 84</td>
<td><code>.TEMP</code> (temperature) on page 109</td>
<td><code>.FUNC</code> (function) on page 48</td>
<td><code>.LIB</code> (library file) on page 51</td>
<td><code>.ENDS</code> (end subcircuit) on page 45</td>
</tr>
<tr>
<td><code>.FOUR</code> (Fourier analysis) on page 47</td>
<td><code>.PROBE</code> (Probe) on page 86</td>
<td></td>
<td><code>.INC</code> (include file) on page 50</td>
<td><code>.PARAM</code> (parameter) on page 80</td>
<td></td>
</tr>
<tr>
<td><code>.NOISE</code> (noise analysis) on page 67</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>.OP</code> (bias point) on page 70</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>.VECTOR</code> (digital output) on page 116</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>.SENS</code> (sensitivity analysis) on page 98</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>.TF</code> (transfer) on page 112</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>.TRAN</code> (transient analysis) on page 113</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td><code>.WATCH</code> (watch analysis results) on page 119</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Commands</td>
<td>Page</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------------------</td>
<td>------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Initial conditions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.IC (initial bias point condition)</td>
<td>49</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.LOADBIAS (load bias point file)</td>
<td>53</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.NODESET (set approximate node voltage for bias point)</td>
<td>66</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.SAVEBIAS (save bias point to file)</td>
<td>94</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Miscellaneous</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.ALIASES .ENDALIASES (aliases and endaliases)</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.AUTOCONVERGE (Autoconvergence of simulations)</td>
<td>35</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.CHKPT (Generate CheckPoints)</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.EXTERNAL (external port)</td>
<td>46</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.OPTIONS (analysis options)</td>
<td>71</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.STIMLIB (stimulus library file)</td>
<td>103</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.RESTART (Restart simulation from a CheckPoint)</td>
<td>69</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.STIMULUS (stimulus)</td>
<td>104</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.TEXT (text parameter)</td>
<td>110</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* (comment)</td>
<td>125</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>; (in-line comment)</td>
<td>126</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+ (line continuation)</td>
<td>127</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Command reference for PSpice and PSpice A/D

Schematics users enter analysis specifications through the Analysis Setup dialog box (from the Analysis menu, select Setup).

<table>
<thead>
<tr>
<th>Function</th>
<th>PSpice command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Standard Analyses</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.AC (AC analysis)</td>
<td>frequency response</td>
<td></td>
</tr>
<tr>
<td>.DC (DC analysis)</td>
<td>DC sweep</td>
<td></td>
</tr>
<tr>
<td>.FOUR (Fourier analysis)</td>
<td>Fourier components</td>
<td></td>
</tr>
<tr>
<td>.NOISE (noise analysis)</td>
<td>noise</td>
<td></td>
</tr>
<tr>
<td>.OP (bias point)</td>
<td>bias point</td>
<td></td>
</tr>
<tr>
<td>.SENS (sensitivity analysis)</td>
<td>DC sensitivity</td>
<td></td>
</tr>
<tr>
<td>.TF (transfer)</td>
<td>small-signal DC transfer function</td>
<td></td>
</tr>
<tr>
<td>.TRAN (transient analysis)</td>
<td>transient</td>
<td></td>
</tr>
<tr>
<td><strong>Simple Multi-Run Analyses</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.STEP (parametric analysis)</td>
<td>parametric</td>
<td></td>
</tr>
<tr>
<td>.TEMP (temperature)</td>
<td>temperature</td>
<td></td>
</tr>
<tr>
<td><strong>Statistical Analyses</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.MC (Monte Carlo analysis)</td>
<td>Monte Carlo</td>
<td></td>
</tr>
<tr>
<td>.WCASE (sensitivity/worst-case analysis)</td>
<td>sensitivity/worst-case</td>
<td></td>
</tr>
<tr>
<td><strong>Initial Conditions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.IC (initial bias point condition)</td>
<td>clamp node voltage for bias point calculation</td>
<td></td>
</tr>
<tr>
<td>.LOADBIAS (load bias point file)</td>
<td>to restore a .NODESET bias point</td>
<td></td>
</tr>
<tr>
<td>.NODESET (set approximate node voltage for bias point)</td>
<td>to suggest a node voltage for bias calculation</td>
<td></td>
</tr>
<tr>
<td>.SAVEBIAS (save bias point to file)</td>
<td>to store .NODESET bias point information</td>
<td></td>
</tr>
<tr>
<td><strong>Device Modeling</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.ENDS (end subcircuit)</td>
<td>end of subcircuit definition</td>
<td></td>
</tr>
<tr>
<td>.DISTRIBUTION (user-defined distribution)</td>
<td>model parameter tolerance distribution</td>
<td></td>
</tr>
<tr>
<td>.MODEL (model definition)</td>
<td>modeled device definition</td>
<td></td>
</tr>
<tr>
<td>.SUBCKT (subcircuit)</td>
<td>to start subcircuit definition</td>
<td></td>
</tr>
<tr>
<td>Function</td>
<td>PSpice command</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------</td>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Output Control</td>
<td>.PLOT (plot)</td>
<td>to send an analysis plot to output file (line printer format)</td>
</tr>
<tr>
<td></td>
<td>.PRINT (print)</td>
<td>to send an analysis table to output file</td>
</tr>
<tr>
<td></td>
<td>.PROBE (Probe)</td>
<td>to send simulation results to Probe data file</td>
</tr>
<tr>
<td></td>
<td>.VECTOR (digital output)</td>
<td>digital state output</td>
</tr>
<tr>
<td></td>
<td>.WATCH (watch analysis results)</td>
<td>view numerical simulation results in progress</td>
</tr>
<tr>
<td>Circuit File Processing</td>
<td>.END (end of circuit)</td>
<td>end of circuit simulation description</td>
</tr>
<tr>
<td></td>
<td>.FUNC (function)</td>
<td>expression function definition</td>
</tr>
<tr>
<td></td>
<td>.INC (include file)</td>
<td>include specified file</td>
</tr>
<tr>
<td></td>
<td>.LIB (library file)</td>
<td>reference specified library</td>
</tr>
<tr>
<td></td>
<td>.PARAM (parameter)</td>
<td>parameter definition</td>
</tr>
<tr>
<td>Function</td>
<td>PSpice command</td>
<td>Description</td>
</tr>
<tr>
<td>---------------</td>
<td>----------------</td>
<td>------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>miscellaneous</td>
<td>.ALIASES, .ENDALIASES (aliases and endaliases) on page 34</td>
<td>to begin and end an alias definition</td>
</tr>
<tr>
<td></td>
<td>.AUTOCONVERGE (Autoconvergence of simulations) on page 35</td>
<td>to identify nets representing the outermost (or peripheral) connections to the circuit being simulated</td>
</tr>
<tr>
<td></td>
<td>.EXTERNAL (external port) on page 46</td>
<td>to set miscellaneous simulation limits, analysis control parameters, and output characters</td>
</tr>
<tr>
<td></td>
<td>.OPTIONS (analysis options) on page 71</td>
<td>to specify a stimulus library name containing .STIMULUS information</td>
</tr>
<tr>
<td></td>
<td>.STIMLIB (stimulus library file) on page 103</td>
<td>stimulus device definition</td>
</tr>
<tr>
<td></td>
<td>.STIMULUS (stimulus) on page 104</td>
<td>text expression, parameter, or file name used by digital devices</td>
</tr>
<tr>
<td></td>
<td>.TEXT (text parameter) on page 110</td>
<td>to create a comment line</td>
</tr>
<tr>
<td></td>
<td>* (comment) on page 125</td>
<td>to add an in-line comment</td>
</tr>
<tr>
<td></td>
<td>; (in-line comment) on page 126</td>
<td>to continue the text of the previous line</td>
</tr>
<tr>
<td></td>
<td>+ (line continuation) on page 127</td>
<td></td>
</tr>
</tbody>
</table>
.AC (AC analysis)

Purpose
The .AC command calculates the frequency response of a circuit over a range of frequencies.

General form
.AC <sweep type> <points value>
+ <start frequency value> <end frequency value>

Examples
.AC LIN 101 100Hz 200Hz
.AC OCT 10 1kHz 16kHz
.AC DEC 20 1MEG 100MEG

Arguments and options

<sweep type>
Must be LIN, OCT, or DEC, as described below.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIN</td>
<td>linear sweep</td>
<td>The frequency is swept linearly from the starting to the ending frequency. The &lt;points value&gt; is the total number of points in the sweep.</td>
</tr>
<tr>
<td>OCT</td>
<td>sweep by octaves</td>
<td>The frequency is swept logarithmically by octaves. The &lt;points value&gt; is the number of points per octave.</td>
</tr>
<tr>
<td>DEC</td>
<td>sweep by decades</td>
<td>The frequency is swept logarithmically by decades. The &lt;points value&gt; is the number of points per decade.</td>
</tr>
</tbody>
</table>

<points value>
Specifies the number of points in the sweep, using an integer.

<start frequency value> <end frequency value>
The end frequency value must not be less than the start frequency value, and both must be greater than zero. The whole sweep must include at least one point. If a group delay (G suffix) is specified as an output, the frequency steps must be close enough together that the phase of that output changes smoothly from one frequency to the next. Calculate group delay by subtracting the phases of successive outputs and dividing by the frequency increment.
A .PRINT (print) on page 84, .PLOT (plot) on page 82, or .PROBE (Probe) on page 86 command must be used to get the results of the AC sweep analysis.

AC analysis is a linear analysis. The simulator calculates the frequency response by linearizing the circuit around the bias point.

All independent voltage and current sources that have AC values are inputs to the circuit. During AC analysis, the only independent sources that have nonzero amplitudes are those using AC specifications. The SIN specification does not count, as it is used only during transient analysis.

To analyze nonlinear functions such as mixers, frequency doublers, and AGC, use .TRAN (transient analysis).
.ALIASES, .ENDALIASES
(aliases and endaliases)

Purpose
The Alias commands set up equivalences between node names and pin names, so that traces in the Probe display can be identified by naming a device and pin instead of a node. They are also used to associate a net name with a node name.

General form

```
.ALIIASES
<device name>  <device alias> (<<pin>=<node>>)
_  _  (<<net>=<node>>)
.ENDALIASES
```

Examples

```
.ALIIASES
R_RBIAS  RBIAS (1=$N_0001 2=VDD)
Q_Q3  Q3 (c=$N_0001 b=$N_0001 e=VEE)
_  _  (OUT=$N_0007)
.ENDALIASES
```

The first alias definition shown in the example allows the name RBIAS to be used as an alias for R_RBIAS, and it relates pin 1 of device R_RBIAS to node $N_0001 and pin 2 to VDD.

The last alias definition equates net name OUT to node name $N_0007.
.AUTOCONVERGE (Autoconvergence of simulations)

**Purpose**
Users can suggest relaxed limits for various options that PSpice can modify during a simulation to achieve convergence.

**General form**
```
.AUTOCONVERGE <option 1>=<relaxed value> <option 2>=<relaxed value> ...<option n>=<relaxed value>
<restart=0>
```

**Examples**
```
.autoconverge ITL1=1000 RELTOL=0.05 ABSTOL=.001
```

**Arguments and options**
- `<option n>`
  can be any one or more of the options ITL1, ITL2, ITL4, RELTOL, ABSTOL, VNTOL, or PIVTOL. PSpice will modify the specified options to achieve convergence of simulations.

- `<relaxed value>`
  The relaxed limit for the option.

- `<restart=0>`
  Restarts simulation with relaxed value from T=0 when convergence is not achieved with relaxed value at the end of the current simulation run.

**Comments**
The `<relaxed value>` must be more relaxed then the normal limit.
.CHKPT (Generate CheckPoints)

**Purpose**

Save the state of a transient simulation at different moments as CheckPoints. You can specify any of the CheckPoints as a restart point and rerun the simulation from that point using the .restart command.

**General form**

```
.chkp <Checkpoint_name> <time_interval_type> <time_interval_value> ][TSTEP <timestep>]
```

**Examples**

```
.chkp "D:/simdata/checkset1" SINT 1ms RINT 10min TP 2.5ms,7.5ms
```

**Arguments and options**

- `<checkpoint_name>`
  Specifies the name of the CheckPoint.

- `time_interval_type`
  Can be any one of:
  - SINT: Specifies simulation time interval
  - RINT: Specifies real time interval
  - TP: Specifies time points

- `time_interval_value`
  The time interval in seconds.

- `TSTEP <timestep>`
  Specifies the time step to be used, where:
  - 0 generates checkpoints closest to default time step of PSpice engine
  - 1 generates checkpoints at user specified time points.

- `<RESTART=n>`
  where RESTART=0 means CheckPoint Restart is off and RESTART=1 means it is on. RESTART=1 is the default.
.DC (DC analysis)

Purpose
The .DC command performs a linear, logarithmic, or nested DC sweep analysis on the circuit. The DC sweep analysis calculates the circuit’s bias point over a range of values for <sweep variable name>.

Sweep type
The sweep can be linear, logarithmic, or a list of values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIN</td>
<td>linear sweep</td>
<td>The sweep variable is swept linearly from the starting to the ending value.</td>
</tr>
<tr>
<td>OCT</td>
<td>sweep by octaves</td>
<td>Sweep by octaves. The sweep variable is swept logarithmically by octaves.</td>
</tr>
<tr>
<td>DEC</td>
<td>sweep by decades</td>
<td>Sweep by decades. The sweep variable is swept logarithmically by decades.</td>
</tr>
<tr>
<td>LIST</td>
<td>List of values</td>
<td>Use a list of values.</td>
</tr>
</tbody>
</table>

Linear sweep

General form
.DC [LIN] <sweep variable name>
+ <start value> <end value> <increment value>
+ [nested sweep specification]

Examples
.DC VIN -.25 .25 .05
.DC LIN I2 5mA -2mA 0.1mA
.DC VCE 0V 10V .5V IB 0mA 1mA 50uA
.DC RES RMOD(R) 0.9 1.1 .001

Arguments and options

<start value>
Can be greater or less than <end value>: that is, the sweep can go in either direction.

<increment value>
The step size. This value must be greater than zero.
Comments

The sweep variable is swept linearly from the starting to the ending value.

The keyword LIN is optional.

Logarithmic sweep

General form

```
.DC <logarithmic sweep type> <sweep variable name>
+ <start value> <end value> <points value>
+ [nested sweep specification]
```

Examples

```
.DC DEC NPN QFAST(IS) 1E-18 1E-14 5
```

Arguments and options

- `<logarithmic sweep type>`
  Must be specified as either DEC (to sweep by decades) or OCT (to sweep by octaves).

- `<start value>`
  Must be positive and less than `<end value>`.

- `<points value>`
  The number of steps per octave or per decade in the sweep. This value must be an integer.

Comments

Either OCT or DEC must be specified for the `<logarithmic sweep type>`.
Nested sweep

General form

```
.DC <sweep variable name> LIST <value>*
+[nested sweep specification]
```

Examples

```
.DC TEMP LIST 0 20 27 50 80 100 PARAM Vsupply 7.5 15 .5
```

Arguments and options

```
<sweep variable name>
```
After the DC sweep is finished, the value associated with
```
<sweep variable name>
```
is set back to the value it had before the sweep started. The following items can be used as sweep variables in a DC sweep:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>A name of an independent voltage or current source.</td>
<td>During the sweep, the source’s voltage or current is set to the sweep value.</td>
</tr>
<tr>
<td>Model Parameter</td>
<td>A model type and model name followed by a model parameter name in parenthesis.</td>
<td>The parameter in the model is set to the sweep value. The following model parameters cannot be (usefully) swept: L and W for the MOSFET device (use LD and WD as a work around), and any temperature parameters, such as TC1 and TC2 for the resistor.</td>
</tr>
<tr>
<td>Temperature</td>
<td>Use the keyword TEMP for &lt;sweep variable name&gt;.</td>
<td>Set the temperature to the sweep value. For each value in the sweep, all the circuit components have their model parameters updated to that temperature.</td>
</tr>
<tr>
<td>Global Parameter</td>
<td>Use the keyword PARAM, followed by the parameter name, for &lt;sweep variable name&gt;.</td>
<td>During the sweep, the global parameter’s value is set to the sweep value and all expressions are reevaluated.</td>
</tr>
</tbody>
</table>
Comments For a nested sweep, a second sweep variable, sweep type, start, end, and increment values can be placed after the first sweep. In the nested sweep example, the first sweep is the inner loop: the entire first sweep is performed for each value of the second sweep.

When using a list of values, there are no start and end values. Instead, the numbers that follow the keyword LIST are the values that the sweep variable is set to.

The rules for the values in the second sweep are the same as for the first. The second sweep generates an entire .PRINT (print) on page 84 table or .PLOT (plot) on page 82 plot for each value of the sweep. Probe displays nested sweeps as a family of curves.
.DISTRIBUTION (user-defined distribution)

Purpose
The .DISTRIBUTION command defines a user distribution for tolerances, and is only used with Monte Carlo and sensitivity/worst-case analyses. The curve described by a .DISTRIBUTION command controls the relative probability distribution of random numbers generated by PSpice to calculate model parameter deviations.

General form
DISTRIBUTION <name> (<deviation> <probability>)*

Examples
.DISTRIBUTION bi_modal (-1,1) (-.5,1) (-.5,0) (.5,0) + (.5,1) (1,1)
.DISTRIBUTION triangular (-1,0) (0,1) (1,0)

Arguments and options
(<deviation> <probability>)
Defines the distribution curve by pairs, or corner points, in a piecewise linear fashion. You can specify up to 100 value pairs.

<deviation>
Must be in the range (-1,+1), which matches the range of the random number generator. No <deviation> can be less than the previous <deviation> in the list, although it can repeat the previous value.

<probability>
Represents a relative probability, and must be positive or zero.

Comments
When using Schematics, several distributions can be defined by configuring an include file containing the .DISTRIBUTION command. For details on how to do this, refer to your PSpice User’s Guide.

If you are not using Schematics, a user-defined distribution can be specified as the default by setting the DISTRIBUTION parameter in the .OPTIONS (analysis options) command.
Deriving updated parameter values

The updated value of a parameter is derived from a combination of a random number, the distribution, and the tolerance specified. This method permits distributions which have different excursions in the positive and negative directions. It also allows the use of one distribution even if the tolerances of the components are different so long as the general shape of the distributions are the same.

1. Generate a <temporary random number> in the range (0, 1).

2. Normalize the area under the specified distribution.

3. Set the <final random number> to the point where the area under the normalized distribution equals the <temporary random number>.

4. Multiply this <final random number> by the specified tolerance.

Usage example

To illustrate, assume there is a 1.0 μfd capacitor that has a variation of -50% to +25%, and another that has tolerances of -10% to +5%. Note that both capacitors’ tolerances are in the same general shape, i.e., both have negative excursions twice as large as their positive excursions.

```
.distribution cdistrib (-1,1) (.5, 1) (.5, 0) (1, 0)
c1 1 0 cmod lu
c2 1 0 cmod2 lu
.model cmod1 cap (c=1 dev/cdistrib 50%)
.model cmod2 cap (c=1 dev/cdistrib 10%)
```

The steps taken for this example are as follows:

1. Generate a <temporary random value> of 0.3.

2. Normalize the area under the cdistrib distribution (1.5) to 1.0.

3. The <final random number> is therefore -0.55 (the point where the normalized area equals 0.3).

4. For c1, this -0.55 is then scaled by 50%, resulting in -0.275; for c2, it is scaled by 10%, resulting in -0.055.

**Note:** Separate random numbers are generated for each parameter that has a tolerance unless a tracking number is specified.
.DMFACTOR

Purpose  The .DMFACTOR command sets the relative factor for minimum delta. This value specifies the relative value by which the minimum time step size is changed.

General form  .DMFACTOR <value>

Arguments and options

<value>
any value that is a factor of 10 and is less than or equal to 1, such as .1, .001, or .0001

Examples

.DMFACTOR .1

In this example, if the default value of the minimum time step size is $10^{-18}$, with .DMFACTOR .1 the minimum time step value will be $10^{-19}$.
.END (end of circuit)

**Purpose**
The .END command marks the end of the circuit. All the data and every other command must come before it. When the .END command is reached, PSpice does all the specified analyses on the circuit.

**General form**
.END

**Examples**
* 1st circuit in file
  ... circuit definition
  .END
* 2nd circuit in file
  ... circuit definition
  .END

**Comments**
There can be more than one circuit in an input file. Each circuit is marked by an .END command. PSpice processes all the analyses for each circuit before going on to the next one.

Everything is reset at the beginning of each circuit. Having several circuits in one file gives the same results as having them in separate files and running each one separately. However, all the simulation results go into one .OUT file and one .DAT file. This is a convenient way to arrange a set of runs for overnight operation.

**Note:** The last statement in an input file must be an .END command.
.ENDS (end subcircuit)

Purpose The .ENDS command marks the end of a subcircuit definition (started by a .SUBCKT (subcircuit) statement.

General form .ENDS [subcircuit name]

Examples

.ENDS
.ENDS circuit_name

Comments It is a good practice to repeat the subcircuit name though this is not required.

For a detailed explanation see .SUBCKT (subcircuit) on page 105.
.EXTERNAL (external port)

Purpose
External ports are provided as a means of identifying and distinguishing those nets representing the outermost (or peripheral), connections to the circuit being simulated. The external port statement .EXTERNAL applies only to nodes that have digital devices attached to them.

General form
.EXTERNAL <attribute> <node-name>*

Examples
.EXTERNAL INPUT Data1, Data2, Data3
.EXTERNAL OUTPUT P1
.EXTERNAL BIDIRECTIONAL BPort1 BPort2 BPort3

Arguments and options

<attribute>
One of the keywords INPUT, OUTPUT, or BIDIRECTIONAL, describing the usage of the port.

<node_name>
One or more valid PSpice A/D node names.

Comments
When a node is included in a .EXTERNAL statement it is identified as a primary observation point. For example, if you are modeling and simulating a PCB-level description, you could place an .EXTERNAL (or its Capture symbol counterparts) on the edge pin nets to describe the pin as the external interface point of the network.

PSpice recognizes the nets marked as .EXTERNAL when reporting any sort of timing violation. When a timing violation occurs, PSpice analyzes the conditions that would permit the effects of such a condition to propagate through the circuit. If, during this analysis, a net marked external is encountered, PSpice reports the condition as a Persistent Hazard, signifying that it has a potential effect on the externally visible behavior of the circuit.

For more information on Persistent Hazards, refer to your PSpice User’s Guide.

Port specifications are inserted in the netlist by Capture whenever an external port symbol, EXTERNAL_IN, EXTERNAL_OUT, or EXTERNAL_BI is used. Refer to your PSpice User’s Guide for more information.
.FOUR (Fourier analysis)

**Purpose**
Fourier analysis decomposes the results of a transient analysis into Fourier components.

**General form**
```
.FOUR <frequency value> [no. harmonics value] <output variable>
```

**Examples**
- `.FOUR 10kHz V(5) V(6,7) I(VSENS3)`
- `.FOUR 60Hz 20 V(17)`
- `.FOUR 10kHz V([OUT1],[OUT2])`

**Arguments and options**
- `<output variable>`
  An output variable of the same form as in a `.PRINT (print)` command or `.PLOT (plot)` command for a transient analysis.

- `<frequency value>`
  The fundamental frequency. Not all of the transient results are used, only the interval from the end, back to `1/<frequency value>` before the end is used. This means that the transient analysis must be at least `1/<frequency value>` seconds long.

**Comments**
The analysis results are obtained by performing a Fourier integral on the results from a transient analysis. The analysis must be supplied with specified output variables using evenly spaced time points. The time interval used is `<print step value>` in the `.TRAN (transient analysis)` command, or 1% of the `<final time value>` (TSTOP) if smaller, and a 2nd-order polynomial interpolation is used to calculate the output value used in the integration. The DC component, the fundamental, and the 2nd through 9th harmonics of the selected voltages and currents are calculated by default, although more harmonics can be specified.

A .FOUR command requires a .TRAN command, but Fourier analysis does not require .PRINT, .PLOT, or .PROBE (Probe) commands. The tabulated results are written to the output file (.out) as the transient analysis is completed.

**Note:** The results of the .FOUR command are only available in the output file. They cannot be viewed in Probe.
.FUNC (function)

Purpose  The .FUNC command defines functions used in expressions. Besides their obvious flexibility, they are useful for where there are several similar sub expressions in a circuit file.

General form

```
.FUNC <name> ([arg]*) {<body>}
```

Examples

```
.FUNC E(x) {exp(x)}
.FUNC DECAY(CNST) {E(-CNST*TIME)}
.FUNC TRIWAV(x) {ACOS(COS(x))/3.14159}
.FUNC MIN3(A,B,C) {MIN(A,MIN(B,C))}
```

Arguments and options

`.FUNC`
Does not have to precede the first use of the function name. Functions cannot be redefined and the function name must not be the same as any of the predefined functions (e.g., SIN and SQRT). See Numeric expression conventions on page 13 for a list of valid expressions. These arguments cannot be node names.

`<body>`
Refers to other (previously defined) functions; the second example, DECAY, uses the first example, E.

`[arg]`
Specifies up to 10 arguments in a definition. The number of arguments in the use of a function must agree with the number in the definition. Functions can be defined as having no arguments, but the parentheses are still required. Parameters, TIME, other functions, and the Laplace variable s are allowed in the body of function definitions.

Comments

The `<body>` of a defined function is handled in the same way as any math expression; it is enclosed in curly braces {}. Previous versions of PSpice did not require this, so for compatibility the `<body>` can be read without braces, but a warning is generated.

Note: Creating a file of frequently used .FUNC definitions and accessing them using an .INC command near the beginning of the circuit file can be helpful. .FUNC commands can also be defined in subcircuits. In those cases they only have local scope.
.IC (initial bias point condition)

**Purpose**
The .IC command sets initial conditions for both small-signal and transient bias points. Initial conditions can be given for some or all of the circuit’s nodes.

.IC sets the initial conditions for the bias point only. It does not affect a .DC (DC analysis) sweep.

**General form**
```
.IC < V(<node> [,<node>])=<value> >*
.IC <I(<inductor>)=<value>>*
```

**Examples**
```
.IC V(2)=3.4 V(102)=0 V(3)=-1V I(L1)=2uAmp
.IC V(InPlus,InMinus)=1e-3 V(100,133)=5.0V
```

**Arguments and options**

<value>
A voltage assigned to <node> (or a current assigned to an inductor) for the duration of the bias point calculation.

**Comments**
The voltage between two nodes and the current through an inductor can be specified. During bias calculations, PSpice clamps the voltages to specified values by attaching a voltage source with a 0.0002 ohm series resistor between the specified nodes. After the bias point has been calculated and the transient analysis started, the node is released.

If the circuit contains both the .IC command and .NODESET (set approximate node voltage for bias point) command for the same node or inductor, the .NODESET command is ignored (.IC overrides .NODESET).

Refer to your *PSpice User’s Guide* for more information on setting initial conditions.

**Note:** An .IC command that imposes nonzero voltages on inductors cannot work properly, since inductors are assumed to be short circuits for bias point calculations. However, inductor currents can be initialized.
.INC (include file)

**Purpose**  The .INC command inserts the contents of another file.

**General form**  .INC <file name>

**Examples**  .INC "SETUP.CIR"
               .INC "C:\LIB\VCO.CIR"

**Arguments and options**

<file name>
Any character string that is a valid file name for your computer system.

**Comments**  Including a file is the same as bringing the file’s text into the circuit file. Everything in the included file is actually read in. The comments of the included file are then treated just as if they were found in the parent file.

Included files can contain any valid PSpice statements, with the following conditions:

- The included files should not contain title lines unless they are commented.
- Included files can be nested up to 4 levels.

**Note:** Every model and subcircuit definition, even if not needed, takes up memory.
.LIB (library file)

**Purpose**  The .LIB command references a model or subcircuit library in another file.

**General form**  .LIB [file_name]

**Examples**  
- .LIB
- .LIB linear.lib
- .LIB "C:\lib\bipolar.lib"

**Arguments and options**

[file_name]
Any character string that is a valid file name for the computer system.

**Comments**  Library files can contain any combination of the following:

- comments
- .MODEL (model definition) commands
- subcircuit definitions (including the .SUBCKT (subcircuit) command)
- .PARAM (parameter) commands
- .FUNC (function) commands
- .LIB commands

No other statements are allowed. For further discussion of library files, refer to your *PSpice User’s Guide*.

If [file_name] is left off, all references point to the master library file, nom.lib. When a library file is referenced in Schematics, PSpice first searches for the file in the current working directory, then searches in the directory specified by the LIBPATH variable (set in PSpice.ini).

When any library is modified, PSpice creates an index file based on the first use of the library. The index file is organized so that PSpice can find a particular .MODEL or .SUBCKT (subcircuit) quickly, despite the size of the library file.

**Note:** The index files have to be regenerated each time the library is changed. Because of this, it is advantageous to configure separately any frequently changed libraries.
Nom.lib normally contains references to all parts in the PSpice Standard Model Library. You can edit nom.lib to include your custom model references.
.LOADBIAS (load bias point file)

**Purpose**
The .LOADBIAS command loads the contents of a bias point file. It is helpful in setting initial bias conditions for subsequent simulations. However, the use of .LOADBIAS does not guarantee convergence.

**General form**
```plaintext
.LOADBIAS <file name>
```

**Examples**
```plaintext
.LOADBIAS "SAVETRAN.NOD"
.LOADBIAS "C:\PROJECT\INIT.FIL"
```

**Arguments and options**

<file name>
Any character string which is a valid computer system file name, but it must be enclosed in quotation marks.

**Comments**
Normally, the bias point file is produced by a previous circuit simulation using the .SAVEBIAS (save bias point to file) command.

The bias point file is a text file that contains one or more comment lines and a .NODESET (set approximate node voltage for bias point) command setting the bias point voltage or inductor current values. If a fixed value for a transient analysis bias point needs to be set, this file can be edited to replace the .NODESET command with an .IC (initial bias point condition) command.

**Note:** Any nodes mentioned in the loaded file that are not present in the circuit are ignored, and a warning message will be generated.

To echo the .LOADBIAS file contents to the output file, use the EXPAND option on the .OPTIONS (analysis options) command.
**.MC (Monte Carlo analysis)**

**Purpose**
The .MC command causes a Monte Carlo (statistical) analysis of the circuit and causes PSpice to perform multiple runs of the selected analysis (DC, AC, or transient).

**General form**
```
.MC <#runs value> <analysis> <output variable> <function>
[option]* SEED=value
```

**Examples**
```
.MC 10 TRAN V(5) YMAX
.MC 50 DC IC(Q7) YMAX LIST
.MC 20 AC VP(13,5) YMAX LIST OUTPUT ALL
.MC 10 TRAN V([OUT1],[OUT2]) YMAX SEED=9321
```

**Arguments and options**

<#runs value>
The total number of runs to be performed (for printed results the upper limit is 2,000, and for results to be viewed in Probe, the limit is 400).

<analysis>
Specifies at least one analysis type: .DC (DC analysis), Table on page 32, or .TRAN (transient analysis). This analysis is repeated in subsequent passes. All analyses that the circuit contains are performed during the nominal pass. Only the selected analysis is performed during subsequent passes.

<output variable>
Identical in format to that of a .PRINT (print) output variable.

<function>
Specifies the operation to be performed on the values of <output variable> to reduce these to a single value. This value is the basis for the comparisons between the nominal and subsequent runs. The <function> can be any one of the following:

<table>
<thead>
<tr>
<th>Function</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>YMAX</td>
<td>Find the absolute value of the greatest difference in each waveform from the nominal run.</td>
</tr>
<tr>
<td>MAX</td>
<td>Find the maximum value of each waveform.</td>
</tr>
</tbody>
</table>
### Function | Definition
--- | ---
MIN | Find the minimum value of each waveform.
RISE_EDGE(<value>) | Find the first occurrence of the waveform crossing above the threshold <value>. The waveform must have one or more points at or below <value> followed by one above; the output value listed is the first point that the waveform increases above <value>.
FALL_EDGE(<value>) | Find the first occurrence of the waveform crossing below the threshold <value>. The waveform must have one or more points at or above <value> followed by one below; the output value listed is where the waveform decreases below <value>.
**Note:** `<function>` and all `[option]s (except for `<output type>`) have no effect on the Probe data that is saved from the simulation. They are only applicable to the output file.

```
[option]*
Can include zero or more of the following options:
```

<table>
<thead>
<tr>
<th>Option</th>
<th>Definition</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST</td>
<td>Lists, at the beginning of each run, the model parameter values actually used for each component during that run.</td>
<td></td>
</tr>
<tr>
<td>OUTPUT</td>
<td>Asks for an output from subsequent runs, after the nominal (first) run. The output from any run is governed by a .PRINT, .PLOT, and .PROBE command in the file. If OUTPUT is omitted, then only the nominal run produces output. The <code>&lt;output type&gt;</code> is one of the ones shown in the examples to the right.</td>
<td></td>
</tr>
<tr>
<td>ALL</td>
<td>Forces all output to be generated (including the nominal run).</td>
<td></td>
</tr>
<tr>
<td>FIRST &lt;N&gt;</td>
<td>Generates output only during the first n runs.</td>
<td></td>
</tr>
<tr>
<td>EVERY &lt;N&gt;</td>
<td>Generates output every n&lt;sup&gt;th&lt;/sup&gt; run.</td>
<td></td>
</tr>
<tr>
<td>RUNS &lt;N&gt;*</td>
<td>Does analysis and generates output only for the listed runs. Up to 25 values can be specified in the list.</td>
<td></td>
</tr>
<tr>
<td>RANGE&lt;sup&gt;1&lt;/sup&gt; (&lt;low value&gt;, &lt;high value&gt;)</td>
<td>Restricts the range over which <code>&lt;function&gt;</code> is evaluated. An asterisk (*) can be used in place of a <code>&lt;value&gt;</code> to show for all values.</td>
<td></td>
</tr>
<tr>
<td>YMAX RANGE(*, .5)</td>
<td>YMAX is evaluated for values of the sweep variable (e.g., time and frequency) of .5 or less.</td>
<td></td>
</tr>
<tr>
<td>MAX RANGE(-1,*)</td>
<td>The maximum of the output variable is found for values of the sweep variable of -1 or more.</td>
<td></td>
</tr>
</tbody>
</table>

---

1. If RANGE is omitted, then `<function>` is evaluated over the whole sweep range. This is equivalent to RANGE(*, *).
[SEED=value]
Defines the seed for the random number generator within the Monte Carlo analysis (The Art of Computer Programming, Donald Knuth, vol. 2, pg. 171, “subtractive method”).

<Value>
Must be an odd integer ranging from 1 to 32,767. If the seed value is not set, its default value is 17,533.

Note: For almost all analyses, the default seed value is adequate to achieve a constant set of results. The seed value can be modified within the integer value as required.

Comments
The first run uses nominal values of all components. Subsequent runs use variations on model parameters as specified by the DEV and LOT tolerances on each .MODEL (model definition) parameter.

The other specifications on the .MC command control the output generated by the Monte Carlo analysis.

For more information on Monte Carlo analysis, refer to your PSpice User’s Guide.
.MODEL (model definition)

Purpose
The .MODEL command defines a set of device parameters that can be referenced by devices in the circuit.

General form
```
.MODEL <model name> [AKO: <reference model name>]
+ <model type>
+ ([<parameter name> = <value> [tolerance specification]]*
+ [T_MEASURED=<value>] [[T_ABS=<value>] or
+ [T_REL_GLOBAL=<value>] or [T_REL_LOCAL=<value>]]
```

Examples
```
.MODEL RMAX RES (R=1.5 TC1=.0002 TC2=.005)
.MODEL DNOM D (IS=1E-9)
.MODEL QDRIV NPN (IS=1E-7 BF=30)
.MODEL MLOAD NMOS (LEVEL=1 VTO=.7 CJ=.02pF)
.MODEL CMOD CAP (C=1 DEV 5%)
.MODEL DLOAD D (IS=1E-9 DEV .5% LOT 10%)
.MODEL RTRACK RES (R=1 DEV/GAUSS 1% LOT/UNIFORM 5%)
.MODEL QDR2 AKO:QDRIV NPN (BF=50 IKF=50m)
```

Arguments and options

`<model name>`
The model name which is used to reference a particular model.

`<reference model name>`
The model types of the current model and the AKO (A Kind Of) reference model must be the same. The value of each parameter of the referenced model is used unless overridden by the current model, e.g., for QDR2 in the last example, the value of IS derives from QDRIV, but the values of BF and IKF come from the current definition. Parameter values or formulas are transferred, but not the tolerance specification. The referenced model can be in the main circuit file, accessed through a .INC command, or it can be in a library file; see .LIB (library file).

`<model type>`
Must be one of the types outlined in the table that follows.

Devices can only reference models of a corresponding type; for example:

- A JFET can reference a model of types NJF or PJF, but not of type NPN.
- There can be more than one model of the same type in a circuit, although they must have different names.
Following the `<model type>` is a list of parameter values enclosed by parentheses. None, any, or all of the parameters can be assigned values. Default values are used for all unassigned parameters. The lists of parameter names, meanings, and default values are found in the individual device descriptions.
<table>
<thead>
<tr>
<th>Model type</th>
<th>Instance name</th>
<th>Type of device</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAP</td>
<td>Cxxx</td>
<td>capacitor</td>
</tr>
<tr>
<td>CORE</td>
<td>Kxxx</td>
<td>nonlinear, magnetic core (transformer)</td>
</tr>
<tr>
<td>D</td>
<td>Dxxx</td>
<td>diode</td>
</tr>
<tr>
<td>DINPUT</td>
<td>Nxxx</td>
<td>digital input device (receive from digital)</td>
</tr>
<tr>
<td>DOUTPUT</td>
<td>Oxxx</td>
<td>digital output device (transmit to digital)</td>
</tr>
<tr>
<td>GASFET</td>
<td>Bxxx</td>
<td>N-channel GaAs MESFET</td>
</tr>
<tr>
<td>IND</td>
<td>Lxxx</td>
<td>inductor</td>
</tr>
<tr>
<td>ISWITCH</td>
<td>Wxxx</td>
<td>current-controlled switch</td>
</tr>
<tr>
<td>LPNP</td>
<td>Qxxx</td>
<td>lateral PNP bipolar transistor</td>
</tr>
<tr>
<td>NIGBT</td>
<td>Zxxx</td>
<td>N-channel insulated gate bipolar transistor (IGBT)</td>
</tr>
<tr>
<td>NJF</td>
<td>Jxxx</td>
<td>N-channel junction FET</td>
</tr>
<tr>
<td>NMOS</td>
<td>Mxxx</td>
<td>N-channel MOSFET</td>
</tr>
<tr>
<td>NPN</td>
<td>Qxxx</td>
<td>NPN bipolar transistor</td>
</tr>
<tr>
<td>PJF</td>
<td>Jxxx</td>
<td>P-channel junction FET</td>
</tr>
<tr>
<td>PMOS</td>
<td>Mxxx</td>
<td>P-channel MOSFET</td>
</tr>
<tr>
<td>PNP</td>
<td>Qxxx</td>
<td>PNP bipolar transistor</td>
</tr>
<tr>
<td>RES</td>
<td>Rxxx</td>
<td>resistor</td>
</tr>
<tr>
<td>TRN</td>
<td>Txxx</td>
<td>lossy transmission line</td>
</tr>
<tr>
<td>UADC</td>
<td>Uxxx</td>
<td>multi-bit analog-to-digital converter</td>
</tr>
<tr>
<td>UDAC</td>
<td>Uxxx</td>
<td>multi-bit digital-to-analog converter</td>
</tr>
<tr>
<td>UDLY</td>
<td>Uxxx</td>
<td>digital delay line</td>
</tr>
<tr>
<td>UEFF</td>
<td>Uxxx</td>
<td>edge-triggered flip-flop</td>
</tr>
<tr>
<td>UGATE</td>
<td>Uxxx</td>
<td>standard gate</td>
</tr>
<tr>
<td>UGFF</td>
<td>Uxxx</td>
<td>gated flip-flop</td>
</tr>
<tr>
<td>UIO</td>
<td>Uxxx</td>
<td>digital I/O model</td>
</tr>
<tr>
<td>UTGATE</td>
<td>Uxxx</td>
<td>tristate gate</td>
</tr>
<tr>
<td>VSWITCH</td>
<td>Sxxx</td>
<td>voltage-controlled switch</td>
</tr>
</tbody>
</table>
[tolerance specification]

Appended to each parameter, using the format:

```
[DEV [track&dist] <value>[%]] [LOT [track&dist] <value>[%]]
```

to specify an individual device (DEV) and the device lot (LOT) parameter value deviations. The tolerance specification is used by the .MC (Monte Carlo analysis) analysis only.

The LOT tolerance requires that all devices that refer to the same model use the same adjustments to the model parameter. DEV tolerances are independent, that is each device varies independently. The % shows a relative (percentage) tolerance. If it is omitted, <value> is in the same units as the parameter itself.

[track & dist]

Specifies the tracking and non-default distribution, using the format:

```
[/<lot #>][/<distribution name>]
```

These specifications must immediately follow the keywords DEV and LOT (without spaces) and are separated by /.

<lot #>

Specifies which of ten random number generators, numbered 0 through 9, are used to calculate parameter value deviations. This allows deviations to be correlated between parameters in the same model, as well as between models. The generators for DEV and LOT tolerances are distinct: there are ten generators for DEV tracking and ten generators for LOT tracking. Tolerances without <lot #> are assigned individually generated random numbers.

<distribution name>

The distribution name is one of the following. The default distribution can be set by using the DISTRIBUTION parameter of the .OPTIONS (analysis options) command.

<table>
<thead>
<tr>
<th>Distribution name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNIFORM</td>
<td>Generates uniformly distributed deviations over the range $\pm$&lt;value&gt;.</td>
</tr>
</tbody>
</table>
Generates deviations using a Gaussian distribution over the range $\pm 3\sigma$ and `<value>` specifies the $\pm 1\sigma$ deviation (i.e., this generates deviations greater than $\pm <value>$).

Generates deviations using a user-defined distribution and `<user name>` specifies the $\pm 1$ deviation in the user definition; see the `.DISTRIBUTION` (user-defined distribution).

Generates deviations with default distribution of $\pm 1\sigma$. The default distribution can be changed using the `.OPTIONS` command as follows:

- `.OPTIONS DISTRIBUTION GAUSS_USER`
- `.OPTIONS USER_SIGMA_VAL=<value>`

Where, `<value>` can be any integer from 1 to 9.

**Comments**

The examples are for the `.MODEL` parameter. The last example uses the AKO syntax to reference the parameters of the model QDRIV in the third example.

For more information, refer to your *PSpice User’s Guide*. 

<table>
<thead>
<tr>
<th>Distribution name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAUSS</td>
<td>Generates deviations using a Gaussian distribution over the range $\pm 3\sigma$ and <code>&lt;value&gt;</code> specifies the $\pm 1\sigma$ deviation (i.e., this generates deviations greater than $\pm &lt;value&gt;$).</td>
</tr>
<tr>
<td><code>&lt;user name&gt;</code></td>
<td>Generates deviations using a user-defined distribution and <code>&lt;value&gt;</code> specifies the $\pm 1$ deviation in the user definition; see the <code>.DISTRIBUTION</code> (user-defined distribution).</td>
</tr>
<tr>
<td>GAUSS_USER</td>
<td>Generates deviations with default distribution of $\pm 1\sigma$. The default distribution can be changed using the <code>.OPTIONS</code> command as follows:</td>
</tr>
<tr>
<td></td>
<td><code>.OPTIONS DISTRIBUTION GAUSS_USER</code></td>
</tr>
<tr>
<td></td>
<td><code>.OPTIONS USER_SIGMA_VAL=&lt;value&gt;</code></td>
</tr>
<tr>
<td></td>
<td>Where, <code>&lt;value&gt;</code> can be any integer from 1 to 9.</td>
</tr>
</tbody>
</table>
Parameters for setting temperature

Some passive and semiconductor devices (C, L, R, B, D, J, M, and Q) have two levels of temperature attributes that can be customized on a model-by-model basis.

First, the temperature at which the model parameters were measured can be defined by using one of the following model parameter formats in the .MODEL command line:

\[
\begin{align*}
\text{T\_MEASURED} & = \text{<literal value>} \\
\text{T\_MEASURED} & = \{ \text{<parameter>} \}
\end{align*}
\]

This overrides the nominal $T_{\text{NOM}}$ value which is set in the .OPTIONS (analysis options) command line (default = $27^\circ \text{C}$). All other parameters listed in the .MODEL command are assumed to have been measured at $T_{\text{MEASURED}}$.

In addition to the measured model parameter temperature, current device temperatures can be customized to override the circuit’s global temperature specification defined by the .TEMP (temperature) command line (or equivalent .STEP TEMP or .DC TEMP). There are three forms, as described below.

### Table 1-1 Model parameters for device temperature

<table>
<thead>
<tr>
<th>Description</th>
<th>.MODEL format</th>
<th>Parameter format</th>
<th>Referencing device temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>absolute temperature</td>
<td>standard</td>
<td>$T_{\text{ABS}}$=$&lt;$\text{value}$&gt;$</td>
<td>$T_{\text{ABS}}$</td>
</tr>
<tr>
<td>relative to current</td>
<td>standard</td>
<td>$T_{\text{REL_GLOBAL}}$=$&lt;$\text{value}$&gt;$</td>
<td>$\text{global temperature}$ + $T_{\text{REL_GLOBAL}}$</td>
</tr>
<tr>
<td>temperature</td>
<td>AKO</td>
<td>$T_{\text{REL_LOCAL}}$=$&lt;$\text{value}$&gt;$</td>
<td>$T_{\text{ABS}}(\text{AKO Model})$ + $T_{\text{REL_LOCAL}}$</td>
</tr>
</tbody>
</table>

For all formats, $<$\text{value}$>$ can be a literal value or a parameter of the form \{<\text{parameter name}>\}. A maximum of one device temperature customization can coexist using the T\_MEASURED customization. For example,

\[.\text{MODEL } \text{PNP\_NEW PNP} \{ \text{T\_ABS}=35 \text{ T\_MEASURED}=0 \text{ BF}=90 \}\]

defines a new model PNP\_NEW, where BF was measured at $0^\circ \text{C}$. Any bipolar transistor referencing this model has an absolute device temperature of $35^\circ \text{C}$. 
Examples

One

This example demonstrates device temperatures set relative to the global temperature of the circuit:

```plaintext
.TEMP 10 30 40
.MODEL PNP_NEW PNP ( T_REL_GLOBAL=-5 BF=90 )
```

This produces three PSpice runs where global temperature changes from 10° to 30° to 40°C, respectively, and any bipolar transistor that references the PNP_NEW model has a device temperature of 5°, 25°, or 35°C, respectively.

Two

This example sets the device temperature relative to a referenced AKO model:

```plaintext
.MODEL PNP_NEW  AKO: PNP_OLD PNP T_REL_LOCAL=10
```

Any bipolar transistor referencing the PNP_NEW model has a device temperature of 30°C.

Special considerations

There are a few special considerations when using these temperature parameters:

- If the technique for current device temperature is using the value relative to an AKO model's absolute temperature (T_ABS), and the AKO referenced model does not specify T_ABS, then the T_REL_LOCAL specification is ignored and the standard global temperature specification is used.

- These temperature parameters cannot be used with the DEV and LOT model parameter tolerance feature.

- A DC sweep analysis can be performed on these parameters so long as the temperature parameter assignment is to a variable parameter. For example:

```plaintext
.PARAM PTEMP 27
.MODEL PNP_NEW PNP ( T_ABS={PTEMP} )
.DC PARAM PTEMP 27 35 1
```

This method produces a single DC sweep in PSpice where any bipolar transistor referencing the PNP_NEW model has a device temperature which is swept from 27°C to 35°C in 1°C increments.

A similar effect can be obtained by performing a parametric analysis. For instance:
This method produces nine PSpice runs where the PNP_NEW model temperature steps from
27°C to 35°C in increments of 1°C, one step per run.

The effect of a temperature parameter is evaluated once prior to the bias point
calculation, unless parameters are swept by means of a .DC PARAM or .STEP PARAM
analysis described above. In these cases, the temperature parameter’s effect is
reevaluated once for each value of the swept variable.
.NODESET (set approximate node voltage for bias point)

Purpose
The .NODESET command helps calculate the bias point by providing an initial best guess for some node voltages and/or inductor currents. Some or all of the circuit’s node voltages and inductor currents can be given the initial guess, and in addition, the voltage between two nodes can be specified.

General form

```
.NODESET < V(<node> [,<node>])=value >*
.NODESET <I(<inductor>)>=value>
```

Examples

```
.NODESET V(2)=3.4 V(102)=0 V(3)=-1V I(L1)=2uAmp
.NODESET V(InPlus,InMinus)=1e-3 V(100,133)=5.0V
```

Comments
This command is effective for the bias point (both small-signal and transient bias points) and for the first step of the DC sweep. It has no effect during the rest of the DC sweep, nor during a transient analysis.

Unlike the .IC (initial bias point condition) command, .NODESET provides only an initial guess for some initial values. It does not clamp those nodes to the specified voltages. However, by providing an initial guess, .NODESET can be used to break the tie in, for instance, a flip-flop, and make it come up in a required state.

If both the .IC command and .NODESET command are present, the .NODESET command is ignored for the bias point calculations (.IC overrides .NODESET).

Note: For Capture-based designs, refer to your PSpice User’s Guide for more information on setting initial conditions.
.NOISE (noise analysis)

Purpose
The .NOISE command performs a noise analysis of the circuit.

General form
.NOISE V(<node> [,<node>]) <name> [interval value]

Examples
.NOISE V(5)   VIN
.NOISE V(101) VSRC 20
.NOISE V(4,5) ISRC
.NOISE V([OUT1]) V1
.NOISE V([OUT1],[OUT2]) V1

Note: For the node names starting with alphabets, square brackets should be used as in the last example.

Arguments and options

V(<node> [,<node>])
Output voltage. It has a form such as V(5), which is the voltage at the output node five, or a form such as V(4,5), which is the output voltage between two nodes four and five.

<name>
The name of an independent voltage or current source where the equivalent input noise is calculated. The <name> is not itself a noise generator, but only a place where the equivalent input noise is calculated.

[interval value]
Integer that specifies how often the detailed noise analysis data is written to the output file.
Comments

A noise analysis is performed in conjunction with an AC sweep analysis and requires an `.AC (AC analysis)` command. When `.NOISE` is used, noise data is recorded in the Probe `.DAT` file for each frequency in the AC sweep.

The simulator computes:

- Device noise for every resistor and semiconductor in the circuit (propagated to a specified output node)
- Total input and output noise

At each frequency, each noise generator’s contribution is calculated and propagated to the output node. At that point, all the propagated noise values are RMS-summed to calculate the total output noise. The gain from the input source to the output voltage, the total output noise, and the equivalent input noise are all calculated.

For more information, refer to the AC Analyses chapter of your PSpice User’s Guide.

If:

- `<name>` is a voltage source

then:

the input noise units are volt/hertz\(^{1/2}\)

If:

- `<name>` is a current source

then:

the input noise units are amp/hertz\(^{1/2}\)

The output noise units are always volt/hertz\(^{1/2}\).

Every nth frequency, where n is the print interval, a detailed table is printed showing the individual contributions of all the circuit’s noise generators to the total noise. These values are the noise amounts propagated to the output node, not the noise amounts at each generator. If `[interval value]` is not present, then no detailed table is printed.

The detailed table is printed while the analysis is being performed and does not need a `.PRINT (print)` command or a `.PLOT (plot)` command. The output noise and equivalent input noise can be printed in the output by using a `.PRINT` command or a `.PLOT` command.
.RESTART (Restart simulation from a CheckPoint)

Purpose
Restarts a simulation from specified checkpoint instead of rerunning a simulation from the start. CheckPoints are defined using the .CHKPT command.

General form
.restart <checkPoint_name> <state_number> [0/1]

Examples
.restart "D:/simdata/checkset1" state20

Arguments and options

<checkPoint_name>
Name of the CheckPoint to be used.

<state_number>
The specific state from which to restart the simulation.

<0/1>
Specifies the time step to be used, where:

❑ 0 generates checkpoints closest to default time step of PSpice engine
❑ 1 generates checkpoints at user specified time points.
.OP (bias point)

Purpose
The .OP command causes detailed information about the bias point to be printed.

General form
.OP

Examples
.OP

Comments
This command does not write output to the Probe data file. The bias point is calculated regardless of whether there is a .OP command. Without the .OP command, the only information about the bias point in the output is a list of the node voltages, voltage source currents, and total power dissipation.

Using a .OP command can cause the small-signal (linearized) parameters of all the nonlinear controlled sources and all the semiconductor devices to be printed in the output file.

The .OP command controls the output for the regular bias point only. The .TRAN (transient analysis) command controls the output for the transient analysis bias point.

Note: If no other analysis is performed, then no Probe data file is created.
.OPTIONS (analysis options)

Purpose
The .OPTIONS command is used to set all the options, limits, and control parameters for the simulator.

General form
.OPTIONS [option name]* [ <option name>=<value> ]*

Examples
.OPTIONS NOECHO NOMOD DEFL=12u DEFW=8u DEFAD=150p
+ DEFAS=150p
.OPTIONS ACCT RELTOL=.01
.OPTIONS DISTRIBUTION=GAUSS
.OPTIONS DISTRIBUTION=USERDEF1

Comments
The options can be listed in any order. There are two kinds of options: those with values, and those without values. Options without values are flags that are activated by simply listing the option name.

The .OPTIONS command is cumulative. That is, if there are two (or more) of the .OPTIONS command, the effect is the same as if all the options were listed together in one .OPTIONS command. If the same option is listed more than once, only its last value is used.

For SPICE options not available in PSpice, see Differences between PSpice and Berkeley SPICE2 on page 128.

Flag options

The default for any flag option is off or no (i.e., the opposite of specifying the option). Flag options affect the output file unless otherwise specified.

<table>
<thead>
<tr>
<th>Flag option</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCT</td>
<td>Summary and accounting information is printed at the end of all the analyses (refer to your PSpice User’s Guide for further information on ACCT).</td>
</tr>
<tr>
<td>ADVCONV</td>
<td>Enables all convergence algorithms, such as Pseudo Tran, STEPGMIN, and step sources. ON by default.</td>
</tr>
<tr>
<td>NOSTEPDEP</td>
<td>Suppresses stepping of dependent sources during source stepping.</td>
</tr>
<tr>
<td>Flag option,</td>
<td>Meaning, continued</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>DMFACTOR</td>
<td>Sets the relative factor for minimum delta. The value specifies the relative value</td>
</tr>
<tr>
<td></td>
<td>by which the minimum time step size is changed. The value should be less than or</td>
</tr>
<tr>
<td></td>
<td>equal to 1 and a factor of 10, such as 0.1, 0.001, or 0.0001.</td>
</tr>
<tr>
<td>EXPAND</td>
<td>Lists devices created by subcircuit expansion and lists contents of the bias point</td>
</tr>
<tr>
<td></td>
<td>file; see <code>.SAVEBIAS</code> (save bias point to file) and <code>.LOADBIAS</code> (load bias point</td>
</tr>
<tr>
<td></td>
<td>file).</td>
</tr>
<tr>
<td>GMINSRC</td>
<td>Enables step GMIN inside source-stepping</td>
</tr>
<tr>
<td>LIBRARY</td>
<td>Lists lines used from library files.</td>
</tr>
<tr>
<td>LIST</td>
<td>Lists a summary of the circuit elements (devices).</td>
</tr>
<tr>
<td>NOBIAS</td>
<td>Suppresses the printing of the bias point node voltages.</td>
</tr>
<tr>
<td>NODE</td>
<td>Lists a summary of the connections (node table).</td>
</tr>
<tr>
<td>NOECHO</td>
<td>Suppresses a listing of the input file(s).</td>
</tr>
<tr>
<td>NOGMINI</td>
<td>Specifies not to add GMIN across current sources.</td>
</tr>
</tbody>
</table>
**Flag option, Meaning, continued**

<table>
<thead>
<tr>
<th>Flag option</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>NOICTRANSLATE</td>
<td>Suppresses the translation of initial conditions (IC attributes) specified on capacitors and inductors into .IC statements (IC pseudocomponents). This means that IC attributes are ignored if the keyword Skip Bias Point (SKIPBP) is not put at the end of the .TRAN statement. See .TRAN (transient analysis).</td>
</tr>
<tr>
<td>NOMOD</td>
<td>Suppresses listing of model parameters and temperature updated values.</td>
</tr>
<tr>
<td>NOOPITER</td>
<td>Switches off source stepping.</td>
</tr>
<tr>
<td>NOOUTMSG</td>
<td>Suppresses simulation error messages in output file.</td>
</tr>
<tr>
<td>NOPAGE</td>
<td>Suppresses paging and the banner for each major section of output.</td>
</tr>
<tr>
<td>NOPRBMSG</td>
<td>Suppresses simulation error messages in Probe data file.</td>
</tr>
<tr>
<td>NOREUSE</td>
<td>Suppresses the automatic saving and restoring of bias point information between different temperatures, Monte Carlo runs, worst-case runs, and .STEP (parametric analysis). See also .SAVEBIAS (save bias point to file) and .LOADBIAS (load bias point file).</td>
</tr>
<tr>
<td>OPTS</td>
<td>Lists values for all options.</td>
</tr>
<tr>
<td>PSEUDOTRAN</td>
<td>Uses Pseudo-Transient Method.</td>
</tr>
<tr>
<td>STEPGMIN</td>
<td>Enables GMIN stepping. This causes a GMIN stepping algorithm to be applied to circuits that fail to converge. GMIN stepping is applied first, and if that fails, the simulator falls back to supply stepping.</td>
</tr>
<tr>
<td>BRKDEPSRC</td>
<td>Sets automatic break-points for behavioral sources</td>
</tr>
</tbody>
</table>

**Option with a name as its value**

The following option has a name as its value.

<table>
<thead>
<tr>
<th>Option</th>
<th>Meaning</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISTRIBUTION</td>
<td>default distribution for Monte Carlo deviations</td>
<td>UNIFORM</td>
</tr>
</tbody>
</table>
Default distribution values

The default distribution is used for all of the deviations throughout the Monte Carlo analyses, unless specifically overridden for a particular tolerance. The default value for the default distribution is UNIFORM, but can also be set to GAUSS or to a user-defined (\textit{<user name>}) distribution. If a user-defined distribution is selected (as illustrated in the last example under \textbf{OPTIONS} (analysis options)), a \textbf{.DISTRIBUTION} (user-defined distribution) command must be included in the circuit file to define the user distribution for the tolerances. An example would be:

\texttt{.DISTRIBUTION USERDEF1 (-1,1) (.5,1) (.5,0) (1,0)}

\texttt{.OPTIONS DISTRIBUTION=USERDEF1}

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABSTOL\textsuperscript{1}</td>
<td>best accuracy of currents</td>
<td>amp</td>
<td>1.0 pA</td>
</tr>
<tr>
<td>CHGTOL</td>
<td>best accuracy of charges</td>
<td>coulomb</td>
<td>0.01 pC</td>
</tr>
<tr>
<td>CPTIME\textsuperscript{2}</td>
<td>CPU time allowed for this run</td>
<td>sec</td>
<td>0.0\textsuperscript{3}</td>
</tr>
<tr>
<td>DEFFAD</td>
<td>MOSFET default drain area (AD).</td>
<td>meter\textsuperscript{2}</td>
<td>0.0</td>
</tr>
<tr>
<td>DEFFAS</td>
<td>MOSFET default source area (AS).</td>
<td>meter\textsuperscript{2}</td>
<td>0.0</td>
</tr>
<tr>
<td>DEFL</td>
<td>MOSFET default length (L).</td>
<td>meter</td>
<td>100.0 u</td>
</tr>
<tr>
<td>DEFW</td>
<td>MOSFET default width (W).</td>
<td>meter</td>
<td>100.0 u</td>
</tr>
<tr>
<td>DIGFREQ</td>
<td>minimum digital time step is \textbf{1/DIGFREQ}</td>
<td>hertz</td>
<td>10.0 GHz</td>
</tr>
<tr>
<td>DIGDRVVF</td>
<td>minimum drive resistance (Input/Output UIO type model, \textbf{DRVH} (high) and \textbf{DRVL} (low) values)</td>
<td>ohm</td>
<td>2.0</td>
</tr>
<tr>
<td>DIGDRVVZ</td>
<td>maximum drive resistance (UIO type model, \textbf{DRVH} and \textbf{DRVL} values)</td>
<td>ohm</td>
<td>20K</td>
</tr>
<tr>
<td>DIGERRDEFAULT</td>
<td>default error limit per digital constraint device</td>
<td></td>
<td>20.0</td>
</tr>
<tr>
<td>DIGERRLIMIT</td>
<td>maximum digital error message limit</td>
<td></td>
<td>0\textsuperscript{3}</td>
</tr>
<tr>
<td>DIGINITSTATE</td>
<td>sets initial state of all flip-flops and latches in circuit: 0=clear, 1=set, 2=X</td>
<td></td>
<td>2.0</td>
</tr>
<tr>
<td>DIGIOLVL</td>
<td>default digital I/O level: 1-4; see UIO in \textbf{.MODEL} (model definition) on page 58</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>DIGMNTYMX\textsuperscript{4}</td>
<td>default delay selector: 1=min, 2-typical, 3=max, 4=min/max</td>
<td></td>
<td>2.0</td>
</tr>
</tbody>
</table>
## PSpice Reference Guide
### Commands

<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIGMNTYSCALE</td>
<td>scale factor used to derive minimum delays from typical delays</td>
<td></td>
<td>0.4</td>
</tr>
<tr>
<td>DIGOVRDVRV</td>
<td>ratio of drive resistances required to allow one output to override another driving the same node</td>
<td></td>
<td>3.0</td>
</tr>
<tr>
<td>DIGTYMXS SCALE</td>
<td>scale factor used to derive maximum delays from typical delays</td>
<td></td>
<td>1.6</td>
</tr>
<tr>
<td>GMIN</td>
<td>minimum conductance used for any branch</td>
<td>ohm⁻¹</td>
<td>1.0E-12</td>
</tr>
<tr>
<td>GMINSTEPS</td>
<td>the GMIN stepping size in integer (any positive value). Set to 0 for engine default.</td>
<td></td>
<td>Same as ITL1</td>
</tr>
<tr>
<td>ITL1</td>
<td>DC and bias point blind repeating limit (the first evaluation of the operating point of the system)</td>
<td></td>
<td>150.0</td>
</tr>
<tr>
<td>ITL2</td>
<td>DC and bias point educated guess repeating limit (DC transfer curve iteration limit)</td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>ITL4</td>
<td>the limit at any repeating point in transient analysis</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>ITL5</td>
<td>total repeating limit for all points for transient analysis</td>
<td></td>
<td>0.0³</td>
</tr>
<tr>
<td></td>
<td>(ITL5=0 means ITL5=infinity)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ITL6</td>
<td>the number of steps of the source stepping algorithm. Can have any positive integer value. Set to 0 for engine default.</td>
<td></td>
<td>Same as ITL1</td>
</tr>
<tr>
<td>LIMIT</td>
<td>the absolute voltage limit. The default, 0, specifies that there is no limit on data values. You can modify it to a large value, such as 1e12, to eliminate overflow errors, especially when using exponential sources.</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>LIMPTS</td>
<td>maximum points allowed for any print table or plot (LIMPTS=0 means LIMPTS=infinity)</td>
<td></td>
<td>0.0³</td>
</tr>
</tbody>
</table>

| method            | integration method (values can be either TRAPEZOIDAL or GEAR)              |                |               |

October 2012 75 Product Version 16.6
<table>
<thead>
<tr>
<th>Options</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>NUMDGT</td>
<td>number of digits output in print tables (maximum of 8 useful digits)</td>
<td></td>
<td>4.0</td>
</tr>
<tr>
<td>PIVREL²</td>
<td>relative magnitude required for pivot in matrix solution</td>
<td></td>
<td>1.0E-3</td>
</tr>
<tr>
<td>PIVTOL²</td>
<td>absolute magnitude required for pivot in matrix solution</td>
<td></td>
<td>1.0E-13</td>
</tr>
<tr>
<td>PTRANSTEP</td>
<td>number of steps for a pseudo transient analysis to find the operating point.</td>
<td></td>
<td>Same as ITL1</td>
</tr>
<tr>
<td>RELTOL¹</td>
<td>relative accuracy of V and I</td>
<td></td>
<td>0.001</td>
</tr>
<tr>
<td>SOLVER⁵</td>
<td>performance package solution algorithm</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>(Solver = 0 selects the original solution algorithm;</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Solver = 1 selects the advanced solution algorithm)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TNOM</td>
<td>default nominal temperature (also the temperature at which model parameters are assumed to have been measured)</td>
<td>°C</td>
<td>27.0</td>
</tr>
<tr>
<td>THREADS</td>
<td>maximum number of threads. Set to 0 for engine default. THREADS=1 implies single-thread.</td>
<td></td>
<td>Number of cores/2</td>
</tr>
<tr>
<td>TRTOL</td>
<td>tolerance for integration error calculated using transient analysis. It is a relative tolerance where a higher TRTOL value results in bigger time steps and reduced accuracy. The TRTOL value should NOT be greater than 1/RELTOL.</td>
<td></td>
<td>7</td>
</tr>
<tr>
<td>VNTOL¹</td>
<td>best accuracy of voltages</td>
<td>volt</td>
<td>1.0 uV</td>
</tr>
<tr>
<td>WCDEVIATION</td>
<td>worst case deviation. It can have double values between 0 and 1.</td>
<td></td>
<td>Same as RELTOL</td>
</tr>
<tr>
<td>WIDTH</td>
<td>same as the .WIDTH OUT= statement (can be set to either 80 or 132)</td>
<td></td>
<td>80.0</td>
</tr>
</tbody>
</table>

1. These options can have an expression that uses the SCHEDULE function, which is a function of time.
Options for scheduling changes to runtime parameters

Purpose

The .OPTIONS command can be used to schedule automatic changes to certain runtime parameters during a simulation. A special command syntax is used for this (see General Form below).

Note: The ability to schedule such parameter changes only applies to transient analysis. You cannot interact with other analysis types.

General form

.OPTIONS <Parameter Name>={SCHEDULE(<time-value>, <parameter value>, <time-value>, <parameter value>, …)}

Examples

.OPTIONS RELTOL={SCHEDULE( 0s,.001,2s,.005)}

indicates that RELTOL should have a value of .001 from time 0 up to time 2s, and a value of .005 from time 2s and beyond (that is: RELTOL=.001 for t, where 0 <= t < 2s, and RELTOL=.005 for t, where t >= 2s).
PSpice A/D digital simulation condition messages

Other PSpice features produce warning messages in simulations (e.g., for the digital CONSTRAINT devices monitoring timing relationships of digital nodes). These messages are directed to the PSpice output file (and in Windows, to the Probe data file).

You can use options to control where and how many of these messages are generated. Below is a summary of the PSpice message types and a brief description of their meaning. The condition messages are specific to digital device timing violations and digital worst-case timing hazards. Refer to the Digital Simulation chapter of your PSpice User’s Guide for more information on digital worst-case timing.

<table>
<thead>
<tr>
<th>Message type</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Timing violations</strong></td>
<td></td>
</tr>
<tr>
<td>FREQUENCY</td>
<td>The minimum or maximum frequency specification for a signal has not been satisfied. Minimum frequency violations show that the period of the measured signal is too long, while maximum frequency violations describe signals changing too rapidly.</td>
</tr>
<tr>
<td>GENERAL</td>
<td>A boolean expression described within the GENERAL constraint checker was evaluated and produced a true result.</td>
</tr>
<tr>
<td>HOLD</td>
<td>The minimum time required for a data signal to be stable after the assertion of a clock, has not been met.</td>
</tr>
<tr>
<td>SETUP</td>
<td>The minimum time required for a data signal to be stable prior to the assertion of a clock, has not been met.</td>
</tr>
<tr>
<td>RELEASE</td>
<td>The minimum time for a signal that has gone inactive (usually a control such as CLEAR) to remain inactive before the asserting clock edge, has not been met.</td>
</tr>
<tr>
<td>WIDTH</td>
<td>The minimum pulse width specification for a signal has not been satisfied. That is, a pulse that is too narrow was observed on the node.</td>
</tr>
<tr>
<td><strong>Hazards</strong></td>
<td></td>
</tr>
<tr>
<td>AMBIGUITY CONVERGENCE</td>
<td>The convergence of conflicting rising and falling states (timing ambiguities) arriving at the inputs of a primitive, have produced a pulse (glitch) on the output.</td>
</tr>
</tbody>
</table>
### CUMULATIVE AMBIGUITY
Signal ambiguities are additive, increased by propagation through each level of logic in the circuit. When the ambiguities associated with both edges of a pulse increase to the point where they would overlap, this is flagged as a cumulative ambiguity hazard.

### DIGITAL INPUT VOLTAGE
When a voltage is out of range on a digital pin, PSpice uses the state whose voltage range is closest to the input voltage and continues using the simulation. A warning message is reported.

### NET-STATE CONFLICT
When two or more outputs attempt to drive a net to different states, PSpice represents the conflict as an X (unknown) state. This usually results from improper selection of a bus driver's enable inputs.

### SUPPRESSED GLITCH
A pulse applied to the input of a primitive that is shorter than the active propagation delay is ignored by PSpice. This can or cannot be significant, depending upon the nature of the circuit. The reporting of the suppressed glitch hazard shows that there might be a problem with either the stimulus, or the path delay configuration of the circuit.

### PERSISTENT HAZARD
If the effects of any of the other logic hazard messages mentioned in the output file are able to propagate to either an EXTERNAL port, or to any storage device in the circuit, they are flagged as PERSISTENT HAZARDS. (Refer to your PSpice User’s Guide for more details on PERSISTENT HAZARDs.)

### ZERO-DELAY OSCILLATION
If the output of a primitive changes more than 50 times within a single digital time step, the node is considered to be oscillating. PSpice reports this and cancels the run.
.PARAM (parameter)

Purpose  The .PARAM statement defines the value of a parameter. A parameter name can be used in place of most numeric values in the circuit description. Parameters can be constants, or expressions involving constants, or a combination of these, and they can include other parameters.

General form  .PARAM <name> = <value> >*

Examples  .PARAM VSUPPLY = 5V
          .PARAM VCC = 12V, VEE = -12V
          .PARAM BANDWIDTH = {100kHz/3}
          .PARAM PI = 3.14159, TWO_PI = {2*3.14159}
          .PARAM VNUM = {2*TWO_PI}

Arguments and options

<name>
Cannot begin with a number, and it cannot be one of the following predefined parameters, or TIME, or .TEXT (text parameter) names.

There are several predefined parameters. The parameter values must be either constants or expressions:

<table>
<thead>
<tr>
<th>Predefined parameter</th>
<th>Meaning</th>
</tr>
</thead>
</table>
| TEMP | temperature  
      | (works using ABM expressions and digital models only) |
| VT | thermal voltage (reserved) |
| GMIN | shunt conductance for semiconductor p-n junctions |

<value>
Constants (<value>) do not need braces {}.

<expression>
Can contain constants or parameters.
Comments  The .PARAM statements are order independent. They can be used inside a subcircuit definition to create local subcircuit parameters. Once defined, a parameter can be used in place of almost all numeric values in the circuit description with the following exceptions:

- *Not* in the *in-line* temperature coefficients for resistors (parameters can be used for the $\text{tc1}$ and $\text{tc2}$ resistor model parameters).
- *Not* in the PWL values for independent voltage and current source (V and I device) parameters.
- *Not* the E, F, G, and H device SPICE2G6 syntax for polynomial coefficient values and gain.

A .PARAM command can be in a library. The simulator can search libraries for parameters not defined in the circuit file, in the same way it searches for undefined models and subcircuits.

**Note:** Parameters cannot be used in place of node numbers, nor can the values on an analysis command (e.g., TRAN and AC) be parameterized. In addition the value cannot have the percent sign (%) at the end. A percent sign (%) is ignored by PSpice.
.PLOT (plot)

Purpose
The .PLOT command causes results from DC, AC, noise, and transient analyses to be line printer plots in the output file.

Note: This command is included for backward compatibility with earlier versions of PSpice. It is more effective to print plots from within Probe. Printing from Probe yields higher-resolution graphics and provides an opportunity to preview the plot before printing.

General form
.PLOT <analysis type> [output variable]* + ( [<lower limit value>, <upper limit value>] )*  

Examples
.PLOT DC V(3) V(2,3) V(R1) I(VIN) I(R2) IB(Q13) VBE(Q13)
.PLOT AC VM(2) VP(2) VM(3,4) VG(5) VDB(5) IR(D4)
.PLOT NOISE INOISE ONOISE DB(INOISE) DB(ONOISE)
.PLOT TRAN V(3) V(2,3) (0,5V) ID(M2) I(VCC) (-50mA,50mA)
.I.PLOT TRAN D(QA) D(QB) V(3) V(2,3)
.PLOT TRAN V(3) V(R1) V([RESET])

Arguments and options

<analysis type>
DC, AC, NOISE, or TRAN. Only one analysis type can be specified.

<output variable>
Following the analysis type is a list of the output variables and (possibly) Y axis scales. A maximum of 8 output variables are allowed on one .PLOT command. However, an analysis can have any number of a .PLOT command. See .PROBE (Probe) for the syntax of the output variables.

(<lower limit value>, <upper limit value>)
Sets the range of the y-axis. This forces all output variables on the same y-axis to use the specified range.

The same form, (<lower limit value>, <upper limit value>), can also be inserted one or more times in the middle of a set of output variables. Each occurrence defines one Y axis that has the specified range. All the output variables that come between it and the next range to the left in the .PLOT command are put on its corresponding Y axis.
Comments

Plots are made by using text characters to draw the plot, which print on any kind of printer. However, plots printed from within Probe look much better.

The range and increment of the x-axis is fixed by the analysis being plotted. The y-axis default range is determined by the ranges of the output variables. In the fourth example, the two voltage outputs go on the y-axis using the range (0,5V) and the two current outputs go on the y-axis using the range (-5mA, 50mA).

**Note:** Lower and upper limit values do not apply to AC Analysis.

If the different output variables differ considerably in their output ranges, then the plot is given more than one y-axis using ranges corresponding to the different output variables.

**Note:** The y-axis of frequency response plots (AC) is always logarithmic.

The last example illustrates how to plot the voltage at a node that has a name rather than a number. The first item to plot is a node voltage, the second item is the voltage across a resistor, and the third item is another node voltage, even though the second and third items both begin with the letter R. The square brackets force the interpretation of names to mean node names.
.PRINT (print)

Purpose
The .PRINT command allows results from DC, AC, noise, and transient analyses to be an output in the form of tables, referred to as print tables in the output file.

General form
```
.PPRINT[/DGTLCHG] <analysis type> [output variable]*
```

Examples
```
.PPRINT DC V[3] V[2],V(R1) I(VIN) I(R2) IB(Q13) VBE(Q13)
.PPRINT NOISE INOISE ONOISE DB(INOISE) DB(ONOISE)
.PPRINT/DGTLCHG TRAN QA QB RESET
.PPRINT TRAN V[3] V(R1) V([RESET])
```

The last example illustrates how to print a node that has a name, rather than a number. The first item to print is a node voltage, the second item is the voltage across a resistor, and the third item to print is another node voltage, even though the second and third items both begin with the letter R. The square brackets force the names to be interpreted as node names.

Arguments and options

```
[/DGTLCHG]
```
For digital output variables only. Values are printed for each output variable whenever one of the variables changes.

```
<analysis type>
```
Only one analysis type—DC, AC, NOISE, or TRAN—can be specified for each .PRINT command.

```
<output variable>
```
Following the analysis type is a list of the output variables. There is no limit to the number of output variables: the printout is split up depending on the width of the data columns (set using NUMDGT option) and the output width (set using WIDTH option). See .PROBE (Probe) for the syntax of output variables.
Comments

The values of the output variables are printed as a table where each column corresponds to one output variable. You can change the number of digits printed for analog values by using the NUMDGT option of the .OPTIONS (analysis options) command.

An analysis can have multiple .PRINT commands.
.PROBE (Probe)

Purpose
The .PROBE command writes the results from DC, AC, and transient analyses to a data file used by Probe.

General form
.PROBE[/CSDF][output variable]*

Examples
.PROBE
.PROBE V[3] V([2],[3]) V(R1) I(VIN) I(R2) IB(Q13) VBE(Q13)
.PROBE/CSDF
.PROBE V[3] V(R1) V([RESET])
.PROBE D(QBAR)

The first example (with no output variables) writes all the node voltages and all the device currents to the data file. The list of device currents written is the same as the device currents allowed as output variables.

The second example writes only those output variables specified to the data file, to restrict the size of the data file.

The third example creates a data file in a text format using the Common Simulation Data File (CSDF) format, not a binary format. This format is used for transfers between different computer families. CSDF files are larger than regular text files.

The fourth example illustrates how to specify a node that has a name rather than a number. The first item to output is a node voltage, the second item is the voltage across a resistor, and the third item to output is another node voltage, even though the second and third items both begin with the letter R. The square brackets force the interpretation of names to mean node names.

The last example writes only the output at digital node QBAR to the data file, to restrict the size of the data file.

Arguments and options
[output variable]
This section describes the types of output variables allowed in a .PRINT (print), .PLOT (plot), and .PROBE command. Each .PRINT or .PLOT can have up to 8 output variables. This format is similar to that used when calling up waveforms while running Probe.
See the tables below for descriptions of the possible output variables. If .PROBE is used without specifying a list of output variables, all of the circuit voltages and currents are stored for post-processing. When an output variable list is included, the data stored is limited to the listed items. This form is intended for users who want to limit the size of the Probe data file.

Comments
Refer to your *PSpice User’s Guide* for a description of Probe, for information about using the Probe data file, and for more information on the use of text files in Probe. You can also consult Probe Help.

**Note:** Unlike the .PRINT and .PLOT commands, there are no analysis names before the output variables. Also, the number of output variables is unlimited.

### DC Sweep and transient analysis output variables

For DC sweep and transient analysis, these are the available output variables:

<table>
<thead>
<tr>
<th>General form</th>
<th>Meaning of output variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>D(&lt;name&gt;)</td>
<td>digital value of &lt;name&gt; (a digital node)</td>
</tr>
<tr>
<td>D(*)</td>
<td>all digital data</td>
</tr>
<tr>
<td>D(alias(*))</td>
<td>all digital data except sub-circuit data</td>
</tr>
<tr>
<td>I(&lt;name&gt;)</td>
<td>current through a two terminal device</td>
</tr>
<tr>
<td>Ix(&lt;name&gt;)</td>
<td>current into a terminal of a three or four terminal device</td>
</tr>
<tr>
<td></td>
<td>(x is one of B, E, C, D, G, or S)</td>
</tr>
<tr>
<td>Iz(&lt;name&gt;)</td>
<td>current into one end of a transmission line (z is either A or B)</td>
</tr>
<tr>
<td>I(*)</td>
<td>all currents</td>
</tr>
<tr>
<td>I(alias(*))</td>
<td>all currents except sub-circuit currents</td>
</tr>
<tr>
<td>V(&lt;node&gt;)</td>
<td>voltage at a node</td>
</tr>
<tr>
<td>V(&lt;+ node&gt;, &lt;- node&gt;)</td>
<td>voltage between two nodes</td>
</tr>
<tr>
<td>V(&lt;name&gt;)</td>
<td>voltage across a two-terminal device</td>
</tr>
<tr>
<td>Vx(&lt;name&gt;)</td>
<td>voltage at a non-grounded terminal of a device</td>
</tr>
<tr>
<td></td>
<td>(x is one of B, E, C, D, G, or S)</td>
</tr>
<tr>
<td>Vz(&lt;name&gt;)</td>
<td>voltage at one end of a transmission line (z is either A or B)</td>
</tr>
<tr>
<td>General form</td>
<td>Meaning of output variable</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------</td>
</tr>
<tr>
<td>Vxy(&lt;name&gt;)</td>
<td>voltage across two terminals of a three or four terminal device type</td>
</tr>
<tr>
<td>V(*)</td>
<td>all voltages</td>
</tr>
<tr>
<td>V(alias(*))</td>
<td>all voltages except sub-circuit voltages</td>
</tr>
<tr>
<td>W(&lt;name&gt;)</td>
<td>power dissipation of a device</td>
</tr>
<tr>
<td>W(*)</td>
<td>all power data</td>
</tr>
<tr>
<td>W(alias(*))</td>
<td>all power data except sub-circuit data</td>
</tr>
</tbody>
</table>

1. These values are available for transient only.
2. For the .PRINT/DGTLCHG statement, the \( D(\) ) is optional.

**Note:** The aliases must be defined in the circuit file to generate the .dat file.

<table>
<thead>
<tr>
<th>Example</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>D(QA)</td>
<td>the value of digital node QA</td>
</tr>
<tr>
<td>I(D5)</td>
<td>current through diode D5</td>
</tr>
<tr>
<td>IG(J10)</td>
<td>current into gate of J10</td>
</tr>
<tr>
<td>V(3)</td>
<td>voltage between node three and ground</td>
</tr>
<tr>
<td>V(3,2)</td>
<td>voltage between nodes three and two</td>
</tr>
<tr>
<td>V(R1)</td>
<td>voltage across resistor R1</td>
</tr>
<tr>
<td>VA(T2)</td>
<td>voltage at port A of T2</td>
</tr>
<tr>
<td>VB(Q3)</td>
<td>voltage between base of transistor Q3 and ground</td>
</tr>
<tr>
<td>VGS(M13)</td>
<td>gate-source voltage of M13</td>
</tr>
<tr>
<td>W(U7)</td>
<td>power dissipated by device U7</td>
</tr>
</tbody>
</table>
Multiple-terminal devices

For the $V(<name>)$ and $I(<name>)$ forms, where $<name>$ must be the name of a two-terminal device, the devices are:

<table>
<thead>
<tr>
<th>Character ID</th>
<th>Two-terminal device</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>capacitor</td>
</tr>
<tr>
<td>D</td>
<td>diode</td>
</tr>
<tr>
<td>E</td>
<td>voltage-controlled voltage source</td>
</tr>
<tr>
<td>F</td>
<td>current-controlled current source</td>
</tr>
<tr>
<td>G</td>
<td>voltage-controlled current source</td>
</tr>
<tr>
<td>H</td>
<td>current-controlled voltage source</td>
</tr>
<tr>
<td>I</td>
<td>independent current source</td>
</tr>
<tr>
<td>L</td>
<td>inductor</td>
</tr>
<tr>
<td>R</td>
<td>resistor</td>
</tr>
<tr>
<td>S</td>
<td>voltage-controlled switch</td>
</tr>
<tr>
<td>V</td>
<td>independent voltage source</td>
</tr>
<tr>
<td>W</td>
<td>current-controlled switch</td>
</tr>
</tbody>
</table>

For the $Vx(<name>)$, $Vxy(<name>)$, and $Ix(<name>)$ forms, where $<name>$ must be the name of a three or four-terminal device and $x$ and $y$ must each be a terminal abbreviation, the devices and the terminals areas follows. For the $Vz(<name>)$ and $Iz(<name>)$ forms, $<name>$ must be the name of a transmission line (T device) and $z$ must be A or B.

<table>
<thead>
<tr>
<th>Three &amp; four-terminal device type</th>
<th>Terminal abbreviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>B (GaAs MESFET)</td>
<td>D (drain)</td>
</tr>
<tr>
<td></td>
<td>G (gate)</td>
</tr>
<tr>
<td></td>
<td>S (source)</td>
</tr>
<tr>
<td>J (Junction FET)</td>
<td>D (drain)</td>
</tr>
<tr>
<td></td>
<td>G (gate)</td>
</tr>
<tr>
<td></td>
<td>S (source)</td>
</tr>
</tbody>
</table>
### Terminal Abbreviation, continued

<table>
<thead>
<tr>
<th>Three &amp; four-terminal device type, continued</th>
<th>Terminal abbreviation, continued</th>
</tr>
</thead>
<tbody>
<tr>
<td>M (MOSFET)</td>
<td>D (drain)</td>
</tr>
<tr>
<td></td>
<td>G (gate)</td>
</tr>
<tr>
<td></td>
<td>S (source)</td>
</tr>
<tr>
<td></td>
<td>B (bulk, substrate)</td>
</tr>
<tr>
<td>Q (Bipolar transistor)</td>
<td>C (collector)</td>
</tr>
<tr>
<td></td>
<td>B (base)</td>
</tr>
<tr>
<td></td>
<td>E (emitter)</td>
</tr>
<tr>
<td></td>
<td>S (substrate)</td>
</tr>
<tr>
<td>T (transmission line)</td>
<td>Va (near side voltage)</td>
</tr>
<tr>
<td></td>
<td>Ia (near side current)</td>
</tr>
<tr>
<td></td>
<td>Vb (far side voltage)</td>
</tr>
<tr>
<td></td>
<td>Ib (far side current)</td>
</tr>
<tr>
<td>Z (IGBT)</td>
<td>C (collector)</td>
</tr>
<tr>
<td></td>
<td>G (gate)</td>
</tr>
<tr>
<td></td>
<td>E (emitter)</td>
</tr>
</tbody>
</table>
AC analysis

For AC analysis, these are the available output variables:

<table>
<thead>
<tr>
<th>General form</th>
<th>Meaning of output variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>I(&lt;name&gt;)</td>
<td>current through a two terminal device</td>
</tr>
<tr>
<td>IX(&lt;name&gt;)</td>
<td>current into a terminal of a three or four terminal device</td>
</tr>
<tr>
<td></td>
<td>(x is one of E, B, C, D, G, or S)</td>
</tr>
<tr>
<td>IZ(&lt;name&gt;)</td>
<td>current into one end of a transmission line (z is either A or B)</td>
</tr>
<tr>
<td>I(*)</td>
<td>all currents</td>
</tr>
<tr>
<td>I(alias(*))</td>
<td>all currents except internal sub-circuit currents</td>
</tr>
<tr>
<td>Noise(*)</td>
<td>all analog noise components</td>
</tr>
<tr>
<td>Noise(alias(*))</td>
<td>all except internal subcircuit analog noise components</td>
</tr>
<tr>
<td>V(&lt;node&gt;)</td>
<td>voltage at a node</td>
</tr>
<tr>
<td>V(&lt;+ node&gt;, &lt;- node&gt;)</td>
<td>voltage between two nodes</td>
</tr>
<tr>
<td>V(&lt;name&gt;)</td>
<td>voltage across a two-terminal device</td>
</tr>
<tr>
<td>VX(&lt;name&gt;)</td>
<td>voltage at a non-grounded terminal of a device</td>
</tr>
<tr>
<td></td>
<td>(x is one of E, B, C, D, G, or S)</td>
</tr>
<tr>
<td>VZ(&lt;name&gt;)</td>
<td>voltage at one end of a transmission line (z is either A or B)</td>
</tr>
<tr>
<td>VXY(&lt;name&gt;)</td>
<td>voltage across two terminals of a three or four terminal device</td>
</tr>
<tr>
<td></td>
<td>type</td>
</tr>
<tr>
<td>V(*)</td>
<td>all voltages</td>
</tr>
<tr>
<td>V(alias(*))</td>
<td>all voltages except internal sub-circuit voltages</td>
</tr>
<tr>
<td>W(&lt;name&gt;)</td>
<td>power dissipation of a device</td>
</tr>
<tr>
<td>W(*)</td>
<td>all power data</td>
</tr>
<tr>
<td>W(alias(*))</td>
<td>all power data except internal sub-circuit data</td>
</tr>
</tbody>
</table>

For AC analysis, the output variables listed in the preceding table are augmented by adding a suffix.

**Note:** For AC analysis, the suffixes are ignored for a .PROBE command, but can be used in a .PRINT (print) on page 84 command and a .PLOT (plot) on page 82 command, and when
adding a trace in Probe. For example, in a .PROBE command, VDB(R1) is translated to V(R1), which is the raw data.

Current outputs are not available for devices, such as F device (current-controlled current sources) and G device (voltage-controlled current source). For these devices, you must first put a zero-valued voltage source in series with the device (or terminal) of interest and then print or plot the current through this voltage source.

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Meaning of output variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td>magnitude</td>
</tr>
<tr>
<td>DB</td>
<td>magnitude in decibels</td>
</tr>
<tr>
<td>G</td>
<td>group delay (-dPHASE/dFREQUENCY)</td>
</tr>
<tr>
<td>I</td>
<td>imaginary part</td>
</tr>
<tr>
<td>M</td>
<td>magnitude</td>
</tr>
<tr>
<td>P</td>
<td>phase in degrees</td>
</tr>
<tr>
<td>R</td>
<td>real part</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Example</th>
<th>Meaning of output variables for AC analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>II(R13)</td>
<td>imaginary part of current through R13</td>
</tr>
<tr>
<td>IGG(M3)</td>
<td>group delay of gate current for M3</td>
</tr>
<tr>
<td>IR(VIN)</td>
<td>real part of I through VIN</td>
</tr>
<tr>
<td>IAG(T2)</td>
<td>group delay of current at port A of T2</td>
</tr>
<tr>
<td>V(2,3)</td>
<td>magnitude of complex voltage across nodes 2 &amp; 3</td>
</tr>
<tr>
<td>VDB(R1)</td>
<td>db magnitude of V across R1</td>
</tr>
<tr>
<td>VBEP(Q3)</td>
<td>phase of base-emitter V at Q3</td>
</tr>
<tr>
<td>VM(2)</td>
<td>magnitude of V at node 2</td>
</tr>
<tr>
<td>WM(U7)</td>
<td>magnitude of power dissipated by device U7</td>
</tr>
</tbody>
</table>

Note: Current outputs for the F and G devices are not available for DC and transient analyses.
Noise analysis

For noise analysis, the output variables are predefined as follows:

<table>
<thead>
<tr>
<th>Output variable</th>
<th>Meaning of output variables for noise analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>INOISE</td>
<td>Total RMS summed noise at input node</td>
</tr>
<tr>
<td>ONOISE</td>
<td>INOISE equivalent at output node</td>
</tr>
<tr>
<td>DB(INOISE)</td>
<td>INOISE in decibels</td>
</tr>
<tr>
<td>DB(ONOISE)</td>
<td>ONOISE in decibels</td>
</tr>
</tbody>
</table>

Note: .PRINT (print) on page 84 and .PLOT (plot) on page 82 cannot be used for the noise from any one device. However, the print interval on the .NOISE (noise analysis) on page 67 command can be used to output this information.
.SAVEBIAS (save bias point to file)

Purpose
The .SAVEBIAS command saves the bias point node voltages and inductor currents, to a file. It is used concurrently with .LOADBIAS (load bias point file) on page 53.

Only one analysis is specified in a .SAVEBIAS command, which can be OP, TRAN, or DC. However, a circuit file can contain a .SAVEBIAS command for each of the three analysis types. If the simulation parameters do not match the keywords and values in the .SAVEBIAS command, then no file is produced.

General form
```
.SAVEBIAS <"file_name"> <[OP] [TRAN] [DC]> [NOSUBCKT]
+[TIME=<value> [REPEAT]] [TEMP=<value>]
+[STEP=<value>] [MCRUN=<value>] [DC=<value>]
+[DC1=<value>] [DC2=<value>]
```

Examples
For the first example, the small-signal operating point (.AC or .OP) bias point is saved.
```
.SAVEBIAS "OPPOINT" OP
```

In the second example, the transient bias point is written out at the time closest to, but not less than 10.0 u/sec. No bias point information for subcircuits is saved.
```
.SAVEBIAS "TRANDATA.BSP" TRAN NOSUBCKT TIME=10u
```

Use of the [REPEAT] keyword in the third example causes the bias point to be written out every 5.0 ns when the temperature of the run is 50.0 degrees.
```
.SAVEBIAS "SAVETRAN.BSP" TRAN TIME=5n REPEAT TEMP=50.0
```

In the fourth example, because there are no parameters supplied, only the very first DC bias point is written to the file.
```
.SAVEBIAS "DCBIAS.SAV" DC
```

The fifth example saves the DC bias point when the following three conditions are all met: the first DC sweep value is 3.5, the second DC sweep value is 100, and the simulation is on the third Monte Carlo run. If only one DC sweep is being performed, then the keyword DC can be substituted for DC1.
```
.SAVEBIAS "SAVEDC.BSP" DC MCRUN=3 DC1=3.5 DC2=100
```
Arguments and options

<"file name">
Any valid file name for the computer system, which must be enclosed in quotation marks.

[NOSUBCKT]
When used, the node voltages and inductor currents for subcircuits are not saved.

[TIME=<value> [REPEAT]]
Used to define the transient analysis time at which the bias point is to be saved.

[TEMP=<value>]
Defines the temperature at which the bias point is to be saved.

[STEP=<value>]
The step value at which the bias point is to be saved.

[MCRUN=<value>]
The number of the Monte Carlo or worst-case analysis run for which the bias point is to be saved.

[DC=<value>], [DC1=<value>], and [DC2=<value>]
Used to specify the DC sweep value at which the bias point is to be saved.

Comments

If REPEAT is not used, then the bias at the next time point greater than or equal to TIME=<value> is saved. If REPEAT is used, then TIME=<value> is the interval at which the bias is saved. However, only the latest bias is saved; any previous times are overwritten. The [TIME=<value> [REPEAT]] can only be used with a transient analysis.

The [DC=<value>] should be used if there is only one sweep variable. If there are two sweep variables, then [DC1=<value>] should be used to specify the first sweep value and [DC2=<value>] should be used to specify the second sweep value.
The saved bias point information is in the following format: one or more comment lines that list items such as:

- circuit name, title, date and time of run, analysis, and temperature, or
- a single .NODESET (set approximate node voltage for bias point) on page 66 command containing the bias point voltage values and inductor currents.

Only one bias point is saved to the file during any particular analysis. At the specified time, the bias point information and the operating point data for the active devices and controlled sources are written to the output file. When the supplied specifications on the .SAVEBIAS command line match the state of the simulator during execution, the bias point is written out.

**Usage examples**

A .SAVEBIAS command and a .LOADBIAS (load bias point file) command can be used to shorten the simulation time of large circuits, and also to aid in convergence.

A typical application for a .SAVEBIAS and a .LOADBIAS command is for a simulation that takes a considerable amount of time to converge to a bias point. The bias point is saved using a .SAVEBIAS command so that when the simulation is run again, the previous bias point calculated is used as a starting point for the bias solution, to save processing time.

The following example illustrates this procedure for a transient simulation.

```
.SAVEBIAS "SAVEFILE.TRN" TRAN
```

When the simulation is run, the transient analysis bias point information is saved to the file `savefile.trn` in the form of a .NODESET command. This .NODESET command provides the simulator with a starting solution for determining the bias point calculation for future simulations. To use this file, replace the .SAVEBIAS command in the circuit file using the following .LOADBIAS (Load Bias Point File) command.

```
.LOADBIAS "SAVEFILE.TRN"
```

**Note:** A .SAVEBIAS and .LOADBIAS command should not refer to the same file during the same simulation run. Use the .SAVEBIAS during the first simulation and the .LOADBIAS for subsequent ones.

The simulator algorithms have been changed to provide an automatic saving and loading of bias point information under certain conditions. This automatic feature is used in the following analysis types: **.STEP (parametric analysis)** on page 99, **.DC (DC analysis)** on page 37,
A typical application is a transient analysis where the bias point is calculated at several temperatures (such as .TEMP 0 10 20 30). As each new temperature is processed, the bias point for the previous temperature is used to find the new bias point. Since this process is automatic, the user does not have to change anything in the circuit file. However, there is some memory overhead since the bias point information is saved during the simulation. Disable the automatic saving feature, using the NOREUSE flag option in the .OPTIONS (analysis options) on page 71 command as follows:

```
.OPTIONS NOREUSE
```

Another application for the .LOADBIAS and .SAVEBIAS command is the handling of convergence problems. Consider a circuit which has difficulty in starting a DC sweep. The designer has added a .NODESET command as shown below to help the simulator determine the bias point solution.

```
.NODESET V(3)=5.0V V(4)=2.75V
```

Even though this helps the simulator determine the bias point, the simulator still has to compute the starting values for each of the other nodes. These values can be saved using the following statement:

```
.SAVEBIAS "DCOP.NOD" DC
```

The next time the simulation is run, the .NODESET and .SAVEBIAS command should be removed and replaced using the following:

```
.LOADBIAS "DCOP.NOD"
```

This provides the starting values for all of the nodes in the circuit, and can assist the simulator in converging to the correct bias point for the start of the sweep. If convergence problems are caused by a change in the circuit topology, the designer can edit the bias point save file to change the values for specific nodes or to add new nodes.
.SENS (sensitivity analysis)

Purpose
The .SENS command performs a DC sensitivity analysis.

General form
.SENS <output variable>*

Examples
.SENS V(9) V(4,3) V(17) I(VCC)

Arguments and options

<output variable>
Same format and meaning as in the .PRINT command for DC and transient analyses. However, when <output variable> is a current, it is restricted to be the current through a voltage source.

Comments
By linearizing the circuit about the bias point, the sensitivities of each of the output variables to all the device values and model parameters is calculated and output data generated. This can generate large amounts of output data.

Device sensitivities are only provided for the following device types:

- resistors
- independent voltage and current sources
- voltage and current-controlled switches
- diodes
- bipolar transistors

Note: The results of the .SENS command are only available in the output file. They cannot be viewed in Probe.
.STEP (parametric analysis)

Purpose
The .STEP command performs a parametric sweep for all of the analyses of the circuit.

The .STEP command is similar to the .TEMP (temperature) command in that all of the typical analyses—such as .DC (DC analysis), .AC (AC analysis), and .TRAN (transient analysis)—are performed for each step.

Once all the runs finish, the specified .PRINT (print) table or .PLOT (plot) plot for each value of the sweep is an output, just as for the .TEMP or .MC (Monte Carlo analysis) command.

Probe displays nested sweeps as a family of curves.

General form
The first general form is for doing a linear sweep. The second form is for doing a logarithmic sweep. The third form is for using a list of values for the sweep variable.

Examples
The first three examples are for doing a linear sweep. The fourth example is for doing a logarithmic sweep. The fifth example is for using a list of values for the sweep variable.

Arguments and options
Sweep type
The sweep can be linear, logarithmic, or a list of values. For [linear sweep type], the keyword LIN is optional, but either OCT or DEC must be specified for the <logarithmic sweep type>. The sweep types are described below.
### Sweep types
<table>
<thead>
<tr>
<th></th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIN</td>
<td>Linear sweep. The sweep variable is swept linearly from the starting to the ending value. The <code>&lt;increment value&gt;</code> is the step size.</td>
</tr>
<tr>
<td>OCT</td>
<td>Sweep by octaves. The sweep variable is swept logarithmically by octaves. The <code>&lt;points value&gt;</code> is the number of steps per octave.</td>
</tr>
<tr>
<td>DEC</td>
<td>Sweep by decades. The sweep variable is swept logarithmically by decades. The <code>&lt;points value&gt;</code> is the number of steps per decade.</td>
</tr>
<tr>
<td>LIST</td>
<td>Use a list of values. In this case there are no start and end values. Instead, the numbers that follow the keyword LIST are the values that the sweep variable is set to. <strong>Note:</strong> The LIST values must be in either ascending or descending order.</td>
</tr>
</tbody>
</table>

### Sweep Variable Name
- `<sweep variable name>`

The `<sweep variable name>` can be one of the types described below.

<table>
<thead>
<tr>
<th>Sweep Variable Name</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>source</td>
<td>A name of an independent voltage or current source. During the sweep, the source's voltage or current is set to the sweep value.</td>
</tr>
<tr>
<td>model parameter</td>
<td>A model type and model name followed by a model parameter name in parenthesis. The parameter in the model is set to the sweep value.</td>
</tr>
<tr>
<td>temperature</td>
<td>Use the keyword TEMP for <code>&lt;sweep variable name&gt;</code>. The temperature is set to the sweep value. For each value in the sweep, all the circuit components have their model parameters updated to that temperature.</td>
</tr>
</tbody>
</table>
Usage examples

One

The .STEP command only steps the DC component of an AC source. In order to step the AC component of an AC source, a variable parameter has to be created. For example,

```
Vac 1 0 AC {variable}
.param variable=0
.step param variable 0 5 1
.ac dec 100 1000 1e6
```
Two

This is one way of stepping a resistor from 30 to 50 ohms in steps of 5 ohms, using a global parameter:

```plaintext
.PARAM RVAL = 1
R1 1 2 {RVAL}
.STEP PARAM RVAL 30,50,5
```

The parameter RVAL is global and PARAM is the keyword used by the .STEP command when using a global parameter.

Three

The following example steps the resistor model parameter R. This is another way of stepping a resistor from 30 to 50 ohms in steps of 5 ohms.

```plaintext
R1 1 2 RMOD 1
.MODEL RMOD RES(R=30)
.STEP RES RMOD(R) 30,50,5
```

Note: Do not use R={30}.

Here RMOD is the model name, RES is the sweep variable name (a model type), and R is the parameter within the model to step. To step the value of the resistor, the line value of the resistor is multiplied by the R parameter value to achieve the final resistance value, that is:

```plaintext
final resistor value = line resistor value \cdot R
```

Therefore, if the line value of the resistor is set to one ohm, the final resistor value is 1 \cdot R or R. Stepping R from 30 to 50 ohms then steps the resistor value from 1 \cdot 30 ohms to 1 \cdot 50 ohms.

In examples 2 and 3, all of the ordinary analyses (e.g., .DC, .AC, and .TRAN) are run for each step.
.STIMLIB (stimulus library file)

Purpose
The .STIMLIB command makes stimulus library files created by StmEd available to PSpice.

General form
.STIMLIB <file name [.stl]>

Examples
.STIMLIB mylib.stl
.STIMLIB volts.stl
.STIMLIB dgpulse

Arguments and options

<file name>
Specification that identifies a file containing .STIMULUS commands.
.STIMULUS (stimulus)

Purpose
The .STIMULUS command encompasses only the Transient specification portion of what is allowed in the V or I device syntax.

General form
.STIMULUS <stimulus name> <type> <type-specific parameters>*

Examples
.STIMULUS InputPulse PULSE (-1mv 1mv 2ns 2ns 50ns 100ns)
.STIMULUS DigitalPulse STIM (1,1)
+ 0S 1
+ 10NS 0
+ 20NS 1
.STIMULUS 50KHZSIN SIN (0 5 50KHZ 0 0 0)

Arguments and options
<stimulus name>
The name by which the stimulus is referred to by the source devices (V or I), or by the digital STIM device.

Comments
.STIMULUS commands generally appear within stimulus libraries created by StmEd.
.SUBCKT (subcircuit)

Purpose
The .SUBCKT command/statement starts the subcircuit definition by specifying its name, the number and order of its terminals, and the names and default parameters that control its behavior. Subcircuits are instantiated using X (Subcircuit instantiation on page 329) devices. The .ENDS command marks the end of a subcircuit definition.

General form
```plaintext
.SUBCKT <name> [node]*
+ [OPTIONAL: < <interface node> = <default value> >*]  
+ [PARAMS: < <name> = <value> >* ]  
+ [TEXT: < <name> = <text value> >* ]
...
.ENDS
```

Examples
```plaintext
.SUBCKT OPAMP 1 2 101 102 17
...
.ENDS

.SUBCKT FILTER INPUT OUTPUT PARAMS: CENTER=100kHz,  
+ BANDWIDTH=10kHz
...
.ENDS

.SUBCKT PLD IN1 IN2 IN3 OUT1  
+ PARAMS: MNTYMXDLY=0 IO_LEVEL=0  
+ TEXT: JEDEC_FILE="PROG.JED"
...
.ENDS

.SUBCKT 74LS00 A B Y  
+ OPTIONAL: DPWR=SG_DPWR DGND=SG_DGND  
+ PARAMS: MNTYMXDLY=0 IO_LEVEL=0
...
.ENDS
```

Arguments and options
- `<name>`
The name is used by an X (Subcircuit Instantiation) device to reference the subcircuit.

- `[node]`*
An optional list of nodes (pins). This is optional because it is possible to specify a subcircuit that has no interface nodes.

- `OPTIONAL:`
Allows specification of one or more optional nodes (pins) in the subcircuit definition.
Comments

The subcircuit definition ends with a .ENDS command. All of the netlist between .SUBCKT and .ENDS is included in the definition. Whenever the subcircuit is used by an X (Subcircuit Instantiation) device, all of the netlist in the definition replaces the X device.

There must be the same number of nodes in the subcircuit calling statements as in its definition. When the subcircuit is called, the actual nodes (the ones in the calling statement) replace the argument nodes (the ones in the defining statement).

Note: Do not use 0 (zero) in this node list. Zero is reserved for the global ground node.

The optional nodes are stated as pairs consisting of an interface node and its default value. If an optional node is not specified in an X device, its default value is used inside the subcircuit; otherwise, the value specified in the definition is used.

This feature is particularly useful when specifying power supply nodes, because the same nodes are normally used in every device. This makes the subcircuits easier to use because the same two nodes do not have to be specified in each subcircuit statement. This method is used in the libraries provided with the Digital Simulation feature.

Subcircuits can be nested. That is, an X device can appear between .SUBCKT and .ENDS commands. However, subcircuit definitions cannot be nested. That is, a .SUBCKT statement cannot appear in the statements between a .SUBCKT and a .ENDS.

Subcircuit definitions should contain only device instantiations (statements without a leading period) and possibly these statements:

- .IC (initial bias point condition) on page 49
- .NODESET (set approximate node voltage for bias point) on page 66
- .MODEL (model definition) on page 58
- .PARAM (parameter) on page 80
- .FUNC (function) on page 48
Models, parameters, and functions defined within a subcircuit definition are available only within the subcircuit definition in which they appear. Also, if a .MODEL, .PARAM, or a .FUNC statement appears in the main circuit, it is available in the main circuit and all subcircuits.

Node, device, and model names are local to the subcircuit in which they are defined. It is acceptable to use a name in a subcircuit which has already been used in the main circuit. When the subcircuit is expanded, all its names are prefixed using the subcircuit instance name: for example, Q13 becomes X3.Q13 and node 5 becomes X3.5 after expansion. After expansion all names are unique. The only exception is the use of global node names (refer to your PSpice User’s Guide) that are not expanded.

The keyword PARAMS: passes values into subcircuits as arguments and uses them in expressions inside the subcircuit. The keyword TEXT: passes text values into subcircuits as arguments and uses them as expressions inside the subcircuit. Once defined, a text parameter can be used in the following places:

- To specify a JEDEC file name on a PLD device.
- To specify an Intel Hex file name to program a ROM device or initialize a RAM device.
- To specify a stimulus file name or signal name on a FSTIM device.
- To specify a text parameter to a (lower level) subcircuit.
- As part of a text expression used in one of the above.

**Note:** The text parameters and expressions are currently only used in Digital Simulation.

### Usage examples

#### One

In the example of the 74LS00 subcircuit, the following subcircuit reference uses the default power supply nodes $G_DPWR and $G_DGND:

```
x1 in1 in2 out 74LS00
```
Two

To specify your own power supply nodes MYPOWER and MYGROUND, use the following subcircuit instantiation:

\[ X2 \text{ IN1 IN2 OUT MYPOWER MYGROUND 74LS00} \]

Three

If wanted, one optional node in the subcircuit instantiation can be provided. In the following subcircuit instantiation, the default $G_DGND$ would be used:

\[ X3 \text{ IN1 IN2 OUT MYPOWER 74LS00} \]

Four

However, to specify values beyond the first optional node, all nodes previous to that node must be specified. For example, to specify your own ground node, the default power node before it must be explicitly stated:

\[ X4 \text{ IN1 IN2 OUT $G_DPWR$ MYGROUND 74LS00} \]
.TEMP (temperature)

Purpose
The .TEMP command sets the temperature at which all analyses are done.

General form
.TEMP <temperature value>*

Examples
.TEMP 125
.TEMP 0 27 125

Comments
The temperatures are in degrees Centigrade. If more than one
temperature is given, then all analyses are performed for each
temperature.

It is assumed that the model parameters were measured or derived at the
nominal temperature, TNOM (27°C by default). See the
.OPTIONS (analysis options) command for setting TNOM.

.TEMP behaves similarly to the list variant of the
.STEP (parametric analysis) statement, with the stepped variable being
the temperature.
.TEXT (text parameter)

Purpose
The .TEXT command precedes a list of names and text values.

General form
\[ .TEXT < \text{name} = "\text{text value}" >* \]
\[ .TEXT < \text{name} = \text{|text expression|} >* \]

Examples
\[ .TEXT MYFILE = "FILENAME.EXT" \]
\[ .TEXT FILE = "ROM.DAT", FILE2 = "ROM2.DAT" \]
\[ .TEXT PROGDAT = "ROM"+TEXTINT(RUN_NO)+".DAT" | \]
\[ .TEXT DATA1 = "PLD.JED", PROGDAT = \]
\[ "\PROG\DAT\"+FILENAME | \]

Arguments and options
- \text{name}
  Cannot be a .PARAM name, or any of the reserved parameters names.

- \text{text expression}
  Text expressions can contain the following:

<table>
<thead>
<tr>
<th>Text expressions</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>enclosed in “ ”</td>
<td>text constants</td>
</tr>
<tr>
<td>text parameters</td>
<td>previously defined parameters</td>
</tr>
<tr>
<td>+</td>
<td>the operator that concatenates two text values</td>
</tr>
<tr>
<td>TEXTINT(&lt;value or expression&gt;)</td>
<td>a function which returns a text string which is the integer value closest to the value of the &lt;value or expression&gt;; (&lt;value or expression&gt; is a floating-point value)</td>
</tr>
</tbody>
</table>
Comments  The values can be text constants (enclosed in quotation marks ““) or text expressions (enclosed in |). Text expressions can contain only text constants or previously defined parameters. Once defined, a text parameter has the following uses:

■ To specify a JEDEC file name on a PLD device.
■ To specify an Intel Hex file name to program a ROM device or initialize a RAM device.
■ To specify a stimulus file name or signal name on an FSTIM device.
■ To specify a text parameter to a subcircuit.
■ As part of a text expression used in one of the above.

Note: Text parameters and expressions are only used in digital simulation.
.TF (transfer)

**Purpose**
The .TF command/statement causes the small-signal DC gain to
be calculated by linearizing the circuit around the bias point.

**General form**

```
.TF <output variable> <input source name>
```

**Examples**

```
.TF V(5) VIN
.TF I(VDRIV) ICNTRL
```

**Arguments and options**

- `<output variable>`
  This has the same format and meaning as in the .PRINT (print)
  statement.

**Comments**
The gain from `<input source name>` to `<output
variable>` and the input and output resistances are evaluated
and written to the output file. This output does not require a
.PRINT (print), .PLOT (plot), or .PROBE (Probe) statement. When
`<output variable>` is a current, it is restricted to be the current
through a voltage source.

**Note:** The results of the .TF command are only available in the
output file. They cannot be viewed in Probe.
.TRAN (transient analysis)

Purpose

The .TRAN command causes a transient analysis to be performed on the circuit and specifies the time period for the analysis.

General form

```
.TRAN[/OP] <print step value> <final time value>
+[no-print value [step ceiling value]][SKIPBP]
```

Examples

```
.TRAN 1ns 100ns
.TRAN/OP 1ns 100ns 20ns SKIPBP
.TRAN 1ns 100ns 0ns .1ns
.TRAN 1ns 100ns 0ns {SCHEDULE(0,1ns,25ns,.1ns)}
```

Arguments and options

```
[/OP]
Causes the same detailed printing of the bias point that the .OP (bias point) command does for the regular bias point. Without using this option, only the node voltages are printed for the transient analysis bias point.

<print step value>
Sets the time interval used for printing (.PRINT), plotting (.PLOT), or performing a Fourier integral on (.FOUR) the results of the transient analysis.

Since the results are computed at different times than they are printed, a 2nd-order polynomial interpolation is used to obtain the printed values. This applies only to .PRINT (print), .PLOT (plot), and .FOUR (Fourier analysis) outputs and does not affect Probe.

<final time value>
Sets the end time for the analysis.

[no-print value]
Sets the time interval (from TIME=0) that is not printed, plotted, or given to Probe.

[step ceiling value]
Overrides the default ceiling on the internal time step with a lower value.

The function SCHEDULE(x_1,y_1,x_2,y_2...x_n,y_n) can be used in place of the step ceiling value to define a piecewise constant function (from time x forward use y).
[SKIPBP]
Skips calculation of the bias point.

When this option is used, the bias conditions are fully determined by the IC= specifications for capacitors and inductors.

Comments
The transient analysis calculates the circuit's behavior over time, always starting at TIME=0 and finishing at <final time value>, but you can suppress the output of a portion of the analysis. Use a .PRINT (print), .PLOT (plot), .FOUR (Fourier analysis), or .PROBE (Probe) to get the results of the transient analysis.

Prior to performing the transient analysis, PSpice computes a bias point for the circuit separate from the regular bias point. This is necessary because at the start of a transient analysis, the independent sources can have different values than their DC values.

The internal time step of the transient analysis adjusts as the analysis proceeds: over intervals when there is little activity, the time step is increased, and during busy intervals it is decreased. The default ceiling on the internal time step is <final time value>/50, but when there are no charge storage elements, inductances, or capacitances in the circuit, the ceiling is <print step value>.

The .TRAN command also sets the variables TSTEP and TSTOP, which are used in defaulting some waveform parameters. TSTEP is equal to <print step value> and TSTOP is equal to <final time value>.

Refer to your PSpice User’s Guide for more information on setting initial conditions.

Scheduling changes to runtime parameters with the .TRAN statement

Purpose
You can schedule automatic changes to the TMAX parameter for the .TRAN statement during a transient analysis.

General form
For TMAX only, the general form is:

```
.TRAN <time-value> <time-value> <step ceiling>
{SCHEDULE(<time-value>, <parameter value>, <time-value>, <parameter value>, ...)

where the first value is time and the second value is step ceiling. For more details, see the .TRAN (transient analysis) command section.
```
Examples

.TRAN .1ns 100ns 0ns {SCHEDULE(0,1ns,25ns,.1ns)}

Note: For more information on scheduling runtime parameters during a simulation, see the .OPTIONS command section.
**.VECTOR (digital output)**

**Purpose**  
The `.VECTOR` command is used to create files containing digital simulation results.

**General form**
```
.VECTOR <number of nodes> <node>*
  + [ POS = <column position> ]
  + [ FILE = <filename> ]
  + [ RADIX = "Binary" | "Hex" | "Octal"
      + [ BIT = <bit index> ] ]
  + [ SIGNAMES = <signal names> ]
```

**Examples**
```
.VECTOR 1 CLOCK SIGNAMES=SYSCLK
.VECTOR 4 DATA3 DATA2 DATA1 DATA0
.VECTOR 1 ADDR3 POS=2 RADIX=H BIT=4
.VECTOR 1 ADDR2 POS=2 RADIX=H BIT=3
.VECTOR 1 ADDR1 POS=2 RADIX=H BIT=2
.VECTOR 1 ADDR0 POS=2 RADIX=H BIT=1
```

**Arguments and options**
<number of nodes>
This means the number of nodes in the list.

<nodes>
This defines the nodes whose states are to be stored.

The optional parameters on the .VECTOR command can be used to control the file name, column order, radix of the state values, and signal names which appear in the file header. Each of the optional parameter is described below in detail.

POS <column position>

The optional parameter POS specifies the column position in the file. By default, the column position is determined through the order in which the .VECTOR command appears in the circuit file, and by the order of the signals within a .VECTOR command. Valid values for <column position> are 1-255.

FILE <filename>
Specifies the name of the file to which the simulation results are saved. By default, the .VECTOR command creates a file named <circuit filename>.vec. The name of the corresponding netlist file created is <circuit filename>.cir. You can use the FILE parameter to specify a different filename. Multiple .VECTOR commands can be used to specify nodes for the same file.
RADIX
The radix of the values for the specified nodes is defined if <number of nodes> is greater than one. Valid values are BINARY, OCTAL, or HEX (you can abbreviate to the first letter). If <number of nodes> is one, and a radix of OCTAL or HEX is specified, a bit position within the octal or hex digit via the BIT parameter can also be specified. A separate .VECTOR command can be used to construct multi-bit values out of single signals, provided the same POS value is specified. The default radix is BINARY if <number of nodes> is one. Otherwise, the default radix is HEX. If a radix of OCTAL or HEX is specified, the simulator creates dummy entries in the vector file header to fill out the value if <number of nodes> is not an even power of 2.

BIT <bit index>
Defines the bit position within a single hex or octal digit when the VECTOR symbol is attached to a wire. Valid values are 1 through 4 if RADIX=HEX, and 1 through 3 if RADIX=OCTAL.

SIGNAMES <signal names>
Defines the names of the signals which appear in the header of the vector file. If SIGNAMES is not specified, the <node> names are used in the vector file header. If <number of nodes> is greater than one, names are defined positionally, msb to lsb. If fewer signal names than <number of nodes> are specified, the <node> names are used for the remaining unspecified names.

Comments
The file created using the .VECTOR command contains time and state values for the circuit nodes specified in the statement. The file format is identical to that used by the digital file stimulus device (FSTIM). Thus, the results of one simulation can be used to drive inputs of a subsequent simulation. See File stimulus on page 425 for more information on the file stimulus file format.
.WATCH (watch analysis results)

Purpose
The .WATCH command/statement outputs results from DC, AC, and transient analyses to the Simulation Status window in the Probe display under the Watch tab in text format while the simulation is running.

General form
```
.WATCH [DC][AC][TRAN] + [<output variable> [<lower limit value>,<upper limit value>]]*
```

Examples
```
.WATCH DC V(3) (-1V,4V) V(2,3) V(R1)
.WATCH AC VM(2) VP(2) VMC(Q1)
.WATCH TRAN VBE(Q13) (0V,5V) ID(M2) I(VCC) (0,500mA)
.WATCH DC V([RESET]) (2.5V,10V)
```

Arguments and options
DC, AC, and TRAN
The analysis types whose results are displayed during the simulation. You only need to specify one analysis type per .WATCH command, but there can be a .WATCH command for each analysis type in the circuit.

`<output variable>`
A maximum of eight output variables are allowed on a single .WATCH statement.

`<lower limit value>,<upper limit value>`
Specifies the normal operating range of that particular output variable. If the range is exceeded during the simulation, the simulator beeps and pauses. At this point, the simulation can be canceled or continued. If continued, the check for that output variable's boundary condition is eliminated. Each output variable can have its own value range.
Comments

The first example displays three output variables on the screen. The first variable, V(3), has an operating range set from minus one volt to four volts. If during the simulation the voltage at node three exceeds four volts, the simulation will pause. If the simulation is allowed to proceed, and node three continues to rise in value, the simulation is then not interrupted. However, if the simulation is allowed to continue and V(3) falls below -1.0 volt, the simulation would again pause because a new boundary condition was exceeded.

Up to three output variables can be seen on the display at one time. More than three variables can be specified, but they are not all displayed.

The possible output variables are given in .PROBE (Probe), with the exception that digital nodes cannot be used and group delay is not available.
.WCASE (sensitivity/worst-case analysis)

**Purpose**
The .WCASE statement causes a sensitivity and worst-case analysis of the circuit to be performed.

**General form**
```
.WCASE <analysis> <output variable> <function> [option]*
```

**Examples**
```
.WCASE TRAN V(5) YMAX
.WCASE DC IC(Q7) YMAX VARY DEV
.WCASE AC VP(13,5) YMAX DEVICES RQ OUTPUT ALL
.WCASE TRAN V([OUT1],[OUT2]) YMAX RANGE(.4u,.6u)
   + LIST OUTPUT ALL VARY DEV HI
```

**Arguments and options**

- `<analysis>`
  Only one of DC, AC, or TRAN must be specified for `<analysis>`. This analysis is repeated in subsequent passes of the worst-case analysis. All requested analyses are performed during the nominal pass. Only the selected analysis is performed during subsequent passes.

- `<output variable>`
  Identical in format to that of a .PRINT (print) output variable.

- `<function>`
  Specifies the operation to be performed on the values of the `<output variable>` to reduce these to a single value. This value is the basis for the comparisons between the nominal and subsequent runs. The `<function>` must be one of the following:

<table>
<thead>
<tr>
<th>Function</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>YMAX</td>
<td>Find the absolute value of the greatest difference in each waveform from the nominal run.</td>
</tr>
<tr>
<td>MAX</td>
<td>Find the maximum value of each waveform.</td>
</tr>
<tr>
<td>MIN</td>
<td>Find the minimum value of each waveform.</td>
</tr>
</tbody>
</table>
### Function | Meaning
--- | ---
RISE_EDGE(<value>) | Find the *first occurrence* of the waveform crossing *above* the threshold <value>. The waveform must have one or more points at or below <value> followed by one above; the output value listed is where the waveform increases above <value>.  
FALL_EDGE(<value>) | Find the *first occurrence* of the waveform crossing *below* the threshold <value>. The waveform must have one or more points at or above <value> followed by one below; the output value listed is where the waveform decreases below <value>.  
[option] *  
Could have any number of the following.

<table>
<thead>
<tr>
<th>[option]</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST</td>
<td>Prints the updated model parameters for the sensitivity analysis. This does not affect the Probe data generated by the simulation.</td>
</tr>
<tr>
<td>OUTPUT ALL</td>
<td>Prints output from the sensitivity runs, after the nominal (first) run. The output from any run is governed by the .PRINT, .PLOT, and .PROBE command in the file. If OUTPUT ALL is omitted, then only the nominal and worst-case runs produce output. OUTPUT ALL ensures that all sensitivity information is saved for Probe.</td>
</tr>
<tr>
<td>RANGE(^1)</td>
<td>Restricts the range over which &lt;function&gt; can be evaluated. An asterisk * can be used in place of a &lt;value&gt; to show for all values. For example see the next two rows.</td>
</tr>
<tr>
<td>(&lt;low value&gt;, &lt;high value&gt;)</td>
<td></td>
</tr>
<tr>
<td>YMAX</td>
<td>The maximum of the output variable is found for values of the sweep variable of -1 or more.</td>
</tr>
<tr>
<td>YMAX RANGE(*,.5)</td>
<td>YMAX is evaluated for values of the sweep variable (e.g., time, and frequency) of .5 or less.</td>
</tr>
<tr>
<td>MAX RANGE(-1,*)</td>
<td>The maximum of the output variable is found for values of the sweep variable of -1 or more.</td>
</tr>
<tr>
<td>HI or LOW</td>
<td>Specify the direction which &lt;function&gt; should move for the worst-case run is to go (relative to the nominal). If &lt;function&gt; is YMAX or MAX, the default is HI, otherwise the default is LOW.</td>
</tr>
<tr>
<td>VARY DEV</td>
<td></td>
</tr>
<tr>
<td>VARY LOT</td>
<td></td>
</tr>
<tr>
<td>VARY BOTH</td>
<td>By default, any device which has a model parameter specifying either a DEV tolerance or a LOT tolerance is included in the analysis. The analysis can be limited to only those devices which have DEV or LOT tolerances by specifying the appropriate option. The default is VARY BOTH. When VARY BOTH is used, sensitivity to parameters using both DEV and LOT specifications is checked only with respect to LOT variations. The parameter is then maximized or minimized using both DEV and LOT tolerances for the worst-case. All devices referencing the model have the same parameter values for the worst-case simulation.</td>
</tr>
</tbody>
</table>
By default, all devices are included in the sensitivity and worst-case analyses. The devices considered can be limited by listing the device types after the keyword `DEVICES`. Do not use any spaces or tabs in the devices list. For example, to only perform the analysis on resistors and MOSFETs, enter:

```
DEVICES RM
```

1. If `RANGE` is omitted, then `<function>` is evaluated over the whole sweep range. This is equivalent to `RANGE(*,*)`.

**Comments**

Multiple runs of the selected analysis (DC, AC, or transient) are performed while parameters are varied. Unlike `.MC (Monte Carlo analysis)`, `.WCASE` varies only one parameter per run. This allows PSpice to calculate the sensitivity of the output waveform to each parameter. Once all the sensitivities are known, one final run is performed using all parameters varied so as to produce the worst-case waveform. The sensitivity and worst-case runs are performed using variations on model parameters as specified by the `DEV` and `LOT` tolerances on each `.MODEL (model definition)` parameter. Other specifications on the `.WCASE` command control the output generated by the analysis.

**Note:** You can run either `.MC` or `.WCASE` for a circuit, but not both in the same circuit.
(comment)

Purpose
A statement beginning with an asterisk * is a comment line, which PSpice ignores.

General
form
* [any text]

Examples
* This is an example of
   * a multiple-line comment

Comments
Use an asterisk at the beginning of each line you want to be a comment. A single asterisk does not extend to subsequent lines. For example:

   * .MODEL ABC NMOS (. . . + . . . )

produces an error message, because the second line is not covered by the first asterisk.

The use of comment statements throughout the input is recommended. It is good practice to insert a comment line just before a subcircuit definition to identify the nodes, for example:

   * +IN -IN V+ V- +OUT -OUT
   .SUBCKT OPAMP 100 101 1 2 200 201

or to identify major blocks of circuitry.
; (in-line comment)

Purpose  A semicolon ; is treated as the end of a line.

General  circuit file text ;[any text]
form

Examples  R13  6 8 10 ; R13 is a
          ; feedback resistor
          C3  15  0 .1U ; decouple supply

Comments  The simulator moves on to the next line in the circuit file. The text on the line after the semicolon ; is a comment and has no effect. The use of comments throughout the input is recommended. This type of comment can also replace comment lines, which must start with * in the first column.

          Trailing in-line comments that extend to more that one line can use a semicolon to mark the beginning of the subsequent comment lines, as shown in the example.
### + (line continuation)

**Purpose**
A plus sign `+` is treated as the continuation of the previous line.

**General form**
```
circuit file text
+ more text
```

**Examples**
```
.DISTRIBUTION bi_modal (-1,1) (-.5,1) (-.5,0) (.5,0)
+ (.5,1) (1,1)
```

**Comments**
Because the simulator reads the line preceded by a plus sign as a continuation of the previous line, you can use the plus sign to break up long lines of command text.
Differences between PSpice and Berkeley SPICE2

The version of SPICE2 referred to is SPICE2G.6 from the University of California at Berkeley.

PSpice runs any circuit that SPICE2 can run, with these exceptions:

1. Circuits that use .DISTO (small-signal distortion) analysis. U.C. Berkeley SPICE supports the .DISTO analysis, but contains errors. Also, the special distortion output variables (e.g., HD2 and DIM3) are not available. Instead of the .DISTO analysis, we recommend running a transient analysis and looking at the output spectrum using the Fourier transform mode in Probe. This technique shows the distortion (spectral) products for both small-signal and large-signal distortion.

2. These options on the .OPTIONS (analysis options) statement are not available in PSpice:
   - LIMTIM: it is assumed to be 0.
   - LVLCOD: no in-line machine code is generated.
   - METHOD: a combination of trapezoidal and gear integration is always used.
   - MAXORD: a combination of trapezoidal and gear integration is always used.
   - LVLTIM: truncation error time step control is always used.
   - ITL3: truncation error time step control is always used.

3. The IN= option on the .WIDTH statement is not available. PSpice always reads the entire input file regardless of how long the input lines are.

4. Voltage coefficients for capacitors, and current coefficients for inductors must be put into a .MODEL (model definition) statement instead of on the device statement.

5. PSpice does not allow the use of nested subcircuit definitions.

   If this construct is used:
   
   ```
   .SUBCKT ABC 1 2 3
   ...
   .SUBCKT DEF 4 5 6
   ...
   .ENDS
   ...
   .ENDS
   ```

   It is recommended that the definitions be separated into:

   ```
   .SUBCKT ABC 1 2 3
   ...
   X1 ... DEF
   ```
Note: You can nest subcircuit calls.

6. The `.ALTER` command is not supported in PSpice. Instead, use the `.STEP (parametric analysis)` on page 99 command to modify specific parameters over multiple PSpice runs.

7. The syntax for the one-dimensional POLY form of E, F, G, and H (Voltage-controlled voltage source Voltage-controlled current source on page 165 and Current-controlled current source Current-controlled voltage source on page 174) devices is different. PSpice requires a dimension specification of the form POLY(1), while SPICE does not.

PSpice produces basically the same results as SPICE. There can be some small differences, especially for values crossing zero, due to the corrections made for convergence problems.

The semiconductor device models are the same as in SPICE.
## Analog devices

<table>
<thead>
<tr>
<th>Letter</th>
<th>Device type</th>
<th>Letter</th>
<th>Device type</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>GaAsFET on page 137</td>
<td>N</td>
<td>Digital input (N device) on page 434</td>
</tr>
<tr>
<td>C</td>
<td>Capacitor on page 156</td>
<td>O</td>
<td>Digital output (O Device) on page 439</td>
</tr>
<tr>
<td>D</td>
<td>Diode on page 160</td>
<td>Q</td>
<td>Bipolar transistor on page 272</td>
</tr>
<tr>
<td>E</td>
<td>Voltage-controlled voltage source Voltage-controlled current source on page 165</td>
<td>R</td>
<td>Resistor on page 302</td>
</tr>
<tr>
<td>F</td>
<td>Flux source on page 176</td>
<td>S</td>
<td>Voltage-Controlled switch on page 307</td>
</tr>
<tr>
<td>G</td>
<td>Current-controlled current source Voltage-controlled voltage source on page 174</td>
<td>T</td>
<td>Transmission line on page 312</td>
</tr>
<tr>
<td>H</td>
<td>Voltage-controlled voltage source Voltage-controlled current source on page 165</td>
<td>U</td>
<td>Digital primitive summary on page 345</td>
</tr>
<tr>
<td>Letter</td>
<td>Device type</td>
<td>Letter</td>
<td>Device type</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------</td>
<td>--------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>I</td>
<td><strong>Independent current source &amp; stimulus Independent voltage source &amp; stimulus on page 180</strong></td>
<td>U</td>
<td><strong>Stimulus devices on page 417</strong></td>
</tr>
<tr>
<td>J</td>
<td><strong>Junction FET on page 194</strong></td>
<td>V</td>
<td><strong>Independent voltage source &amp; stimulus on page 322</strong></td>
</tr>
<tr>
<td></td>
<td><strong>Coupling on page 202</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K</td>
<td><strong>Transmission line coupling on page 216</strong></td>
<td>W</td>
<td><strong>Current-Controlled switch on page 322</strong></td>
</tr>
<tr>
<td>L</td>
<td><strong>Inductor on page 219</strong></td>
<td>X</td>
<td><strong>Subcircuit instantiation on page 329</strong></td>
</tr>
<tr>
<td>M</td>
<td><strong>MOSFET on page 224</strong></td>
<td>Z</td>
<td><strong>IGBT on page 332</strong></td>
</tr>
</tbody>
</table>
Analog devices

This chapter describes the different types of analog devices supported by PSpice and PSpice A/D. These device types include analog primitives, independent and controlled sources, and subcircuit calls. Each device type is described separately, and each description includes the following information as applicable:

- A description and an example of the proper netlist syntax.
- The corresponding model types and their description.
- The corresponding list of model parameters and their descriptions.
- The equivalent circuit diagram and characteristic equations for the model (as required).
- References to publications that the model is based on.

These analog devices include all of the standard circuit components that normally are not considered part of the two-state (binary) devices that are found in the digital devices.

The model library consists of analog models of off-the-shelf parts that you can use directly in your circuit designs. Refer to the online Library List for available device models and the libraries they are located in. You can also implement models using the .MODEL (model definition) on page 58 statement and implement macromodels as subcircuits using the .SUBCKT (subcircuit) on page 105 statement.

The Device types on page 134 summary table lists all of the analog device primitives supported by PSpice A/D. Each primitive is described in detail in the sections following the table.
Device types

PSpice supports Bipolar transistor on page 272, many types of analog devices, including sources and general subcircuits. PSpice A/D also supports digital devices. The supported devices are categorized into device types, each of which can have one or more model types. For example, the BJT device type has three model types: NPN, PNP, and LPNP (Lateral PNP). The description of each devices type includes a description of any of the model types it supports.

The device declarations in the netlist always begin with the name of the individual device (instance). The first letter of the name determines the device type. What follows the name depends on the device type and its requested characteristics. Below is a summary of the device types and the general form of their declaration formats.

Note: The table below includes the designator (letter) used in device modeling for each device type.

### Analog device summary

<table>
<thead>
<tr>
<th>Device type</th>
<th>Letter</th>
<th>Declaration format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bipolar transistor on page 272</td>
<td>Q</td>
<td>Q&lt;name&gt; &lt;collector node&gt; &lt;base node&gt; &lt;emitter node&gt; + [substrate node] &lt;model name&gt; [area value]</td>
</tr>
<tr>
<td>Capacitor on page 156</td>
<td>C</td>
<td>C&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; [model name] &lt;value&gt; + [IC=&lt;initial value&gt;]</td>
</tr>
<tr>
<td>Voltage-controlled voltage source Voltage-controlled current source on page 165</td>
<td>E</td>
<td>E&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; &lt;+ controlling node&gt; + &lt;- controlling node&gt; &lt;gain&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(additional Analog Behavioral Modeling forms: VALUE, TABLE, LAPLACE, FREQ, and CHEBYSHEV; additional POLY form)</td>
</tr>
<tr>
<td>Voltage-controlled voltage source Voltage-controlled current source on page 165</td>
<td>G</td>
<td>G&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; &lt;+ controlling node&gt; + &lt;- controlling node&gt; &lt;transconductance&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(additional Analog Behavioral Modeling forms: VALUE, TABLE, LAPLACE, FREQ, and CHEBYSHEV; additional POLY form)</td>
</tr>
<tr>
<td>Current-controlled current source Current-controlled voltage source on page 174</td>
<td>F</td>
<td>F&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; &lt;controlling V device name&gt; + &lt;gain&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(additional POLY form)</td>
</tr>
<tr>
<td>Device type</td>
<td>Letter</td>
<td>Declaration format</td>
</tr>
<tr>
<td>----------------------------------------------------</td>
<td>--------</td>
<td>------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Current-Controlled switch on page 322</td>
<td>W</td>
<td>W&lt;name&gt; &lt;+ switch node&gt; &lt;+ switch node&gt; + &lt;controlling V device name&gt; &lt;model name&gt;</td>
</tr>
<tr>
<td>Current-controlled current source Current-controlled voltage source on page 174</td>
<td>H</td>
<td>H&lt;name&gt; &lt;+ node&gt; &lt;+ node&gt; &lt;controlling V device name&gt; + &lt;transresistance&gt; (additional POLY form)</td>
</tr>
<tr>
<td>Digital input (N device) on page 434</td>
<td>N</td>
<td>N&lt;name&gt; &lt;interface node&gt; &lt;low level node&gt; &lt;high level node&gt; + &lt;model name&gt; &lt;input specification&gt;</td>
</tr>
<tr>
<td>Digital output (O Device) on page 439</td>
<td>O</td>
<td>O&lt;name&gt; &lt;interface node&gt; &lt;low level node&gt; &lt;high level node&gt; + &lt;model name&gt; &lt;output specification&gt;</td>
</tr>
<tr>
<td>Digital primitive summary on page 345</td>
<td>U</td>
<td>U&lt;name&gt; &lt;primitive type&gt; ([parameter value]<em>) + &lt;digital power node&gt; &lt;digital ground node&gt; &lt;node&gt;</em> + &lt;timing model name&gt;</td>
</tr>
<tr>
<td>Stimulus devices on page 417*</td>
<td>U</td>
<td>U&lt;name&gt; STIM (&lt;width value&gt;, &lt;format value&gt;) + &lt;digital power node&gt; &lt;digital ground node&gt; &lt;node&gt;* + I/O model name [TIMESTEP=&lt;stepsize value&gt;] + &lt;waveform description&gt;</td>
</tr>
<tr>
<td>Diode on page 160</td>
<td>D</td>
<td>D&lt;name&gt; &lt;anode node&gt; &lt;cathode node&gt; &lt;model name&gt; [area value]</td>
</tr>
<tr>
<td>GaAsFET on page 137</td>
<td>B</td>
<td>B&lt;name&gt; &lt;drain node&gt; &lt;gate node&gt; &lt;source node&gt; + &lt;model name&gt; [area value]</td>
</tr>
<tr>
<td>Independent current source &amp; stimulus</td>
<td>I</td>
<td>I&lt;name&gt; &lt;+ node&gt; &lt;+ node&gt; [DC &lt;value&gt;] + [AC &lt;magnitude value&gt; [phase value]] [transient specification]</td>
</tr>
<tr>
<td>Independent voltage source &amp; stimulus on page 180</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Analog device summary, continued

<table>
<thead>
<tr>
<th>Device type</th>
<th>Letter</th>
<th>Declaration format</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Independent current source &amp; stimulus</strong></td>
<td>V</td>
<td>V&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; [[DC] &lt;value&gt;] + [AC &lt;magnitude value&gt; [phase value]] [transient specification]</td>
</tr>
<tr>
<td><strong>Independent voltage source &amp; stimulus</strong></td>
<td></td>
<td>on page 180</td>
</tr>
<tr>
<td><strong>Inductor</strong> on page 219</td>
<td>L</td>
<td>L&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; [model name] &lt;value&gt; + [IC=&lt;initial value&gt;]</td>
</tr>
<tr>
<td><strong>Coupling</strong> on page 202</td>
<td>K</td>
<td>K&lt;name&gt; L&lt;inductor name&gt; &lt;L&lt;inductor name&gt;&gt;* + &lt;coupling value&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>K&lt;name&gt; &lt;L&lt;inductor name&gt;&gt;* &lt;coupling value&gt; + &lt;model name&gt; [size value]</td>
</tr>
<tr>
<td><strong>IGBT</strong> on page 332</td>
<td>Z</td>
<td>Z&lt;name&gt; &lt;collector&gt; &lt;gate&gt; &lt;emitter&gt; [model name] + [AREA=&lt;value&gt;] [WB=&lt;value&gt;] [AGD=&lt;value&gt;] + [KP=&lt;value&gt;] [TAU=&lt;value&gt;]</td>
</tr>
<tr>
<td><strong>Junction FET</strong> on page 194</td>
<td>J</td>
<td>J&lt;name&gt; &lt;drain node&gt; &lt;gate node&gt; &lt;source node&gt; + [area value]</td>
</tr>
<tr>
<td><strong>MOSFET</strong> on page 224</td>
<td>M</td>
<td>M&lt;name&gt; &lt;drain node&gt; &lt;gate node&gt; &lt;source node&gt; + [bulk/substrate node] [model name] + [common model parameter]*</td>
</tr>
<tr>
<td><strong>Resistor</strong> on page 302</td>
<td>R</td>
<td>R&lt;name&gt; &lt;+ node&gt; &lt;- node&gt; [model name] &lt;value&gt; + [TC=&lt;linear temp. coefficient&gt; [,&lt;quadratic temp. coefficient&gt;]</td>
</tr>
<tr>
<td><strong>Subcircuit instantiation</strong> on page 329</td>
<td>X</td>
<td>X&lt;name&gt; [node]* &lt;subcircuit name&gt; + [PARAMS: &lt;&lt;name&gt;=&lt;value&gt;&gt;<em>] [TEXT:&lt;&lt;name&gt;=&lt;text value&gt;&gt;</em>]</td>
</tr>
<tr>
<td><strong>Transmission line</strong> on page 312</td>
<td>T</td>
<td>T&lt;name&gt; &lt;A port + node&gt; &lt;A port - node&gt; + &lt;B port + node&gt; &lt;B port - node&gt; [ideal or lossy specification]</td>
</tr>
<tr>
<td><strong>Transmission line coupling</strong> on page 216</td>
<td>K</td>
<td>K&lt;name&gt; T&lt;line name&gt; &lt;T&lt;line name&gt;&gt;* + CM=&lt;coupling capacitance&gt; LM=&lt;coupling inductance&gt;</td>
</tr>
<tr>
<td><strong>Voltage-Controlled switch</strong> on page 307</td>
<td>S</td>
<td>S&lt;name&gt; &lt;+ switch node&gt; &lt;- switch node&gt; + &lt;+ controlling node&gt; &lt;- controlling node&gt; [model name]</td>
</tr>
</tbody>
</table>
GaAsFET

General form  

\[ \text{B}<\text{name}> \text{ <drain node> <gate node> <source node> <model name> [area value]} \]

Examples  

\[ \text{BIN 100 10 0 GFAST} \]
\[ \text{B13 22 14 23 GNOM 2.0} \]

Model form  

\[ .\text{MODEL <model name> GASFET [model parameters]} \]

Description  
The GaAsFET is modeled as an intrinsic FET using an ohmic resistance \((R_D/\text{area})\) in series with the drain, another ohmic resistance \((R_S/\text{area})\) in series with the source, and another ohmic resistance \((R_G)\) in series with the gate.

Arguments and options  

\[ [\text{area value}] \]

The relative device area. Its default value is 1.0.
Comments

The LEVEL model parameter selects among different models for the intrinsic GaAsFET as follows:

- LEVEL=1 “Curtice” model (see reference [1])
- LEVEL=2 “Raytheon” or “Statz” model (see reference [3]), equivalent to the GaAsFET model in SPICE3
- LEVEL=3 “TOM” model by TriQuint (see reference [4])
- LEVEL=4 “Parker-Skellern” model (see reference [5] and [6])
- LEVEL=5 “TOM-2” model by TriQuint (see reference [7])
- LEVEL=6 “TOM-3” model by TriQuint

For more information, see References on page 155.

Note: The TOM-2 model is based on the original TriQuint TOM model, retaining the desirable features of the TOM model, while improving accuracy in the subthreshold near cutoff and knee regions (Vds of 1 volt or less). Included are temperature coefficients related to the drain current and corrected behavior of the capacitance as a function of temperature.

Note: The TOM-3 model uses quasi-static charge conservation in the implanted layer of MESFET to improve the accuracy of the capacitance equations.

Capture parts

The following table lists the set of GaAsFET breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

<table>
<thead>
<tr>
<th>part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBREAK</td>
<td>GASFET</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>GASFET model name</td>
</tr>
</tbody>
</table>

Setting operating temperature

Operating temperature can be set to be different from the global circuit temperature by defining one of the model parameters: T_ABS, T_REL_GLOBAL, or T_REL_LOCAL.
Additionally, model parameters can be assigned unique measurement temperatures using the T_MEASURED model parameter.
# Model Parameters

Table 2-1 GaAsFET model parameters for all levels

<table>
<thead>
<tr>
<th>Model parameter¹</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>flicker noise exponent</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>BETA</td>
<td>transconductance coefficient</td>
<td>amp/volt²</td>
<td>0.1</td>
</tr>
<tr>
<td>BETATCE</td>
<td>BETA exponential temperature coefficient</td>
<td>%/°C</td>
<td>0</td>
</tr>
<tr>
<td>CDS</td>
<td>drain-source capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>CGD</td>
<td>zero-bias gate-drain p-n capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>CGS</td>
<td>zero-bias gate-source p-n capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>EG</td>
<td>band gap voltage (barrier height)</td>
<td>eV</td>
<td>1.11</td>
</tr>
<tr>
<td>FC</td>
<td>forward-bias depletion capacitance coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>IS</td>
<td>gate p-n saturation current</td>
<td>amp</td>
<td>1E-14</td>
</tr>
<tr>
<td>KF</td>
<td>flicker noise coefficient</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>LEVEL</td>
<td>model index (1, 2, 3, 4, or 5)</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>gate p-n emission coefficient</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>RD</td>
<td>drain ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>RG</td>
<td>gate ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>RS</td>
<td>source ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>TRD1</td>
<td>RD temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>TRG1</td>
<td>RG temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>TRS1</td>
<td>RS temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>T_ABS</td>
<td>absolute temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_MEASURED</td>
<td>measured temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_GLOBAL</td>
<td>relative to current temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_LOCAL</td>
<td>relative to AKO model temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>VBI</td>
<td>gate p-n potential</td>
<td>volt</td>
<td>1.0</td>
</tr>
<tr>
<td>VTO</td>
<td>pinchoff voltage</td>
<td>volt</td>
<td>-2.5</td>
</tr>
</tbody>
</table>
Table 2-1  GaAsFET model parameters for all levels

<table>
<thead>
<tr>
<th>Model parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTOTC</td>
<td>$V_{TO}$ temperature coefficient</td>
<td>volt/°C</td>
<td>0</td>
</tr>
<tr>
<td>XTI</td>
<td>$I_S$ temperature exponent</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

1. For information on T_ABS, T_MEASURED, T_REL_GLOBAL, and T_REL_LOCAL, see the .MODEL (model definition) on page 58 statement.

Table 2-2  GaAsFET model parameters specific to model levels

<table>
<thead>
<tr>
<th>Model parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level 1</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALPHA</td>
<td>saturation voltage parameter</td>
<td>volt$^{-1}$</td>
<td>2.0</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>channel-length modulation</td>
<td>volt$^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>M</td>
<td>gate p-n grading coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>TAU</td>
<td>conduction current delay time</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td><strong>level 2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALPHA</td>
<td>saturation voltage parameter</td>
<td>volt$^{-1}$</td>
<td>2.0</td>
</tr>
<tr>
<td>B</td>
<td>doping tail extending parameter</td>
<td>volt$^{-1}$</td>
<td>0.3</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>channel-length modulation</td>
<td>volt$^{-1}$</td>
<td>0</td>
</tr>
<tr>
<td>M</td>
<td>gate p-n grading coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>TAU</td>
<td>conduction current delay time</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>VDELTA</td>
<td>capacitance transition voltage</td>
<td>volt</td>
<td>0.2</td>
</tr>
<tr>
<td>VMAX</td>
<td>capacitance limiting voltage</td>
<td>volt</td>
<td>0.5</td>
</tr>
<tr>
<td><strong>level 3</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ALPHA</td>
<td>saturation voltage parameter</td>
<td>volt$^{-1}$</td>
<td>2.0</td>
</tr>
<tr>
<td>BTRK</td>
<td>auxiliary parameter for Monte Carlo analysis*</td>
<td>amp/volt$^3$</td>
<td>0</td>
</tr>
<tr>
<td>DELTA</td>
<td>output feedback parameter</td>
<td>(amp-volt)$^{-1}$</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 2-2 GaAsFET model parameters specific to model levels

<table>
<thead>
<tr>
<th>Model parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DVT</td>
<td>auxiliary parameter for Monte Carlo analysis*</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>DVTT</td>
<td>auxiliary parameter for Monte Carlo analysis*</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>GAMMA</td>
<td>static feedback parameter</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>M</td>
<td>gate p-n grading coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>Q</td>
<td>power-law parameter</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>TAU</td>
<td>conduction current delay time</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>VDELT A</td>
<td>capacitance transition voltage</td>
<td>volt</td>
<td>0.2</td>
</tr>
<tr>
<td>VMAX</td>
<td>gate diode capacitance limiting voltage</td>
<td>volt</td>
<td>0.5</td>
</tr>
<tr>
<td><strong>level 4</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACGAM</td>
<td>capacitance modulation</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>DELTA</td>
<td>output feedback parameter</td>
<td>(amp·volt)^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>HFETA</td>
<td>high-frequency VGS feedback parameter</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>HFE1</td>
<td>HFGAM modulation by VGD</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>HFE2</td>
<td>HFGAM modulation by VGS</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>HFGAM</td>
<td>high-frequency VGD feedback parameter</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>HFG1</td>
<td>HFGAM modulation by VSG</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>HFG2</td>
<td>HFGAM modulation by VDG</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>IBD</td>
<td>gate junction breakdown current</td>
<td>amp</td>
<td>0</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>channel-length modulation</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>LFGAM</td>
<td>low-frequency feedback parameter</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>LFG1</td>
<td>LFGAM modulation by VSG</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>LFG2</td>
<td>LFGAM modulation by VDG</td>
<td>volt^{-1}</td>
<td>0</td>
</tr>
<tr>
<td>MVST</td>
<td>subthreshold modulation</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 2-2  GaAsFET model parameters specific to model levels

<table>
<thead>
<tr>
<th>Model parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>MXI</td>
<td>saturation knee-potential modulation</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>P</td>
<td>linear-region power law exponent</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>Q</td>
<td>power-law parameter</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>TAUD</td>
<td>relaxation time for thermal reduction</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TAUG</td>
<td>relaxation time for GAM feedback</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>VBD</td>
<td>gate junction breakdown potential</td>
<td>volt</td>
<td>1</td>
</tr>
<tr>
<td>VST</td>
<td>subthreshold potential</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>XC</td>
<td>capacitance pinchoff reduction factor</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>XI</td>
<td>saturation knee potential factor</td>
<td></td>
<td>1000</td>
</tr>
<tr>
<td>Z</td>
<td>knee transition parameter</td>
<td></td>
<td>0.5</td>
</tr>
</tbody>
</table>

**level 5**

<table>
<thead>
<tr>
<th>Model parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALPHA</td>
<td>saturation voltage parameter</td>
<td>volt⁻¹</td>
<td>2.0</td>
</tr>
<tr>
<td>ALPHATCE</td>
<td>ALPHA temperature coefficient</td>
<td>%/°C</td>
<td>0</td>
</tr>
<tr>
<td>BTRK</td>
<td>auxiliary parameter for Monte Carlo analysis¹</td>
<td>amp/volt³</td>
<td>0</td>
</tr>
<tr>
<td>CGDTCE</td>
<td>CGD temperature coefficient</td>
<td>°C⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>CGSTCE</td>
<td>CGS temperature coefficient</td>
<td>°C⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>DELTA</td>
<td>output feedback parameter</td>
<td>(amp-volt)⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>DVT</td>
<td>auxiliary parameter for Monte Carlo analysis*</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>DVTT</td>
<td>auxiliary parameter for Monte Carlo analysis*</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>GAMMA</td>
<td>static feedback parameter</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>GAMMATCE</td>
<td>GAMMA temperature coefficient</td>
<td>°C⁻¹</td>
<td>0</td>
</tr>
</tbody>
</table>
### Table 2-2 GaAsFET model parameters specific to model levels

<table>
<thead>
<tr>
<th>Model parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ND</td>
<td>subthreshold slope drain pull parameter</td>
<td>volt⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>NG</td>
<td>subthreshold slope gate parameter</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Q</td>
<td>power-law parameter</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>TAU</td>
<td>conduction current delay time</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>VBITC</td>
<td>vbi temperature coefficient</td>
<td>volt/°C</td>
<td>0</td>
</tr>
<tr>
<td>VDELT A</td>
<td>capacitance transition voltage</td>
<td>volt</td>
<td>0.2</td>
</tr>
<tr>
<td>VMAX</td>
<td>gate diode capacitance limiting voltage level 6</td>
<td>volt</td>
<td>0.5</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>Slope of drain characteristic in the saturated region.</td>
<td>volt⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>VST</td>
<td>Subthreshold slope</td>
<td>volt</td>
<td>1.0</td>
</tr>
<tr>
<td>MST</td>
<td>Subthreshold slope drain parameter</td>
<td>volt⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>IS</td>
<td>Forward gate diode saturation current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>N</td>
<td>Forward gate diode ideality factor</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>ILK</td>
<td>Leakage diode current parameter</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>PLK</td>
<td>Leakage diode potential parameter</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>K</td>
<td>Knee-function parameter</td>
<td></td>
<td>2.0</td>
</tr>
<tr>
<td>IS</td>
<td>As defined in TOM 2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
| Temperature Parameters
| MSTTC           | Linear temperature coefficient                                        |           | 0.0     |
| VSTTC           | Linear temperature coefficient                                        |           | 0.0     |
Auxiliary model parameters BTRK, DVT, and DVTT

The parameters BTRK, DVT, and DVTT are auxiliary model parameters that are used to make the Monte Carlo analysis easier when using PSpice. In the analysis, these affect the parameters VTO and BETA as follows:

\[
\begin{align*}
\text{VTO} &= \text{VTO} + \text{DVT} + \text{DVTT} \\
\text{BETA} &= \text{BETA} + \text{BTRK} \cdot (\text{DVT} + \text{DVTT})
\end{align*}
\]

In Monte Carlo analysis, DEV tolerances placed on the DVT or DVTT cause variations in both VTO and BETA. PSpice does not support correlated DEV variations in Monte Carlo analysis. Without DVT and DVTT, DEV tolerances placed on VTO and BETA can result in independent variations; there is a definite correlation between VTO and BETA on real devices.

The BTRK, DVT, and DVTT parameters are also used to provide tracking between distinct GaAsFETs, such as between depletion mode and enhancement mode. PSpice already provides a limited mechanism for this, but only allows one DEV and one LOT (or LOT/n and DEV/n) tolerance per model parameter. The added parameters circumvent this restriction by extending the capability of Monte Carlo to model correlation between the critical model parameters.
GaAsFET equations

The equations in this section describe an N-channel GaAsFET. The following variables are used:

- $V_{gs}$ = intrinsic gate-intrit = area·(nsic source voltage
- $V_{gd}$ = intrinsic gate-intrinsic drain voltage
- $V_{ds}$ = intrinsic drain-intrinsic source voltage
- $C_{ds}$ = drain-source capacitance
- $C_{gs}$ = gate-source capacitance
- $C_{gd}$ = gate-drain capacitance
- $V_t$ = $kT/q$ (thermal voltage)
- $k$ = Boltzmann constant
- $q$ = electron charge
- $T$ = analysis temperature (°K)

$T_{nom}$ = nominal temperature (set by using .OPTIONS (analysis options) on page 71 $T_{NOM}$=)

Note: Positive current is current flowing into a terminal (for example, positive drain current flows from the drain through the channel to the source).

GaAsFET equations for DC current: all levels

- $I_g$ = gate current = area·($I_{gs}$+$I_{gd}$)
- $I_d$ = drain current = area·($I_{drain}$-$I_{gd}$)
- $I_s$ = source current = area·($-I_{drain}$-$I_{gs}$)

  where

- $I_{gs}$ = gate-source leakage current
- $I_{gd}$ = gate-drain leakage current
GaAsFET equations for DC current: specific to model levels

Levels 1, 2, 3, and 5

\[ I_{gs} = IS \cdot (e^{Vgs/(N \cdot Vt)} - 1) \]
\[ I_{gd} = IS \cdot (e^{Vgd/(N \cdot Vt)} - 1) \]

Level 4

\[ I_{gs} = I_{gsf} + I_{gsr} \]

where

\[ I_{gsf} = IS \cdot \left( \frac{Vgs}{N \cdot Vt} - 1 \right) + Vgs \cdot \text{GMIN} \]
\[ I_{gsr} = IBD \cdot \left(1 - e^{\frac{-Vgs}{VBD}}\right) \]

\[ I_{gd} = I_{gdf} + I_{gdr} \]

where

\[ I_{gdf} = IS \cdot \left( \frac{Vgd}{N \cdot Vt} - 1 \right) + Vgd \cdot \text{GMIN} \]
\[ I_{gdr} = IBD \cdot \left(1 - e^{\frac{-Vgd}{VBD}}\right) \]

Level 1: Idrain

Normal mode: \( Vds \geq 0 \)

Case 1

for cutoff region: \( Vgs - VTO < 0 \)

then: \( Idrain = 0 \)

Case 2

for linear & saturation region: \( Vgs - VTO \geq 0 \)

then:

\[ Idrain = \beta \cdot (1 + \lambda \cdot Vds) \cdot (Vgs - VTO)^2 \cdot \text{tanh} (\alpha \cdot Vds) \]

Inverted mode: \( Vds < 0 \)
Switch the source and drain in the Normal mode equations.

**Level 2: Idrain**

**Normal mode: Vds ≥ 0**

**Case 1**

for cutoff region: \(Vgs - VTO < 0\)

then: \(Idrain = 0\)

**Case 2**

for linear & saturation region: \(Vgs - VTO ≥ 0\)

then:

\[
Idrain = BETA \cdot (1+\text{LAMBDA} \cdot Vds) \cdot (Vgs-VTO)^2 \cdot K_t / (1+B. (Vgs-VTO))
\]

Where

\(K_t\) is a polynomial approximation of \(\tanh\).

for linear region:

\[
0 < Vds < 3/\text{ALPHA}
\]

then

\[
K_t = 1 - (1 - Vds \cdot \text{ALPHA}/3)^3
\]

for saturation region:

\[
Vds ≥ 3/\text{ALPHA}
\]

then

\[
K_t = 1
\]

**Inverted mode: Vds < 0**

Switch the source and drain in the Normal mode equations.

**Level 3: Idrain**

**Normal mode: Vds ≥ 0**

**Case 1**

for cutoff region: \(Vgs - VTO < 0\)

then: \(Idrain = 0\)

**Case 2**

for linear & saturation region: \(Vgs - VTO ≥ 0\)
then:

\[ I_{drain} = \frac{I_{ds}}{1 + \Delta \cdot V_{ds} \cdot I_{ds}} \]

Where

\[ I_{ds} = \text{BETA} \cdot (V_{gs} - V_{to})^Q \cdot K_t \]

and

\[ V_{to} = V_{TO} - \text{GAMMA} \cdot V_{ds} \]

where

\[ K_t \] is same as of level 2.

**Level 4: Idrain**

**Normal mode: \( V_{ds} \geq 0 \)**

\[ I_{drain} = \frac{I_{ds}}{1 + \Delta \cdot P_{avg}} \]

\[ V_{gst} = V_{gs} - V_{TO} - \gamma_{lf} \cdot V_{gd}_{avg} - \gamma_{hf} \cdot (V_{gd} - V_{gd}_{avg}) - \eta_{hf} \cdot (V_{gs} - V_{gs}_{avg}) \]

\[ V_{dst} = V_{ds} \]

**Inverted mode: \( V_{ds} < 0 \)**

\[ I_{drain} = \frac{-I_{ds}}{1 + \Delta \cdot P_{avg}} \]

\[ V_{gst} = V_{gd} - V_{TO} - \gamma_{lf} \cdot V_{gd}_{avg} - \gamma_{hf} \cdot (V_{gs} - V_{gd}_{avg}) - \eta_{hf} \cdot (V_{gd} - V_{gs}_{avg}) \]

\[ V_{dst} = -V_{ds} \]

where

\[ I_{ds} = \text{BETA} \cdot (1 + \text{LAMBDA} \cdot V_{dst}) \cdot (V_{gt}^Q - (V_{gt} - V_{dt})^Q) \]

\[ P_{avg} = V_{ds} \cdot I_{ds} - \text{TAUD} \cdot \frac{d}{dt}(P_{avg}) \]

\[ \gamma_{lf} = \text{LFGAM} - \text{LFG1} \cdot V_{gs}_{avg} - \text{LFG2} \cdot V_{gd}_{avg} \]

\[ V_{gd}_{avg} = V_{gd} - \text{TAUG} \cdot \frac{d}{dt}V_{gd}_{avg} \quad \text{if: } V_{gd} \leq V_{gs} \]

\[ = V_{gs} - \text{TAUG} \cdot \frac{d}{dt}V_{gd}_{avg} \quad \text{if: } V_{gs} < V_{gd} \]

\[ \gamma_{hf} = \text{HFGAM} - \text{HFG1} \cdot V_{gs}_{avg} - \text{HFG2} \cdot V_{gd}_{avg} \]
\[ \eta f = HFETA + HFE1 \cdot V_{gd_{avg}} + HFE2 \cdot V_{gs_{avg}} \]

\[ V_{gs_{avg}} = V_{gs} - TAUG \cdot \frac{d}{dt} V_{gs_{avg}} \quad \text{if: } V_{gd} < V_{gs} \]
\[ = V_{gd} - TAUG \cdot \frac{d}{dt} V_{gs_{avg}} \quad \text{if: } V_{gs} < V_{gd} \]

\[ V_{gt} = VST \cdot (1 + MVST \cdot V_{dst}) \cdot \ln \left( \exp \left( \frac{V_{gst}}{VST \cdot (1 + MVST \cdot V_{dst})} \right) + 1 \right) \]

\[ V_{dt} = \frac{1}{2} \cdot \sqrt{(V_{dp} \cdot \sqrt{1 + Z + V_{sat}})^2 + Z \cdot V_{sat}^2} - \frac{1}{2} \cdot \sqrt{(V_{dp} \cdot \sqrt{1 + Z - V_{sat}})^2 + Z \cdot V_{sat}^2} \]

\[ V_{dp} = V_{dst} \cdot \left( \frac{V_{gt}}{V_{BI} - V_{TO}} \right)^{P - Q} \]

\[ V_{sat} = \frac{V_{gt} \cdot (V_{gt} \cdot MXI + XI \cdot (V_{BI} - V_{TO}))}{V_{gt} + V_{gt} \cdot MXI + XI \cdot (V_{BI} - V_{TO})} \]

**Level 5: Idrain**

**Normal mode: Vds \geq 0**

**Case 1**

For cutoff region: \( V_{gs} - V_{TO} + \text{GAMMA} \cdot V_{ds} \leq 0 \) AND \( \text{NG} + \text{ND} \cdot V_{ds} = 0 \)

then: \( I_{drain} = 0 \)

For linear and saturation region:

\( V_{gs} - V_{TO} + \text{GAMMA} \cdot V_{ds} > 0 \) OR \( \text{NG} + \text{ND} \cdot V_{ds} \neq 0 \)

then: \( I_{drain} = I_{ds0} / (1 + \text{DELTA} \cdot V_{ds} \cdot I_{ds0}) \)

where

\[ I_{ds0} = \frac{\text{BETA} \cdot (V_{g})^Q \cdot \frac{\text{ALPHA} \cdot V_{ds}}{\sqrt{1 + (\text{ALPHA} \cdot V_{ds})^2}}}{\sqrt{1 + (\text{ALPHA} \cdot V_{ds})^2}} \]
\[ V_{gs} = \frac{q}{V_{st}} \cdot \log \left( \exp \left( \frac{V_{gs} - (V_{TO} + \text{GAMMA} \cdot V_{ds})}{q \cdot V_{st}} \right) + 1 \right) \]

\[ V_{st} = \left( N_G + N_D \cdot V_{ds} \right) \cdot \left( \frac{kT}{q} \right) \]

**Inverted mode: \( V_{ds} < 0 \)**
Switch the source and drain in the Normal mode equations.

**GaAsFET equations for capacitance**

All capacitances are between terminals of the intrinsic GaAsFET (i.e., to the inside of the ohmic drain, source, and gate resistances).

**All Levels**
For all conditions:
\[ C_{ds} = \text{area} \cdot C_{DS} \]

**Level 1**
\[ C_{gs} = \text{gate-source capacitance} \]
For: \( V_{gs} \leq FC \cdot VBI \)
\[ C_{gs} = \text{area} \cdot C_{GS} \cdot (1 - V_{gs}/VBI)^{-M} \]
For: \( V_{gs} > FC \cdot VBI \)
\[ C_{gs} = \text{area} \cdot C_{GS} \cdot (1 - FC)^{(1+M)} \cdot (1 - FC \cdot (1+M) + M \cdot V_{gs}/VBI) \]

**C_{gd} = gate-drain capacitance**
For: \( V_{gd} \leq FC \cdot VBI \)
\[ C_{gd} = \text{area} \cdot C_{GD} \cdot (1 - V_{gd}/VBI)^{-M} \]
For: \( V_{gd} > FC \cdot VBI \)
\[ C_{gd} = \text{area} \cdot C_{GD} \cdot (1 - FC)^{(1+M)} \cdot (1 - FC \cdot (1+M) + M \cdot V_{gd}/VBI) \]

**Levels 2, 3, and 5**
\[ C_{gs} = \text{gate-source capacitance} \]
\[ C_{gs} = \text{area} \cdot \left( C_{GS} \cdot K_2 \cdot K_1 / (1 - Vn/VBI) \right)^{1/2} + C_{GD} \cdot K_3 \]
\[ C_{gd} = \text{gate-drain capacitance} \]
\[ C_{gd} = \text{area} \cdot \left( C_{GS} \cdot K3 \cdot K1 / (1 - Vn/VBI)^{1/2} + C_{GD} \cdot K2 \right) \]

Where:
\[ K1 = \left( 1 + (Ve - VTO)/(Ve - VTO)^2 + V_{DELTA}^2 \right)^{1/2} / 2 \]
\[ K2 = \left( 1 + (Vgs - Vgd)/(Vgs - Vgd)^2 + (1/\alpha)^2 \right)^{1/2} / 2 \]
\[ K3 = \left( 1 - (Vgs - Vgd)/(Vgs - Vgd)^2 + (1/\alpha)^2 \right)^{1/2} / 2 \]

\[ Ve = (Vgs + Vgd + ((Vgs - Vgd)^2 + (1/\alpha)^2)^{1/2}) / 2 \]

if:
\[ (Ve + VTO + ((Ve - VTO)^2 + V_{DELTA}^2)^{1/2}) / 2 < V_{MAX} \]
then \[ Vn = (Ve + VTO + ((Ve - VTO)^2 + V_{DELTA}^2)^{1/2}) / 2 \]
else \[ Vn = V_{MAX} \]

Level 4
Note: Charge storage is implemented using a modified Statz model.

\[ C_{gs} = \text{gate-source capacitance} \]
\[ C_{gs} = \frac{1}{2} \cdot K1 \cdot \left( 1 + 2ACGAM + \frac{V_{ds}}{\sqrt{V_{ds}^2 + \alpha^2}} \right) + \frac{1}{2} \cdot C_{GD} \cdot \text{area} \cdot \left( 1 + 2ACGAM - \frac{V_{ds}}{\sqrt{V_{ds}^2 + \alpha^2}} \right) \]

\[ C_{gd} = \text{gate-drain capacitance} \]
\[ C_{gd} = \frac{1}{2} \cdot K1 \cdot \left( 1 - 2ACGAM - \frac{V_{ds}}{\sqrt{V_{ds}^2 + \alpha^2}} \right) + \frac{1}{2} \cdot C_{GD} \cdot \text{area} \cdot \left( 1 - 2ACGAM + \frac{V_{ds}}{\sqrt{V_{ds}^2 + \alpha^2}} \right) \]

where:
\[ K1 = \frac{1}{2} \cdot \frac{C_{GS}}{\sqrt{1 - V_{ge}/VBI}} \left[ 1 + XC + (1 - XC) \cdot \frac{V_{gn}}{\sqrt{V_{gn}^2 + 0.2^2}} \right] \]
if: \( V_x < FC \cdot VBI \)
then \( V_{ge} = V_x \)

if: \( V_x \geq FC \cdot VBI \)
then \( V_{ge} = \frac{VBI}{1 - \frac{4(1 - FC)^3}{2 - 3FC + \left(\frac{V_x}{VBI}\right)^2}} \)

\[
V_x = V_{gs} + ACGAM \cdot V_{ds} - \frac{1}{2} \left( V_{gn} - \sqrt{V_{gn}^2 + 0.2^2} \right) - \frac{1}{2} \left( V_{gn} - \sqrt{V_{gn}^2 + 0.2^2} \right)
\]

\[
V_{gn} = \left[ (V_{gs} + ACGAM) \cdot V_{ds} - VTO - \frac{1}{2} \left( V_{ds} - \sqrt{V_{ds}^2 + \alpha^2} \right) \right] \cdot (1 - XC)
\]

where

\[
\alpha = \frac{XI}{XI + 1} \cdot \frac{VBI - VTO}{2}
\]

Note: If the source and drain potentials swap, the model reverses over a range set by \( \alpha \). The model maintains a straight line relation between gate-source capacitance and gate bias in the region \( Vgs > FC \cdot VBI \).

**GaAsFET equations for temperature effect**

**All Levels**

\[
VTO(T) = VTO + VTOTC \cdot (T - Tnom)
\]

\[
BETA(T) = BETA \cdot 1.01^{BETATCE(T - Tnom)}
\]

\[
IS(T) = IS \cdot e^{(T/Tnom-1) \cdot EG/(N-Vt)} \cdot (T/Tnom)^{XTI/N}
\]

\[
RG(T) = RG \cdot (1 + TRG1 \cdot (T - Tnom))
\]

\[
RD(T) = RD \cdot (1 + TRD1 \cdot (T - Tnom))
\]

\[
RS(T) = RS \cdot (1 + TRS1 \cdot (T - Tnom))
\]

**Levels 1, 2, 3, and 4**
GaAsFET equations for noise

Noise is calculated assuming a 1.0-hertz bandwidth, using the following spectral power densities (per unit bandwidth).

Parasitic resistance thermal noise

\[ I_s^2 = 4 \cdot k \cdot T / (R_S/\text{area}) \]
\[ I_d^2 = 4 \cdot k \cdot T / (R_D/\text{area}) \]
\[ I_g^2 = 4 \cdot k \cdot T / R_G \]

Intrinsic GaAsFET shot and flicker noise

\[ I_d^2 = 4 \cdot k \cdot T \cdot \frac{g_m}{2/3} + K_F \cdot I_d^A F / \text{FREQUENCY} \]

where:

\[ g_m = \frac{dI_{\text{drain}}}{dV_{gs}} \text{ (at the DC bias point)} \]
References

For more information on this GaAsFET model, refer to:


Capacitor

**General form**

C<name> <(+ node) node> <(-) node> [model name] <value>
[IC=<initial value>]

**Examples**

CLOAD 15 0 20pF
C2 1 2 .2E-12 IC=1.5V
CFDBCK 3 33 CMOD 10pF

**Model form**

.MODEL <model name> CAP [model parameters]

![](15v --- 0v C Load)

**Arguments and options**

**(+) and (-) nodes**

Define the polarity when the capacitor has a positive voltage across it. The first node listed (or pin one in Capture) is defined as positive. The voltage across the component is therefore defined as the first node voltage, less the second node voltage.

* [model name]    
  If [model name] is left out, then <value> is the capacitance in farads. If [model name] is specified, then the value is given by the model parameters; see Capacitor value formula on page 159.

* <initial value>  
  The initial voltage across the capacitor during the bias point calculation. It can also be specified in a circuit file using a .IC command as follows:

  .IC V(+node, -node) <initial value>
Comments

Positive current flows from the (+) node through the capacitor to the (-) node. Current flow from the first node through the component to the second node is considered positive.

For details on using the .IC command in a circuit file, see .IC (initial bias point condition) on page 49 and refer to your PSpice User’s Guide for more information.

The initial voltage across the capacitor can also be set in Capture by using the IC1 part if the capacitor is connected to ground or by using the IC2 part for setting the initial conditions between two nodes. These parts can be found in SPECIAL.OLB.

For more information about setting initial conditions, refer to your PSpice User’s Guide.

Capture parts

For standard C parts, the effective value of the part is set directly by the VALUE property. For the variable capacitor, C_VAR, the effective value is the product of the base value (VALUE) and multiplier (SET).

In general, capacitors should have positive component values (VALUE property). In all cases, components must not be given a value of zero.

However, there are cases when negative component values are desired. This occurs most often in filter designs that analyze an RLC circuit equivalent to a real circuit. When transforming from the real to the RLC equivalent, it is possible to end up with negative component values.

PSpice A/D allows negative component values for bias point, DC sweep, AC, and noise analyses. A transient analysis may fail for a circuit with negative components. Negative capacitors may create instabilities in time that the analysis cannot handle.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>capacitor</td>
<td>VALUE</td>
<td>capacitance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IC</td>
<td>initial voltage across the capacitor during bias point calculation</td>
</tr>
</tbody>
</table>
Breakout parts

For non-stock passive and semiconductor devices, Capture provides a set of breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters. Another approach is to use the model editor to derive an instance model and customize this. For example, you could add device and/or lot tolerances to model parameters.

Basic breakout part names consist of the intrinsic PSpice A/D device letter plus the suffix BREAK. By default, the model name is the same as the part name and references the appropriate device model with all parameters set at their default. For instance, the DBREAK part references the DBREAK model which is derived from the intrinsic PSpice A/D D model (.MODEL DBREAK D).

For breakout part CBREAK, the effective value is computed from a formula that is a function of the specified VALUE property.

<table>
<thead>
<tr>
<th>Device type</th>
<th>Part name</th>
<th>Part library</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>capacitor</td>
<td>CBREAK</td>
<td>breakout.olb</td>
<td>VALUE</td>
<td>capacitance</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IC</td>
<td>initial voltage across the capacitor during bias point calculation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>MODEL</td>
<td>CAP model name</td>
</tr>
</tbody>
</table>

Capacitor model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C$</td>
<td>capacitance multiplier</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>
Capacitor equations

Capacitor value formula

If \( \text{[model name]} \) is specified, then the value is given by:

\[
\text{<value>} \cdot C \cdot (1+\text{VC}_1 \cdot \text{V}+\text{VC}_2 \cdot \text{V}^2) \cdot (1+\text{TC}_1 \cdot (\text{T-nom})+\text{TC}_2 \cdot (\text{T-nom})^2)
\]

where \( \text{<value>} \) is normally positive (though it can be negative, but not zero). \( \text{T-nom} \) is the nominal temperature (set using TNOM option).

Capacitor equation for noise

The capacitor does not have a noise model.
Diode

General form
D<name> <(+ node> <(-) node> <model name> [area value]

Examples
  DCLAMP 14  0 DMOD
  D13 15 17 SWITCH 1.5

Model form
.MODEL <model name> D [model parameters]

Description
The diode is modeled as an ohmic resistance (rs/area) in series with an intrinsic diode. Positive current is current flowing from the anode through the diode to the cathode.

Arguments and options

< (+) node>
The anode.

< (-) node>
The cathode.

[area value]
Scales IS, ISR, IKF, RS, CJO, and IBV, and has a default value of 1. IBV and BV are both specified as positive values.
Capture parts

The following table lists the set of diode breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DBREAK</td>
<td>D, X</td>
<td>MODEL</td>
<td>D model name</td>
</tr>
<tr>
<td>DBREAK3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBREAKCR</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBREAKVV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DBREAKZ</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Setting operating temperature

Operating temperature can be set to be different from the global circuit temperature by defining one of the model parameters: T_ABS, T_REL_GLOBAL, or T_REL_LOCAL. Additionally, model parameters can be assigned unique measurement temperatures using the T_MEASURED model parameter. For more information, see Special considerations on page 64.

Diode model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>flicker noise exponent</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>BV</td>
<td>reverse breakdown knee voltage</td>
<td>volt</td>
<td>infinite</td>
</tr>
<tr>
<td>CJO</td>
<td>zero-bias p-n capacitance</td>
<td>farad</td>
<td>0.0</td>
</tr>
<tr>
<td>EG</td>
<td>bandgap voltage (barrier height)</td>
<td>eV</td>
<td>1.11</td>
</tr>
<tr>
<td>FC</td>
<td>forward-bias depletion capacitance coeff</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>IBVL</td>
<td>low-level reverse breakdown knee current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>IBV</td>
<td>reverse breakdown knee current</td>
<td>amp</td>
<td>1E-10</td>
</tr>
<tr>
<td>IKF</td>
<td>high-injection knee current</td>
<td>amp</td>
<td>infinite</td>
</tr>
<tr>
<td>IS</td>
<td>saturation current</td>
<td>amp</td>
<td>1E-14</td>
</tr>
</tbody>
</table>
### Diode equations

The equations in this section use the following variables:

\[ V_d = \text{voltage across the intrinsic diode only} \]
Vt = $k \cdot \frac{T}{q}$ (thermal voltage)

$k$ = Boltzmann’s constant

$q$ = electron charge

$T$ = analysis temperature (°K)

$T_{\text{nom}}$ = nominal temperature (set using TNOM option)

Other variables are listed in Diode model parameters on page 161.

**Diode equations for DC current**

$$I_d = \text{area} \cdot (I_{\text{fwd}} - I_{\text{rev}})$$

$I_{\text{fwd}}$ = forward current = $I_{\text{fwd}} \cdot K_{\text{inj}} + I_{\text{rec}} \cdot K_{\text{gen}}$

$K_{\text{inj}}$ = high-injection factor = \((I_{KF}/(I_{KF}+I_{\text{norm}}))^{1/2}$$

else: $K_{\text{inj}} = 1$

$I_{\text{rec}}$ = recombination current = $I_{\text{rec}} = I_{\text{norm}} \cdot (e^{V_d/(N \cdot V_t)} - 1)$

$I_{\text{fwd}}$ = forward current = $I_{\text{fwd}} = I_{\text{fwd}} \cdot K_{\text{inj}}$ = high-injection factor = \((I_{KF}/(I_{KF}+I_{\text{norm}}))^{1/2}$$

$I_{\text{rev}}$ = reverse current = $I_{\text{rev}} = I_{\text{rev}}_{\text{high}} + I_{\text{rev}}_{\text{low}}$

$I_{\text{rev}}_{\text{high}} = I_{\text{fwd}} \cdot e^{(V_d + BV)/(NBV \cdot V_t)}$

$I_{\text{rev}}_{\text{low}} = I_{\text{fwd}} \cdot e^{(V_d + BV)/(NBVL \cdot V_t)}$

**Diode equations for capacitance**

$$C_d = C_t + \text{area} \cdot C_j$$

$C_t$ = transit time capacitance = $TT \cdot G_d$

$$G_d = \text{DC conductance} = \text{area} \cdot \frac{d(I_{\text{norm}} \cdot K_{\text{inj}} + I_{\text{rec}} \cdot K_{\text{gen}})}{dV_d}$$

$K_{\text{inj}}$ = high-injection factor

$$C_j = C_{JO} \cdot (1 - V_d/VJ)^M$$

$I_{\text{fwd}} \leq FC \cdot VJ$

$$C_j = C_{JO} \cdot (1 - FC)^{1+M} \cdot (1 - FC \cdot V_d/VJ)^{1+M}$$

$I_{\text{fwd}} > FC \cdot VJ$

$C_j = \text{junction capacitance}$
Diode equations for temperature effects

\[ IS(T) = IS \cdot e^{\left( T/T_{nom} - 1 \right) \cdot E_G(T) \cdot V_T \cdot (T/T_{nom})^{X_{TI/N}}} \]

\[ ISR(T) = ISR \cdot e^{\left( T/T_{nom} - 1 \right) \cdot E_G(NR \cdot V_T) \cdot (T/T_{nom})^{X_{TN/R}}} \]

\[ IKF(T) = IKF \cdot (1 + TIKF \cdot (T - T_{nom})) \]

\[ BV(T) = BV \cdot (1 + TBV_1 \cdot (T - T_{nom}) + TBV_2 \cdot (T - T_{nom})^2) \]

\[ RS(T) = RS \cdot (1 + TRS_1 \cdot (T - T_{nom}) + TRS_2 \cdot (T - T_{nom})^2) \]

\[ VJ(T) = VJ \cdot T/T_{nom} - 3 \cdot V_T \cdot ln(T/T_{nom}) - E_G(T) \cdot T/\sqrt{T/T_{nom}} + E_G(T) \]

\[ E_G(T) = \text{silicon bandgap energy} = 1.16 - .000702 \cdot T^2/(T+1108) \]

\[ CJO(T) = CJO \cdot (1 + M \cdot (.0004 \cdot (T - T_{nom}) + (1 - VJ(T)/VJ))) \]

Diode equations for noise

Noise is calculated assuming a 1.0-hertz bandwidth, using the following spectral power densities (per unit bandwidth).

- parasitic resistance thermal noise
  \[ I_{n}^2 = 4 \cdot k \cdot T/(RS/\text{area}) \]

- intrinsic diode shot and flicker noise
  \[ I_{n}^2 = 2 \cdot q \cdot I_D + KF \cdot I_D^{AF}/\text{FREQUENCY} \]

References

For a detailed description of p-n junction physics, refer to:


Voltage-controlled voltage source
Voltage-controlled current source

**General form**

```plaintext
E|G name node1 node2 controlling_node1 controlling_node2 + gain

E<name> <+ node> <-> node <+ controlling node> <-> controlling node

E<name> <+ node> <-> node POLY(<value>) + <+ controlling node> <-> controlling node >*

E<name> <+ node> <-> node VALUE = {<expression>} + [error= {<warn (<condition>, "<warning statement>")}

E<name> <+ node> <-> node TABLE { <expression> } = + [< input value>,<output value> >*

E<name> <+ node> <-> node LAPLACE { <expression> } = + { <transform> }

E<name> <+ node> <-> node FREQ { <expression> } = [KEYWORD] + <+ frequency value>,<magnitude value>,<phase value> >*

E<name> <+ node> <-> node CHEBYSHEV { <expression> } = + [...LP][HP][BP][BR],<cutoff frequencies>,<attenuation> >*
```
Examples

EBUFF 10 11 1 2 1.0
EAMP 13 0 POLY(1) 26 0 0 500
ENONLIN 100 101 POLY(2) 3 0 4 0 0.0 13.6 0.2 0.005
ESQROOT 5 0 VALUE = {5*SQRT(V(3,2))}
ET2 2 0 TABLE {V(ANODE,CATHODE)} = (0,0) (30,1)
ERC 5 0 LAPLACE {V(10)} = {1/(1+.001*s)}
ELOWPASS 5 0 FREQ {V(10)}=(0,0,0)(5kHz, 0,0)(6kHz -60, 0)
DELAY=3.2ms
ELOWPASS 5 0 CHEBYSHEV {V(10)} = LP 800 1.2K .1dB 50dB
Exy 0 7 F = 2*V(5)
Exy 3 0 FREQ VALUE = {V(10) | EXP(1)*(2)/(S+2)} error =
{warn(V(10)>10, "Voltage is greater than 10 volts.
Exy 3 0 FREQ VALUE = {V(10) | EXP(1)*(2)/(S+2)} error =
{error(V(10)>10, "Voltage greater than 10V. Cannot proceed.

GBUFF 10 11 1 2 1.0
GAMP 13 0 POLY(1) 26 0 0 500
GNONLIN 100 101 POLY(2) 3 0 4 0 0.0 13.6 0.2 0.005
GPSK 11 6 VALUE = {5MA*SIN(6.28*10kHz*TIME+V(3))}
GT ANODE CATHODE VALUE = {200E-6*PWR(V(1)*V(2),1.5)}
GLOSSY 5 0 LAPLACE {V(10)} = {exp(-sqrt(C*s*(R+L*s)))}

Description

The voltage-controlled voltage source (E) and the voltage-controlled current source (G) devices have the same syntax. For a voltage-controlled current source just substitute G for E. G generates a current, whereas E generates a voltage.
Arguments and options

POLY(<value>)

Specifies the number of dimensions of the polynomial. The number of pairs of controlling nodes must be equal to the number of dimensions.

(+) and (-) nodes

Output nodes. Positive current flows from the (+) node through the source to the (-) node.

<(+ controlling node) and <(-) controlling node>

Are in pairs and define a set of controlling voltages. A particular node can appear more than once, and the output and controlling nodes need not be different. The TABLE form has a maximum size of 2048 input/output value pairs.

FREQ

If a DELAY value is specified, the simulator modifies the phases in the FREQ table to incorporate the specified delay value. This is useful for cases of tables which the simulator identifies as being non-causal. When this occurs, the simulator provides a delay value necessary to make the table causal. The new syntax allows this value to be specified in subsequent simulation runs, without requiring the user to modify the table.

If a KEYWORD is specified for FREQ tables, it alters the values in the table. The KEYWORD can be one of the following:

- MAG causes magnitude of frequency response to be interpreted as a raw value instead of dB.
- DB causes magnitude to be interpreted as dB (the default).
- RAD causes phase to be interpreted in radians.
- DEG causes phase to be interpreted in degrees (the default).
- R_I causes magnitude and phase values to be interpreted as real and imaginary magnitudes.
error= {<warn|error (<condition>, "<warning message>")}}

This is an optional argument to be used when you want to throw a warning or an error message to the user. The `warn` and the `error` keywords indicate whether a warning or an error is to be displayed. While using this argument, ensure that the message statements are in one line. The message text should not be too long and entire argument including the message must be less than 132 characters.

During a simulation, if the warning condition specified in the Error statement is encountered, simulator displays the predefined warning message. In case error conditions are met, PSpice simulator will stop the simulation.

**Important**

Special characters, such as semi colon (;), comma (,), parenthesis, and curly braces should not be used within the error and warning message statements. In case any of these characters is used in the message statement, PSpice simulator will throw an invalid expression error.

**Caution**

*Error conditions are not supported for digital values*

**Comments**

The first form and the first two examples apply to the linear case; the second form and the third example are for the nonlinear case. The last five forms and examples are analog behavioral modeling (ABM) that have expression, look up table, Laplace transform, frequency response, and filtering. Refer to your *PSpice User’s Guide* for more information on analog behavioral modeling.

Chebyshev filters have two attenuation values, given in dB, which specify the pass band ripple and the stop band attenuation. They can be given in either order, but must appear after all of the cutoff frequencies have been given. Low pass (LP) and high pass (HP) have two cutoff frequencies, specifying the pass band and stop band edges, while band pass (BP) and band reject (BR) filters have four. Again, these can be given in any order.
Note: You can get a list of the filter Laplace coefficients for each stage by enabling the LIST option in the Simulation Settings dialog box. (Click the Options tab, then select the Output file Category and select Device Summary.) The output is written to the .out file after the simulation is complete.

For the linear case, there are two controlling nodes and these are followed by the gain. For all cases, including the nonlinear case (POLY), refer to your PSpice User’s Guide.

Expressions cannot be used for linear and polynomial coefficient values in a voltage-controlled voltage source device statement.
Basic SPICE polynomial expressions (POLY)

PSpice A/D (and SPICE) use the following syntax:

```plaintext
<controlled source> <connecting nodes> 
+ POLY(<dimension>) <controlling input> <coefficients>
```

where

- **<controlled source>** is `<[E][F][G][H]device name>`, meaning the device type is one of E, F, G, or H.
- **<connecting nodes>** specifies `<(+node_name, -node_name)>` pair between which the device is connected.
- **<dimension>** is the dimension `<value>` of the polynomial describing the controlling function.
- **<controlling input>** specifies `<(+node_name, -node_name)>*` pairs used as input to the voltage controlled source (device types E and G), or `<V device name>*` for the current controlled source (device types F and H), and where the number of controlling inputs for either case equals `<dimension>`.
- **<coefficients>** specifies the coefficient `<values>*` for the polynomial transfer function.

If the source is one-dimensional (there is only one controlling source), POLY(1) is required unless the linear form is used. If the source is multidimensional (there is more than one controlling source), the dimension needs to be included in the keyword, for instance POLY(2).

Caution must be exercised with the POLY form. For instance,

```plaintext
EWRONG 1 0 POLY(1) (1,0) .5 1.0
```

tries to set node 1 to .5 volts greater than node 1. In this case, any analyses which you specify will fail to calculate a result. In particular, PSpice A/D cannot calculate the bias point for a circuit containing EWRONG. This also applies to the VALUE form of EWRONG:

```plaintext
(EWRONG 1 0 VALUE = (0.5 * V(1)));
```
Basic controlled source properties

<table>
<thead>
<tr>
<th>Part name</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>GAIN</td>
<td>gain</td>
</tr>
<tr>
<td>F</td>
<td></td>
<td>gain</td>
</tr>
<tr>
<td>G</td>
<td></td>
<td>transconductance</td>
</tr>
<tr>
<td>H</td>
<td></td>
<td>transresistance</td>
</tr>
<tr>
<td>EPOLY, FPOLY, GPOLY, HPOLY</td>
<td>COEFF</td>
<td>polynomial coefficient</td>
</tr>
</tbody>
</table>

PSpice A/D has a built-in capability allowing controlled sources to be defined with a polynomial transfer function of any degree and any dimension. Polynomials have associated coefficients for each term. Consider a voltage-controlled source with voltages \( V_1, V_2, \ldots, V_n \). The coefficients are associated with the polynomial according to this convention:

\[
V_{out} = P_0 + P_1 \cdot V_1 + P_2 \cdot V_2 + \cdots + P_n \cdot V_n + P_{n+1} \cdot V_1 \cdot V_1 + P_{n+2} \cdot V_1 \cdot V_2 + \cdots + P_{n+n} \cdot V_1 \cdot V_n + P_{2n+1} \cdot V_2 \cdot V_2 + P_{2n+2} \cdot V_2 \cdot V_3 + \cdots + P_{2n+n-1} \cdot V_2 \cdot V_n + \cdots + P_{n!/(2(n-2)!)+2n} \cdot V_n \cdot V_n + P_{n!/(2(n-2)!)+2n+1} \cdot V_1^2 \cdot V_1 + P_{n!/(2(n-2)!)+2n+2} \cdot V_1^2 \cdot V_2 + \cdots + \cdots + \cdots 
\]

The above is written for a voltage-controlled voltage source, but the form is similar for the other sources.

The POLY device types shown in Basic controlled source properties on page 171 are defined with a dimension of one, meaning there is only one controlling source. However, similar devices can be defined of any degree and dimension by creating parts with appropriate coefficient and TEMPLATE properties and the appropriate number of input pins.

The current-controlled device models (F, FPOLY, H, and HPOLY) contain a current-sensing voltage source. When netlisted, they generate two device declarations to the circuit file set: one for the controlled source and one for the independent current-sensing voltage source.

When defining a current-controlled source part of higher dimension, the TEMPLATE property must account for the same number of current-sensing voltage sources (equal to the
dimension value). For example, a two dimensional current-controlled voltage source is described by the following polynomial equation:

\[ V_{\text{out}} = C_0 + C_1I_1 + C_2I_2 + C_{11}I_1^2 + C_{12}I_1I_2 + C_{22}I_2^2 \]

To create the two dimensional HPOLY2 part, these properties must be defined:

- COEFF0 = 1
- COEFF1 = 1
- COEFF2 = 1
- COEFF11 = 1
- COEFF12 = 1
- COEFF22 = 1
- COEFFS = @COEFF0 @COEFF1 @COEFF2 @COEFF11 @COEFF12 @COEFF22
- TEMPLATE = H^@REFDES %5 %6 POLY(2) VH1^@REFDES VH2^@REFDES
  + @COEFFS
  VH1^@REFDES %1 %2 0V
  VH2^@REFDES %3 %4 0V

The TEMPLATE definition is actually contained on a single line. The VH1 and VH2 fragments after the `\n` characters represent the device declarations for the two current-sensing voltage sources required by this part. Also, the part graphics must have the appropriate number of pins. When placing an instance of HPOLY2 in your schematic, the COEFF\(n\) properties must be appropriately set.

**Implementation examples**

Following are some examples of traditional SPICE POLY constructs and equivalent ABM parts which could be used instead.

**Example 1: four-input voltage adder**

This is an example of a device which takes four input voltages and sums them to provide a single output voltage.

The representative polynomial expression would be as follows:

\[ V_{\text{out}} = 0.0 + (1.0)V_1 + (1.0)V_2 + (1.0)V_3 + (1.0)V_4 \]

The corresponding SPICE POLY form would be as follows:

```
ESUM 100 101 POLY(4) (1,0) (2,0) (3,0) (4,0) 0.0 1.0 1.0
  + 1.0 1.0
```

This could be represented with a single ABM expression device configured with the following expression properties:

- EXP1 = V(1,0) +
- EXP2 = V(2,0) +
- EXP3 = V(3,0) +
- EXP4 = V(4,0)

Following template substitution for the ABM device, the output becomes:
Example 2: two-input voltage multiplier

This is an example of a device which takes two input voltages and multiplies them together resulting in a single output voltage.

The representative polynomial expression would be as follows:

\[ V_{\text{out}} = 0.0 + (0.0)V_1 + (0.0)V_2 + (0.0)V_1^2 + (1.0)V_1V_2 \]

The corresponding SPICE POLY form would be as follows:

```
EMULT 100 101 POLY(2) (1,0) (2,0) 0.0 0.0 0.0 0.0 1.0
```

This could be represented with a single MULT device. For additional examples of a voltage multiplier device, refer to the Analog Behavioral Modeling chapter of your PSpice User's Guide.

Example 3: voltage squarer

This is an example of a device that outputs the square of the input value.

For the one-dimensional polynomial, the representative polynomial expression reduces to:

\[ V_{\text{out}} = P_0 + P_1 \cdot V + P_2 \cdot V^2 + \ldots P_n \cdot V^n \]

The corresponding SPICE POLY form would be as follows:

```
ESQUARE 100 101 POLY(1) (1,0) 0.0 0.0 1.0
```

This could be represented by a single instance of the MULT part, with both inputs from the same net. This results in the following:

\[ V_{\text{out}} = (V_{\text{in}})^2 \]
Current-controlled current source
Current-controlled voltage source

General form

\[ F<\text{name}> \langle+\rangle \text{ node} \langle-\rangle \text{ node} \]
+ \langle\text{controlling V device name} \rangle \langle\text{gain}\rangle 
\[ F<\text{name}> \langle+\rangle \text{ node} \langle-\rangle \text{ node} \text{ POLY}(<\text{value}>)* \]
+ \langle\text{controlling V device name}\rangle* 
+ \langle <\text{polynomial coefficient value}> \rangle* 

Examples

\begin{align*}
\text{FSENSE 1 2 VSENSE 10.0} \\
\text{FAMP 13 0 POLY(1) VIN 0 500} \\
\text{FNONLIN 100 101 POLY(2) VCNTRL1 VCINTRL2 0.0 13.6 0.2 0.005}
\end{align*}

Description

The Current-Controlled Current Source (F) and the Current-Controlled Voltage Source (H) devices have the same syntax. For a Current-Controlled Voltage Source just substitute an H for the F. The H device generates a voltage, whereas the F device generates a current.

Arguments and options

\(+\) and \((-)\)

Output nodes. A positive current flows from the (+) node through the source to the (-) node. The current through the controlling voltage source determines the output current. The controlling source must be an independent voltage source (V device), although it need not have a zero DC value.

\text{POLY}(<\text{value}>)

Specifies the number of dimensions of the polynomial. The number of controlling voltage sources must be equal to the number of dimensions.

Comments

The first General form and the first two examples apply to the linear case. The second form and the last example are for the nonlinear case.

For the linear case, there must be one controlling voltage source and its name is followed by the gain. For all cases, including the nonlinear case (POLY), refer to your \textit{PSpice User's Guide}.

Note: In a current-controlled current source device statement, expressions cannot be used for linear and polynomial coefficient values.
Basic SPICE polynomial expressions (POLY)

For more information on the POLY form, see Basic SPICE polynomial expressions (POLY) on page 170.
Flux source

General form

\[ E\text{<name> <(+) <node> <(-) node> F} = \{ <expression> \} \]

\[ E\text{<name> <(+) <node> <(-) node> F} = \{ <expression> \} \\
+ \text{[error= \{<warn \text{ (<condition>, "<warning statement>"}) \\
+ \text{error= \{<error \text{ (<condition>, "<error statement>"})]]} \]

Examples

\[ E\text{bl 4b 0 F} = \{(1e-1)*I(Ebl))} \]

Description

The flux source can be modeled using a E- device. The output of a flux source modeled using a E device is a voltage calculated using the following equation.

\[ V = \frac{d\phi}{dt} \]

where \( \phi \) is the flux.

Arguments and options

\((+) \text{ and } (-) \text{ nodes}\)

Output nodes. Positive current flows from the (+) node through the source to the (-) node.

\( F \)

Specifies a Flux source.

\text{error= \{<warn|error \text{ (<condition>, "<statement>"})}}

This is an optional argument to be used when you want to throw a warning or an error message to the user. The \text{warn} and the \text{error} keywords indicate whether a warning or an error is to be displayed.

During simulation if the condition specified in the Error statement gets violated, simulator displays the predefined error or the warning statement.
To simulate a charge source, instantiate the FLUX_GEN part from the FUNCTION library in your design.

\[
\frac{d\Phi}{dt} = \text{FLUX} = ?
\]

FLUX_GEN

The value of the source is defined by the value of the CHARGE parameter.
Charge source

General form

\[ G<name> <(+) <node> <(-) node> Q = \{ <expression> \} \]

\[ G<name> <(+) <node> <(-) node> Q = \{ <expression> \} + [error= \{<warn \{<condition>, "<warning statement>"\})
+ error= \{<error \{<condition>, "<error statement>"\})\] \]

Examples

\[ Gbc 2b 0 Q=(1e-3)*V(2b) \]

Description

A charge source can be modeled using a G device. The output of such a device is current source, calculated using the following equation.

\[ I = \frac{dq}{dt} \]

where q is the charge.

Arguments and options

\((+)\) and \((-)\) nodes

Output nodes. Positive current flows from the \((+)\) node through the source to the \((-)\) node.

Q

Specifies a voltage controlled charge source

error= \{<warn|error \{<condition>, "<statement>"\})\]

This is an optional argument to be used when you want to throw a warning or an error message to the user. The warn and the error keywords indicate whether a warning or an error is to be displayed.

During simulation if the condition specified in the Error statement gets violated, simulator displays the predefined error or the warning statement.
To simulate a charge source, instantiate the CHARGE_GEN part from the FUNCTION library in your design.

The value of the source is defined by the value of the CHARGE parameter.
Independent current source & stimulus
Independent voltage source & stimulus

General form

\[ I\textless \text{name}\rangle \textless (+) \text{node}\rangle \textless (-) \text{node}\rangle \\
+ [ [DC] \textless \text{value}\rangle ] \\
+ [ AC \textless \text{magnitude value}\rangle \text{[phase value]} ] \\
+ [ \text{STIMULUS}=<\text{stimulus name}>] \\
+ [\text{transient specification}] \]

Examples

IBIAS 13 0 2.3mA
IAC 2 3 AC .001
IACPHS 2 3 AC .001 90
IPULSE 1 0 PULSE(-1mA 1mA 2ns 2ns 2ns 50ns 100ns)
I3 26 77 DC .002 AC 1 SIN(.002 .002 1.5MEG)

Description

This element is a current source. Positive current flows from the (+) node through the source to the (-) node: in the first example, IBIAS drives node 13 to have a negative voltage. The default value is zero for the DC, AC, and transient values. None, any, or all of the DC, AC, and transient values can be specified. The AC phase value is in degrees. The pulse and exponential examples are explained later in this section.

Note: The independent current source & stimulus (I) and the independent voltage source & stimulus (V) devices have the same syntax. For an independent voltage source & stimulus just substitute a V for the I. The V device functions identically and has the same syntax as the I device, except that it generates voltage instead of current.
The variables TSTEP and TSTOP, which are used in defaulting some waveform parameters, are set by the `.TRAN (transient analysis)` on page 113 command. TSTEP is `<print step value>` and TSTOP is `<final time value>`. The `.TRAN` command can be anywhere in the circuit file; it need not come after the voltage source.

### Arguments and options

<stimulus name>

References a `.STIMULUS (stimulus)` on page 104 definition.

[transient specification]

<table>
<thead>
<tr>
<th>Use this value...</th>
<th>To produce this result...</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>EXP (&lt;parameters&gt;)</code></td>
<td>an exponential waveform</td>
</tr>
<tr>
<td><code>PULSE (&lt;parameters&gt;)</code></td>
<td>a pulse waveform</td>
</tr>
<tr>
<td><code>PWL (&lt;parameters&gt;)</code></td>
<td>a piecewise linear waveform</td>
</tr>
<tr>
<td><code>SFFM (&lt;parameters&gt;)</code></td>
<td>a frequency-modulated waveform</td>
</tr>
<tr>
<td><code>SIN (&lt;parameters&gt;)</code></td>
<td>a sinusoidal waveform</td>
</tr>
</tbody>
</table>
Independent current source & stimulus (EXP)

**General form**

\[ \text{EXP}(i_1, i_2, td_1, tc_1, td_2, tc_2) \]

**Examples**

IRAMP 10 5 EXP(1 5 1 .2 2 .5)

**Waveform parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>(&lt;i_1&gt;)</td>
<td>Initial current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>(&lt;i_2&gt;)</td>
<td>Peak current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>(&lt;td_1&gt;)</td>
<td>Rise (fall) delay</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>(&lt;tc_1&gt;)</td>
<td>Rise (fall) time constant</td>
<td>sec</td>
<td>TSTEP</td>
</tr>
<tr>
<td>(&lt;td_2&gt;)</td>
<td>Fall (rise) delay</td>
<td>sec</td>
<td>(&lt;td_1&gt;)+TSTEP</td>
</tr>
<tr>
<td>(&lt;tc_2&gt;)</td>
<td>Fall (rise) time constant</td>
<td>sec</td>
<td>TSTEP</td>
</tr>
</tbody>
</table>
**Description**  
The EXP form causes the current to be \(<i1>\) for the first \(<td1>\) seconds. Then, the current decays exponentially from \(<i1>\) to \(<i2>\) using a time constant of \(<tc1>\). The decay lasts \(td2-td1\) seconds. Then, the current decays from \(<i2>\) back to \(<i1>\) using a time constant of \(<tc2>\). [Independent current source and stimulus, exponential waveform formulas](#) on page 183 describe the EXP waveform.

![Graph of EXP waveform](#)

**Table 2-3  Independent current source and stimulus, exponential waveform formulas**

<table>
<thead>
<tr>
<th>Time period</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 to (&lt;td1&gt;)</td>
<td>(&lt;i1&gt;)</td>
</tr>
<tr>
<td>(&lt;td1&gt;) to (&lt;td2&gt;)</td>
<td>(&lt;i1&gt; + (i2-i1) \cdot (1-e^{-(TIME-td1)/tc1}))</td>
</tr>
<tr>
<td>(&lt;td2&gt;) to TSTOP</td>
<td>(&lt;i1&gt; + (i2-i1) \cdot ((1-e^{-(TIME-td1)/tc1})-(1-e^{-(TIME-td2)/tc2})))</td>
</tr>
</tbody>
</table>
Independent current source & stimulus (PULSE)

**General form**

PULSE (<i1> <i2> <td> <tr> <tf> <pw> <per>)

**Examples**

ISW 10 5 PULSE(1A 5A 1sec .1sec .4sec .5sec 2sec)

**Waveform parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;i1&gt;</td>
<td>Initial current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>&lt;i2&gt;</td>
<td>Pulsed current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>&lt;td&gt;</td>
<td>Delay</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>&lt;tf&gt;</td>
<td>Fall time</td>
<td>sec</td>
<td>TSTEP</td>
</tr>
<tr>
<td>&lt;tr&gt;</td>
<td>Rise time</td>
<td>sec</td>
<td>TSTEP</td>
</tr>
<tr>
<td>&lt;pw&gt;</td>
<td>Pulse width</td>
<td>sec</td>
<td>TSTOP</td>
</tr>
<tr>
<td>&lt;per&gt;</td>
<td>Period</td>
<td>sec</td>
<td>TSTOP</td>
</tr>
</tbody>
</table>
**Description** The PULSE form causes the current to start at \( i_1 \), and stay there for \( t_d \) seconds. Then, the current goes linearly from \( i_1 \) to \( i_2 \) during the next \( t_r \) seconds, and then the current stays at \( i_2 \) for \( t_p \) seconds. Then, it goes linearly from \( i_2 \) back to \( i_1 \) during the next \( t_f \) seconds. It stays at \( i_1 \) for \( p_e-r-(t_r+p_w+t_f) \) seconds, and then the cycle is repeated except for the initial delay of \( t_d \) seconds. **Independent current source and stimulus pulse waveform formulas** describe the PULSE waveform.

![Pulse Waveform Diagram]

**Table 2-4  Independent current source and stimulus pulse waveform formulas**

<table>
<thead>
<tr>
<th>Time</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( i_1 )</td>
</tr>
<tr>
<td>( t_d )</td>
<td>( i_1 )</td>
</tr>
<tr>
<td>( t_d+t_r )</td>
<td>( i_2 )</td>
</tr>
<tr>
<td>( t_d+t_r+p_w )</td>
<td>( i_2 )</td>
</tr>
<tr>
<td>( t_d+t_r+p_w+t_f )</td>
<td>( i_1 )</td>
</tr>
<tr>
<td>( t_d+p_e )</td>
<td>( i_1 )</td>
</tr>
<tr>
<td>( t_d+p_e+t_r )</td>
<td>( i_2 )</td>
</tr>
</tbody>
</table>
Independent current source & stimulus (PWL)

**General form**

\[ \text{PWL} + \left[ \text{TIME SCALE FACTOR} = \langle \text{value} \rangle \right]\ + \left[ \text{VALUE SCALE FACTOR} = \langle \text{value} \rangle \right] + (\text{corner_points})^* \]

where corner_points are:

\( (\langle tn \rangle, \langle in \rangle) \) to specify a point

FILE <filename> to read point values from a file

REPEAT FOR <n> (corner_points)*

ENDREPEAT to repeat <n> times

REPEAT FOREVER (corner_points)*

ENDREPEAT to repeat forever

**Examples**

<p>| | | | | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>1</td>
<td>2</td>
<td>PWL</td>
<td>(0,1)</td>
<td>(1.2,5)</td>
<td>(1.4,2)</td>
<td>(2,4)</td>
<td>(3,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>v2</td>
<td>3</td>
<td>4</td>
<td>PWL</td>
<td>REPEAT FOR 5 ((1,0))(2,1)(3,0) ENDREPEAT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>v3</td>
<td>5</td>
<td>6</td>
<td>PWL</td>
<td>REPEAT FOR 5 FILE DATA1.TAB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>v4</td>
<td>7</td>
<td>8</td>
<td>PWL</td>
<td>TIME SCALE FACTOR=0.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>v5</td>
<td>7</td>
<td>8</td>
<td>PWL</td>
<td>REPEAT FOR 5 FILE DATA1.TAB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\( n \) volt square wave (where \( n \) is 1, 2, 3, 4, then 5); 75% duty cycle; 10 cycles; 1 microseconds per cycle:

```plaintext
.PARAM N=1
.STEP PARAM N 1,5,1
V1 1 0 PWL
+ TIME SCALE FACTOR=1e-6 ;all time units are scaled to microseconds
+ REPEAT FOR 10
+ (.25, 0)(.26, (N))(.99, (N))(1, 0)
+ ENDREPEAT
```
5 volt square wave; 75% duty cycle; 10 cycles; 10 microseconds per cycle; followed by 50% duty cycle $n$ volt square wave (where $n$ is 1, 2, 3, 4, then 5) lasting until the end of simulation:

\[
\text{.PARAM N=.2} \\
\text{.STEP PARAM N .2, 1.0, .2} \\
\text{V1 1 0 PWL} \\
+ \quad \text{TIME_SCALE_FACTOR=1e-5 ; all time units are scaled to 10 us} \\
+ \quad \text{VALUE_SCALE_FACTOR=5} \\
+ \quad \text{REPEAT FOR 10} \\
+ \quad (.25, 0)(.26, 1)(.99, 1)(1, 0) \\
+ \quad \text{ENDREPEAT} \\
+ \quad \text{REPEAT FOREVER} \\
+ \quad (+.50, 0) \\
+ \quad (+.01, \{N\}); iteration time .51 \\
+ \quad (+.48, \{N\}); iteration time .99 \\
+ \quad (1, 0) \\
+ \quad \text{ENDREPEAT}
\]

Assuming that a PWL specification has been given for a device to generate two triangular waveforms:

\[
\text{V3 1 0 PWL (1ms, 1)(2ms, 0)(3ms, 1)(4ms, 0)}
\]

Or, to replace the above with

\[
\text{V3 1 0 PWL FILE TRIANGLE.IN}
\]

where the file\textit{ triangle.in} would need to contain:

\[
(1ms, 1)(2ms, 0)(3ms, 1)(4ms, 0)
\]

### Waveform parameters

<table>
<thead>
<tr>
<th>Parameter(^1)</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$&lt;tn&gt;$</td>
<td>time at corner</td>
<td>seconds</td>
<td>none</td>
</tr>
<tr>
<td>$&lt;vn&gt;$</td>
<td>voltage at corner</td>
<td>volts</td>
<td>none</td>
</tr>
<tr>
<td>$&lt;n&gt;$</td>
<td>number of repetitions</td>
<td>positive integer, 0, or -1</td>
<td>none</td>
</tr>
</tbody>
</table>

1. $<tn>$ and $<n>$ cannot be expressions; $<vn>$ may be an expression.
The PWL form describes a piecewise linear waveform. Each pair of time-current values specifies a corner of the waveform. The current at times between corners is the linear interpolation of the currents at the corners.

### Arguments and options

- `<time_scale_factor>` and/or `<value_scale_factor>`
  
  Can be included immediately after the PWL keyword to show that the time and/or current value pairs are to be multiplied by the appropriate scale factor. These scale factors can be expressions, in which case they are evaluated once per outer simulation loop, and thus should be composed of expressions not containing references to voltages or currents.

- `<tn>` and `<in>`
  
  The transient specification corner points for the PWL waveform, as shown in the first example. The `<in>` can be an expression having the same restrictions as the scaling keywords, but `<tn>` must be a literal.
The text file that supplies the time-current (\( <tn> <in> \)) pairs. The contents of this file are read by the same parser that reads the circuit file, so that engineering units (e.g., 10us) are correctly interpreted. Note that the continuation + signs in the first column are unnecessary and therefore discouraged.

A typical file can be created by editing an existing PWL specification, replacing all + signs with blanks (to avoid unintentional +time). Only numbers (with units attached) can appear in the file; expressions for \( <tn> \) and \( <n> \) values are invalid. All absolute time points in \( <file \ name> \) are with respect to the last \( (tn \ in) \) entered. All relative time points are with respect to the last time point.

\section*{REPEAT \ ... \ ENDREPEAT}

These loops permit repetitions.

They can appear anywhere a \( (tn \ in) \) pair can appear. Absolute times within REPEAT loops are with respect to the start of the current iteration. The REPEAT \ ... \ ENDREPEAT specifications can be nested to any depth. Make sure that the current value associated with the beginning and ending time points (within the same REPEAT loop or between adjacent REPEAT loops), are the same when 0 is specified as the first point in a REPEAT loop.

\section*{A REPEAT FOR \(-1\) \ ... \ ENDREPEAT is treated as if it had been REPEAT FOREVER \ ... \ ENDREPEAT. A REPEAT FOR \(0\) \ ... \ ENDREPEAT is ignored (other than syntax checking of the enclosed corner points).
Independent current source & stimulus (SFFM)

**General form**

SFFM (<ioff> <iampl> <fc> <mod> <fm>)

**Examples**

IMOD 10 5 SFFM(2 1 8Hz 4 1Hz)

**Waveform parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ioff&gt;</td>
<td>offset current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>&lt;iampl&gt;</td>
<td>peak amplitude of current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>&lt;fc&gt;</td>
<td>carrier frequency</td>
<td>hertz</td>
<td>1/TSTOP</td>
</tr>
<tr>
<td>&lt;mod&gt;</td>
<td>modulation index</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>&lt;fm&gt;</td>
<td>modulation frequency</td>
<td>hertz</td>
<td>1/TSTOP</td>
</tr>
</tbody>
</table>

**Description**

The SFFM (Single-Frequency FM) form causes the current, as illustrated below, to follow the formula:

\[ i_{off} + i_{ampl} \cdot \sin(2\pi \cdot fc \cdot TIME + mod \cdot \sin(2\pi \cdot fm \cdot TIME)) \]
Independent current source & stimulus (SIN)

**General form**

```
SIN (<ioff> <iampl> <freq> <td> <df> <phase>)
```

**Examples**

```
ISIG 10 5 SIN(2 2 5Hz 1sec 1 30)
```

**Waveform parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;ioff&gt;</td>
<td>offset current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>&lt;iampl&gt;</td>
<td>peak amplitude of current</td>
<td>amp</td>
<td>none</td>
</tr>
<tr>
<td>&lt;freq&gt;</td>
<td>frequency</td>
<td>hertz</td>
<td>1/TSTOP</td>
</tr>
<tr>
<td>&lt;td&gt;</td>
<td>delay</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>&lt;df&gt;</td>
<td>damping factor</td>
<td>sec⁻¹</td>
<td>0</td>
</tr>
<tr>
<td>&lt;phase&gt;</td>
<td>phase</td>
<td>degree</td>
<td>0</td>
</tr>
</tbody>
</table>
Description

The sinusoidal (SIN) waveform causes the current to start at \(<i_{off}\>\) and stay there for \(<t_d>\) seconds. Then, the current becomes an exponentially damped sine wave. **Independent current source and stimulus sinusoidal waveform formulas** describe the SIN waveform.

The SIN waveform is for **transient analysis only**. It does not have any effect on AC analysis. To give a value to a current during AC analysis, use an AC specification, such as:

\[ IAC\ 3\ 0\ AC\ 1mA \]

where IAC has an amplitude of one milliampere during AC analysis, and can be zero during transient analysis. For transient analysis use, for example:

\[ ITRAN\ 3\ 0\ SIN(0\ 1mA\ 1kHz) \]

where ITRAN has an amplitude of one milliampere during transient analysis and is zero during AC analysis. Refer to your *PSpice User’s Guide*. 
### Table 2-5 Independent current source and stimulus sinusoidal waveform formulas

<table>
<thead>
<tr>
<th>Time period</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>to &lt;td&gt;</td>
<td>( i_{off} + i_{ampl} \cdot \sin(2\pi \cdot \text{phase}/360^\circ) )</td>
</tr>
<tr>
<td>&lt;td&gt; to TSTOP</td>
<td>( i_{off} + i_{ampl} \cdot \sin(2\pi \cdot (\text{freq} \cdot (\text{TIME}-\text{td}) + \text{phase}/360^\circ)) \cdot e^{(\text{TIME}-\text{td}) \cdot \text{df}} )</td>
</tr>
</tbody>
</table>
Junction FET

General form

```
J<name> <drain node> <gate node> <source node> <model name>
+ [area value]
```

Examples

```
JIN 100 1 0 JFAST
J13 22 14 23 JNOM 2.0
```

Model form

```
.MODEL <model name> NJF [model parameters]
.MODEL <model name> PJF [model parameters]
```

Description

The JFET is modeled as an intrinsic FET using an ohmic resistance \( \frac{RD}{\text{area}} \) in series with the drain, and using another ohmic resistance \( \frac{RS}{\text{area}} \) in series with the source. Positive current is current flowing into a terminal.

Arguments and options

```
[area value]
```

The relative device area. It has a default value of 1.0.
Capture parts

The following table lists the set of JFET breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JBREAKN</td>
<td>NJF</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>NJF model name</td>
</tr>
<tr>
<td>JBREAKP</td>
<td>PJF</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>PJF model name</td>
</tr>
</tbody>
</table>

Setting operating temperature

Operating temperature can be set to be different from the global circuit temperature by defining one of the model parameters: T_ABS, T_REL_GLOBAL, or T_REL_LOCAL. Additionally, model parameters can be assigned unique measurement temperatures using the T_MEASURED model parameter. For more information, see Model parameters.
# Model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>flicker noise exponent</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>ALPHA</td>
<td>ionization coefficient</td>
<td>volt^-1</td>
<td>0</td>
</tr>
<tr>
<td>BETA</td>
<td>transconductance coefficient</td>
<td>amp/volt</td>
<td>1E-4</td>
</tr>
<tr>
<td>BETATCE</td>
<td>BETA exponential temperature coefficient</td>
<td>%/°C</td>
<td>0</td>
</tr>
<tr>
<td>CGD</td>
<td>zero-bias gate-drain p-n capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>CGS</td>
<td>zero-bias gate-source p-n capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>FC</td>
<td>forward-bias depletion capacitance coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>IS</td>
<td>gate p-n saturation current</td>
<td>amp</td>
<td>1E-14</td>
</tr>
<tr>
<td>ISR</td>
<td>gate p-n recombination current parameter</td>
<td>amp</td>
<td>0</td>
</tr>
<tr>
<td>KF</td>
<td>flicker noise coefficient</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>channel-length modulation</td>
<td>volt^-1</td>
<td>0</td>
</tr>
<tr>
<td>M</td>
<td>gate p-n grading coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>N</td>
<td>gate p-n emission coefficient</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>NR</td>
<td>emission coefficient for isr</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>PB</td>
<td>gate p-n potential</td>
<td>volt</td>
<td>1.0</td>
</tr>
<tr>
<td>RD</td>
<td>drain ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>RS</td>
<td>source ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>T_ABS</td>
<td>absolute temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_MEASURED</td>
<td>measured temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_GLOBAL</td>
<td>relative to current temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_LOCAL</td>
<td>relative to AKO model temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>VK</td>
<td>ionization knee voltage</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>VTO</td>
<td>threshold voltage</td>
<td>volt</td>
<td>-2.0</td>
</tr>
<tr>
<td>VTOTC</td>
<td>VTO temperature coefficient</td>
<td>volt/°C</td>
<td>0</td>
</tr>
</tbody>
</table>
VTO < 0 means the device is a depletion-mode JFET (for both N-channel and P-channel) and VTO > 0 means the device is an enhancement-mode JFET. This conforms to U.C. Berkeley SPICE.

For information on T_MEAURED, T_ABS, T_REL_GLOBAL, and T_REL_LOCAL, see MODEL (model definition) on page 58..

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>XTI</td>
<td>IS temperature coefficient</td>
<td></td>
<td>3</td>
</tr>
</tbody>
</table>

Note: VTO < 0 means the device is a depletion-mode JFET (for both N-channel and P-channel) and VTO > 0 means the device is an enhancement-mode JFET. This conforms to U.C. Berkeley SPICE.
JFET equations

The equations in this section describe an N-channel JFET. For P-channel devices, reverse the sign of all voltages and currents.

The following variables are used:

\( V_{gs} \) = intrinsic gate-intrinsic source voltage  
\( V_{gd} \) = intrinsic gate-intrinsic drain voltage  
\( V_{ds} \) = intrinsic drain-intrinsic source voltage  
\( C_{gs} \) = gate-source capacitance  
\( C_{gd} \) = gate-drain capacitance  
\( Vt \) = \( k \cdot T/q \) (thermal voltage)  
\( k \) = Boltzmann’s constant  
\( q \) = electron charge  
\( T \) = analysis temperature (°K)  
\( T_{nom} \) = nominal temperature (set using TNOM option)

Other variables are listed in Model parameters.

Note: Positive current is current flowing into a terminal (for example, positive drain current flows from the drain through the channel to the source).

JFET equations for DC current

All levels

\( I_g = \text{gate current} = area \cdot (I_{gs} + I_{gd}) \)
\( I_{gs} = \text{gate-source leakage current} = I_n + I_r \cdot K_g \)
\( In = \text{normal current} = IS \cdot (e^{V_{gs}/(N \cdot Vt)} - 1) \)
\( I_r = \text{recombination current} = ISR \cdot (e^{V_{gs}/(NR \cdot Vt)} - 1) \)
\( K_g = \text{generation factor} = ((1 - V_{gs}/PB)^2 + 0.005)^{M/2} \)
\( I_{gd} = \text{gate-drain leakage current} = I_n + I_r \cdot K_g + I_i \)
\( In = \text{normal current} = IS \cdot (e^{V_{gd}/(N \cdot Vt)} - 1) \)
Ir = recombination current = $ISR \cdot (e^{\frac{V_{gd}}{NR \cdot Vt}} - 1)$

$Kg = \text{generation factor} = ((1-V_{gd}/PB)^2 + 0.005)^{\frac{M}{2}}$

$Ii = \text{impact ionization current}$

for forward saturation region:

$0 < V_{gs} - V_{TO} < V_{ds}$

then: $Ii = I_{drain} \cdot ALPHAp \cdot v_{dif} \cdot e^{-VK/v_{dif}}$

where $v_{dif} = V_{ds} - (V_{gs} - V_{TO})$

else: $Ii = 0$

$Id = \text{drain current} = area \cdot (I_{drain} - I_{gd})$

$Is = \text{source current} = area \cdot (-I_{drain} - I_{gs})$

**All levels: I_{drain}**

**Normal mode: V_{ds} \geq 0**

**Case 1**

for cutoff region: $V_{gs} - V_{TO} \leq 0$

then $I_{drain} = 0$

**Case 2**

for linear region: $V_{ds} < V_{gs} - V_{TO}$

then: $I_{drain} = BETA \cdot (1 + LAMBDA \cdot V_{ds}) \cdot V_{ds} \cdot (2 \cdot (V_{gs} - V_{TO}) - V_{ds})$

**Case 3**

for saturation region: $0 < V_{gs} - V_{TO} < V_{ds}$

then: $I_{drain} = BETA \cdot (1 + LAMBDA \cdot V_{ds}) \cdot (V_{gs} - V_{TO})^2$

**Inverted mode: V_{ds} < 0**

Switch the source and drain in the normal mode equations above.

**JFET equations for capacitance**

All capacitances are between terminals of the intrinsic JFET (that is, to the inside of the ohmic drain and source resistances).

**Gate-source depletion capacitance**
JFET equations for temperature effects

The drain and source ohmic (parasitic) resistances have no temperature dependence.

\[ V_{TO}(T) = V_{TO} + V_{TOTC} \cdot (T-T_{nom}) \]
\[ BETA(T) = BETA \cdot 1.01^{BETATCE(T-T_{nom})} \]
\[ IS(T) = IS \cdot e^{(T/T_{nom}-1) \cdot E_G(N \cdot Vt) \cdot (T/T_{nom}) \cdot XTI/N} \]
where \( E_G = 1.11 \)
\[ ISR(T) = ISR \cdot e^{(T/T_{nom}-1) \cdot E_G(NR \cdot Vt) \cdot (T/T_{nom}) \cdot XTI/NR} \]
where \( E_G = 1.11 \)
\[ PB(T) = PB \cdot T/T_{nom} - 3 \cdot Vt \cdot ln(T/T_{nom}) - E_g(T_{nom}) \cdot T/T_{nom} + E_g(T) \]
where
\[ E_g(T) = \text{silicon bandgap energy} = 1.16 - 0.000702 \cdot T^2/(T+1108) \]
\[ CGS(T) = CGS \cdot (1+M \cdot (0.004 \cdot (T-T_{nom}) + (1-PB(T)/PB))) \]
\[ CGD(T) = CGD \cdot (1+M \cdot (0.004 \cdot (T-T_{nom}) + (1-PB(T)/PB))) \]
JFET equations for noise

Noise is calculated assuming a 1.0-hertz bandwidth, using the following spectral power densities (per unit bandwidth).

Parasitic resistance thermal noise

\[ I_s^2 = 4 \cdot k \cdot T / (R_s/\text{area}) \]
\[ I_d^2 = 4 \cdot k \cdot T / (R_d/\text{area}) \]

Intrinsic JFET shot and flicker noise

\[ I_{\text{drain}}^2 = 4 \cdot k \cdot T \cdot \frac{g_m}{3} + K_F \cdot I_{\text{drain}} \frac{A_F}{FREQUENCY} \]

where \( g_m = \frac{dI_{\text{drain}}}{dV_{gs}} \) (at the DC bias point)

Reference

For more information about the U.C. Berkeley SPICE models, including the JFET device, refer to:

Coupling

**General form**

\[ K<\text{name}> L<\text{inductor name}> <L<\text{inductor name}>>* <\text{coupling value}> \]
\[ K<\text{name}> <L<\text{inductor name}>>*<\text{coupling value}><\text{model name}>[\text{size value}] \]
\[ K<\text{name}>T<\text{transmission line name}>T<\text{transmission line name}>+C_m=<\text{capacitive coupling}>L_m=<\text{Inductive coupling}> \]

**Examples**

- KTUNED L3OUT L4IN .8
- KTRNSFRM LPRIMARY LSECNDRY 1
- KXFRM L1 L2 L3 L4 .98 KPOT_3C8
- K2LINES T1 T2 Lm=1m Cm=.5p

**Model form**

- .MODEL <model name> CORE [model parameters]

**Description**

This device can be used to define coupling between inductors (transformers) or between transmission lines. This device also refers to a nonlinear magnetic core (CORE) model to include magnetic hysteresis effects in the behavior of a single inductor (winding), or in multiple coupled windings.
Inductor coupling

Arguments and options

```
K<name> L<inductor name>
```

Couples two or more inductors.

Using the "Dot" convention, place a "DOT" on the first node of each inductor. For example:

```
I1 1 0 AC 1mA
L1 1 0 10uH
L2 2 0 10uH
R2 2 0 .1
K12 L1 L2 1
```

The current through L2 is in the opposite direction as the current through L1. The polarity is determined by the order of the nodes in the L devices and not by the order of inductors in the K statement.

```
<coupling value>
```

This is the coefficient of mutual coupling, which must be between 0 and 1.0.

This coefficient is defined by the equation

```
<coupling value> = M_{ij}/(L_i \cdot L_j)^{1/2}
```

where

```
L_i, L_j = a coupled-pair of inductors
M_{ij} = the mutual inductance between L_i and L_j
```

For transformers of normal geometry, use 1.0 as the value. Values less than 1.0 occur in air core transformers when the coils do not completely overlap.
<model name>

If <model name> is present, four things change:

- The mutual coupling inductor becomes a nonlinear, magnetic core device. The magnetic core’s B-H characteristics are analyzed using either the Jiles-Atherton model (see Inductor coupling: Jiles-Atherton model) or the Spice Plus model (see “Inductor coupling: Spice Plus model”).

- The inductors become windings, so the number specifying inductance now specifies the number of turns.

- The list of coupled inductors could be just one inductor.

- A model statement is required to specify the model parameters.

[size value]

Has a default value of 1.0 and scales the magnetic cross-section. It is intended to represent the number of lamination layers, so only one model statement is needed for each lamination type. For example:

L1 5 9 20 ; inductor having 20 turns
K1 L1 1 K528T500_3C8 ; Ferroxcube toroid core
L2 3 8 15 ; primary winding having ; 15 turns
L3 4 6 45 ; secondary winding having ; 45 turns
K2 L2 L3 1 K528T500_3C8; another core (not the same as K1)
Here is a Probe B-H display of 3C8 ferrite (Ferroxcube).

**Comments**

The linear branch relation for transient analysis is

\[ V_i = L_i \frac{dI_i}{dt} + M_{ij} \frac{dI_j}{dt} + M_{ik} \frac{dI_k}{dt} + \cdots \]

For U.C. Berkeley SPICE2: if there are several coils on a transformer, then there must be K statements coupling all combinations of inductor pairs. For instance, a transformer using a center-tapped primary and two secondaries could be written:

* PRIMARY
  L1 1 2 10uH
  L2 2 3 10uH
* SECONDARY
  L3 11 12 10uH
  L4 13 14 10uH
* MAGNETIC COUPLING
  K12 L1 L2 1
  K13 L1 L3 1
  K14 L1 L4 1
  K23 L2 L3 1
  K24 L2 L4 1
  K34 L3 L4 1
This older technique is still supported, but *not required*, for simulation. The same transformer can also be written:

```
* PRIMARY
L1 1 2 10uH
L2 2 3 10uH
* SECONDARY
L3 11 12 10uH
L4 13 14 10uH
* MAGNETIC COUPLING
KALL L1 L2 L3 L4 1
```

Note: Do not mix the two techniques.

The simulator uses either the Jiles-Atherton model (see 2-5) or SpicePlus model (see “Inductor coupling: Spice Plus model.”) to analyze the B-H curve of the magnetic core and calculate values for inductance and flux for each of the windings.

The state of the nonlinear core can be viewed in Probe by specifying $B(Kxxx)$ for the magnetization or $H(Kxxx)$ for the magnetizing influence. These values are not available for .PRINT (print) on page 84 or .PLOT (plot) on page 82 output.
Capture parts

See your PSpice User’s Guide for information about using nonlinear magnetic cores with transformers.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Properties</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>XFRM_LINEAR</td>
<td>transformer</td>
<td>L1_VALUE</td>
<td>winding inductances in Henries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L2_VALUE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>COUPLING</td>
<td>coefficient of mutual coupling (must lie between 0 and 1)</td>
</tr>
<tr>
<td>K_LINEAR</td>
<td>transformer</td>
<td>Ln</td>
<td>inductor reference designator</td>
</tr>
<tr>
<td>XFRM_NONLINEAR</td>
<td>transformer</td>
<td>L1_TURNS</td>
<td>number of turns on each winding</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L2_TURNS</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>COUPLING</td>
<td>coefficient of mutual coupling (must lie between 0 and 1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>nonlinear CORE model name</td>
</tr>
</tbody>
</table>

Breakout parts

For non-stock passive and semiconductor devices, Capture provides a set of breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters. Another approach is to use the model editor to derive an instance model and customize this. For example, you could add device and/or lot tolerances to model parameters.

Basic breakout part names consist of the intrinsic PSpice A/D device letter plus the suffix BREAK. By default, the model name is the same as the part name and references the appropriate device model with all parameters set at their default. For instance, the DBREAK part references the DBREAK model which is derived from the intrinsic PSpice A/D D model (.MODEL DBREAK D)
Using the KBREAK part

The inductor coupling part, KBREAK, can be used to couple up to six independent inductors on a schematic. A MODEL property is provided for using nonlinear magnetic core (CORE) models, if desired. By default, KBREAK references the KBREAK model contained in breakout.lib; this model, in turn, uses the default CORE model parameters.

<table>
<thead>
<tr>
<th>Device type</th>
<th>Part name</th>
<th>Part library</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inductor coupling</td>
<td>KBREAK</td>
<td>breakout.lib</td>
<td>COUPLING</td>
<td>coupling factor</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Li</td>
<td>inductor reference</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>designator</td>
</tr>
</tbody>
</table>

The KBREAK part can be used to:

- Provide linear coupling between inductors.
- Reference a CORE model in a configured model library file.
- Define a user-defined CORE model with custom model parameter values.

The dot convention for the coupling is related to the direction in which the inductors are connected. The dot is always next to the first pin to be netlisted. For example, when the inductor part L is placed without rotation, the dotted pin is the left one. Rotate on the Edit menu (C+r) rotates the inductor +90°, making this pin the bottom pin.

For nonlinear coupling

L1 must have a value; the rest may be left blank. The model must reference a CORE model such as those contained in MAGNETIC.LIB or other user-defined models. VALUE is set to the number of windings.

For linear coupling

L1 and at least one other Li must have values; the rest may be left blank. The model reference must be blank. VALUE must be in Henries.
Inductor coupling (and magnetic core)

PSpice supports two models for inductor coupling. These are:

- **Inductor coupling: Jiles-Atherton model** on page 209
- **Inductor coupling: Spice Plus model** on page 211

**Inductor coupling: Jiles-Atherton model**

In PSpice, the Jiles-Atherton model is supported as level 2 model. The model parameters are listed below.

<table>
<thead>
<tr>
<th>Model parameters(^1)</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Thermal energy parameter</td>
<td>amp/meter</td>
<td>1E+3</td>
</tr>
<tr>
<td>AREA</td>
<td>Mean magnetic cross-section</td>
<td>cm(^2)</td>
<td>0.1</td>
</tr>
<tr>
<td>C</td>
<td>Domain flexing parameter</td>
<td></td>
<td>0.2</td>
</tr>
<tr>
<td>GAP</td>
<td>Effective air-gap length</td>
<td>cm</td>
<td>0</td>
</tr>
<tr>
<td>K</td>
<td>Domain anisotropy parameter</td>
<td>amp/meter</td>
<td>500</td>
</tr>
<tr>
<td>LEVEL</td>
<td>Model index</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>MS</td>
<td>Magnetization saturation</td>
<td>amp/meter</td>
<td>1E+6</td>
</tr>
<tr>
<td>PACK</td>
<td>Pack(^2) (stacking) factor</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>PATH</td>
<td>Mean magnetic path length</td>
<td>cm</td>
<td>1.0</td>
</tr>
</tbody>
</table>

1. See **.MODEL (model definition)** on page 58.
2. Flux is proportional to PACK.

The Jiles-Atherton model is based on existing ideas of domain wall motion, including flexing and translation. The model derives an anhysteric magnetization curve by using a mean field technique, in which any domain is coupled to the magnetic field (H) and the bulk magnetization (M). This anhysteric value is the magnetization that would be reached in the absence of domain wall pinning. Hysteresis is modeled by the effects of pinning of domain walls on material defect sites. This impedance to motion and flexing due to the differential field exhibits all of the main features of real, nonlinear magnetic devices, such as the initial magnetization curve (initial permeability), saturation of magnetization, coercivity, and hysteresis loss.
A magnetic material that is comprised of loosely coupled domains has an equilibrium B-H curve, called the anhysteric. This curve is the locus of B-H values generated by superimposing a DC magnetic bias and a large AC signal that decays to zero. It is the curve representing minimum energy for the domains and is modeled, in theory, by

$$M_{an} = MS \cdot H / (|H| + A)$$

where

- $M_{an}$ = the anhysteric magnetization
- $MS$ = the saturation magnetization
- $H$ = the magnetizing influence (after GAP correction)
- $A$ = a thermal energy parameter

For a given magnetizing influence ($H$), the anhysteric magnetization is the global flux level the material would attain if the domain walls could move freely. The walls, however, are stopped or pinned on dislocations in the material. The wall remains pinned until enough magnetic potential is available to break free, and travel to the next pinning site. The theory supposes a mean energy required, per volume, to move domain walls. This is analogous to mechanical drag. A simplified equation of this is

$$\text{change-in-magnetization} = \frac{\text{potential}}{\text{drag}}$$

The irreversible domain wall motion can, therefore, be expressed as

$$dM_{irr}/dH = (M_{an} - M)/K$$

where $K$ is the pinning energy per volume (drag).

Reversible wall motion comes from flexing in the domain walls, especially when it is pinned at a dislocation due to the magnetic potential (that is, the magnetization is not the anhysteric value).

The theory supposes spherical flexure to calculate energy values and arrives at the (simplified) equation:

$$dM_{rev}/dH = C \cdot d(M_{an} - M)/dH$$

where $C$ is the domain flexing parameter.

The equation for the total magnetization is the sum of these two state equations:

$$dM/dH = (1/(1 + C)) \cdot (M_{an} - M)/K + (C/(1 + C)) \cdot dM_{an}/dH$$

**Including air-gap effects in the inductor coupling model**

If the gap thickness is small compared with the other dimensions of the core, you can assume that all of the magnetic flux lines go through the gap directly and that there is little fringing flux...
(having a modest amount of fringing flux only increases the effective air-gap length). Checking the field values around the entire magnetic path gives the equation:

\[ H_{\text{core}} \cdot L_{\text{core}} + H_{\text{gap}} \cdot L_{\text{gap}} = n \cdot I \]

where \( n \cdot I \) is the sum of the amp-turns of the windings on the core. Also, the magnetization in the air-gap is negligible, so that \( B_{\text{gap}} = H_{\text{gap}} \) and \( B_{\text{gap}} = B_{\text{core}} \). These combine in the previous equation to yield:

\[ H_{\text{core}} \cdot L_{\text{core}} + B_{\text{core}} \cdot L_{\text{gap}} = n \cdot I \]

This is a difficult equation to solve, especially for the Jiles-Atherton model, which is a state equation model rather than an explicit function (which one would expect, because the B-H curve depends on the history of the material). However, there is a graphical technique that solves for \( B_{\text{core}} \) and \( H_{\text{core}} \), given \( n \cdot I \), which is to:

1. Take the non-gapped B-H curve.
2. Extend a line from the current value of \( n \cdot I \) having a slope of \(-L_{\text{core}}/L_{\text{gap}}\) (this would be vertical if \( L_{\text{gap}} = 0 \)).
3. Find the intersection of the line using the B-H curve.

The intersection is the value for \( B_{\text{core}} \) and \( H_{\text{core}} \) for the \( n \cdot I \) of the gapped core. The \( n \cdot I \) value is the apparent or external value of \( H_{\text{core}} \), but the real value of \( H_{\text{core}} \) is less. The result is a smaller value for \( B_{\text{core}} \) and for the sheared-over B-H curves of a gapped core. The simulator implements the numerical equivalent of this graphical technique.

The resulting B-H values are recorded in the Probe data file as \( B_{\text{core}} \) and \( H_{\text{apparent}} \).

**Getting core inductor coupling model values**

Characterizing core materials can be performed using Parts, and verified by using PSpice and Probe. The model uses MKS (metric) units, however the results for Probe are converted to Gauss and Oersted, which can be displayed using \( B(K_{xxx}) \) and \( H(K_{xxx}) \). The traditional B-H curve is made by a transient run, ramping current through a test inductor, then displaying \( B(K_{xxx}) \) and setting the X axis to \( H(K_{xxx}) \).

For more information on the Jiles-Atherton model, see Reference [1] of References on page 218.

**Inductor coupling: Spice Plus model**

Spice Plus models are treated as level 3 models in PSpice.

Winding in Spice Plus models
In PSpice windings are expressed using the L device.

The general syntax used is:

\[ K1 \ L1 \ L2 \ 1.0 \ N1 \]

The magnetic core model parameters based on Spice plus model are listed below:

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>OD</td>
<td>Outer diameter</td>
<td>cm</td>
<td>1</td>
</tr>
<tr>
<td>ID</td>
<td>Inner diameter</td>
<td>cm</td>
<td>0</td>
</tr>
<tr>
<td>AREA</td>
<td>Effective cross sectional area</td>
<td>cm²</td>
<td>1</td>
</tr>
<tr>
<td>GAP</td>
<td>Effective core air gap length</td>
<td>cm</td>
<td>NULL</td>
</tr>
<tr>
<td>BR</td>
<td>Residual flux density</td>
<td>Gauss</td>
<td>1000</td>
</tr>
<tr>
<td>BM</td>
<td>Saturated flux density</td>
<td>Gauss</td>
<td>2000</td>
</tr>
<tr>
<td>HC</td>
<td>Coercive magnetic force</td>
<td>Oersted</td>
<td>0.2</td>
</tr>
</tbody>
</table>

For Nonlinear Ferrite cores path length represented by LENGTH is used instead of the inner and outer diameters. The following figure illustrates the diameters and area specifications.

Geometry of Cores

The formula used to calculate magnetic path length is:

\[
\frac{(OD + ID) \pi}{2}
\]

Apart from the magnetic path length calculation, there is no difference between the toroidal and nonlinear ferrite core models.
**Defining Static Behavior**

In Spice Plus core models, following three parameters describe the DC hysteresis loop:

- $Br$ is the permanent flux density (where the loop intersects the y-axis).
- $Bm$ is the saturation flux density.
- $Hc$ is the coercive magnetic force at 0 (where the loop intersects the x-axis).

These parameters must be specified in CGS (centimeter-gram-second) units. The figure below illustrates the static B-H hysteresis loop parameters.

**Figure 2-1 Static B-H Loop Parameters**

![Static B-H Loop Diagram]

**Converting Units**

You can use the following table to convert units in data sheets to the unit used in the menu, and vice-versa. The table gives conversion factors for changing between MKS, CGS, and FPS units.

<table>
<thead>
<tr>
<th>Magnetic Quantity</th>
<th>Data Sheet Units</th>
<th>Menu Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>1 amp-turn per inch</td>
<td>0.495 oersteds</td>
</tr>
<tr>
<td>$H$</td>
<td>1 amp-turn per cm</td>
<td>1.257 oersteds</td>
</tr>
<tr>
<td>$B$</td>
<td>1 line per square inch</td>
<td>0.155 gauss</td>
</tr>
<tr>
<td>$B$</td>
<td>1 tesla</td>
<td>10,000 gauss</td>
</tr>
<tr>
<td>$B$</td>
<td>1 maxwell per square cm</td>
<td>1 gauss</td>
</tr>
<tr>
<td></td>
<td>Menu Units</td>
<td>Data Sheet Units</td>
</tr>
<tr>
<td>$H$</td>
<td>1 oersted</td>
<td>2.02 amp-turns per inch</td>
</tr>
<tr>
<td>$H$</td>
<td>1 oersted</td>
<td>0.796 amp-turns per cm</td>
</tr>
</tbody>
</table>
Core Model Theory

The ideal core has no current or voltage dependency; it is perfectly linear and never saturates. In a core constructed out of non-linear material, however, the effective inductance of a coil depends upon the current through the coil and on the history of the magnetizing currents applied to the coil. The permeability (change in flux density per change in magnetic field strength) varies as a function of magnetic field strength, and depends upon the previous application of magnetic fields.

To describe the behavior of a non-linear material, the flux density (B) is plotted as a function of field strength (H). In such a representation, the permeability (μ) is the slope of the B-H curve. Assuming an initially demagnetized core, the flux density rises steeply as field strength increases, until the saturation region of the material is approached. Physically, saturation of a non-linear material occurs as the majority of magnetic moments within the core become aligned with the magnetic field. Upon saturation, the flux density reaches a limiting value (Bm) which remains constant with further increases in field strength.

If the field strength is reduced following saturation of the core, the B-H curve follows a different path, returning to a positive finite value of magnetization as H falls to zero. The flux density remaining in the core following saturation defines the remanent point (Br) of the B-H curve. To reduce the value of B to zero once the core has been magnetized, a negative field strength must be applied to the material. The value of the field strength required to return the flux density to zero defines the coercive point (Hc) of the B-H loop.

The B-H characteristic of a magnetic material forms a symmetrical curve, so that if the magnetic field strength is increased in the negative direction, the flux density eventually reaches a limiting value of negative Bm. Like the positive characteristic, the negative B-H curve returns to a remaining point (minus Br) when H is reduced to zero, and requires a magnetic field of value Hc to return the magnetic flux density to zero. The total B-H characteristic of a material will form a hysteresis loop, as shown in Figure 2-2.
Figure 2-2  Hysteresis Loop Depicting B-H Characteristics of a Ferrite Core Material

**Limitations of Spice Plus Core Model**

The following limitations apply Spice Plus level3 core model:

1. The Spice Plus cores are static DC models. Frequency of operation during the simulation does not affect the B-H characteristic.

2. The Spice Plus core model defines saturation flux density (Bm) as an asymptote, or limiting value to the B-H curve.

   Core manufacturers typically define saturation as a point on the B-H curve above which the core’s loss of permeability begins to severely impact the intended application.

   If you copy Bm values directly from the tables in manufacturers’ databooks the Spice Plus core models will yield unexpectedly low Bm values. To account for the different definitions of saturation density, do not use the Bm values listed in the databook tables; instead read Bm values from the saturated regions of the accompanying B-H plots.

3. The air-gap model is inaccurate under DC and very low frequency operation (usually < 100 Hz).

   The inaccuracy is caused by a one milliohm resistor placed in series with the core inductance. To ensure accurate modeling of air gap, make sure that one of the following conditions is met:

   a. The winding resistance is greater than or equal to 100 milliohms.

   b. The inductive component of the impedance (Z_L) is greater than or equal to 100 milliohms. The magnitude of the inductive component of the impedance is given as:

      \[ |Z_L| = \omega \times L_{eq} \]

      where \( \omega \) is radian frequency, and \( L_{eq} \) is equivalent inductance.
Replacing $L_{eq}$ with an equivalent expression ($2\mu An$) yields:

$$|Z| = \frac{2\mu An}{L}$$

where $\mu$ is permeability, $A$ is area, $n$ is number of turns and $L$ is length.

4. Circuits containing cores with $B_m$ greater than $10^6 \times H_c$ sometimes have convergence problems.

If you encounter convergence problems when modeling a core with very high saturation flux and very low density, substitute an ideal core, which models infinite saturation flux and zero coercive force.

**Transmission line coupling**

If a $K$ device is used to couple two transmission lines, then two coupling parameters are required.

<table>
<thead>
<tr>
<th>Device</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_m$</td>
<td>capacitive coupling</td>
<td>farad/length$^1$</td>
<td>none</td>
</tr>
<tr>
<td>$L_m$</td>
<td>inductive coupling</td>
<td>henries/length$^*$</td>
<td>none</td>
</tr>
</tbody>
</table>

1. Length units must be consistent using the LEN parameter for the transmission lines being coupled.

These parameters can be thought of as the off-diagonal terms of a capacitive coupling matrix, $[C]$, and an inductive coupling matrix, $[L]$, respectively. $[C]$ and $[L]$ are both symmetric matrices, and for two coupled lines, the following relationships hold:

$$[C] = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \quad [L] = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}$$

$C_{m} = C_{12} = C_{21}$ and $L_{m} = L_{12} = L_{21}$

$C_{12}$ represents the charge induced on the first conductor when the second conductor has a potential of one volt. In general, for a system of $N$ coupled lines, $C_{ij}$ is the charge on the $i^{th}$ conductor when the $j^{th}$ conductor is set to one volt, and all other conductors are grounded. The diagonal of the matrix is determined with the understanding that the self-capacitance is really the capacitance between the conductor and ground, so that:

$$C_{ii} = C_{ig} + \sum |C_{ij}|$$
where \( C_{ig} \) is equal to the capacitance per unit length for the \( i^{th} \) transmission line, and is provided along with the T device that describes the \( i^{th} \) line. The simulator calculates \( C_{ii} \) from this.

The values of \( C_{ij} \) in the matrix are negative values. Note that the simulator assigns \(-|C_{ml}|\) to the appropriate \( C_{ij} \), so that the sign used when specifying \( C_{ml} \) is ignored.

\( L_{12} \) is defined in terms of the flux between the \( 1^{st} \) conductor and the ground plane, when the \( 2^{nd} \) conductor carries a current of one ampere. If there are more than two conductors, all other conductors are assumed to be open.

\( L_{11} \) is equal to the inductance per unit length for the \( 1^{st} \) line and is obtained directly from the appropriate T device.

**Example**

The following circuit fragment shows an example using two coupled lines:

\[
\begin{align*}
T1 & 1 0 2 0 R=.31 L=.38u G=6.3u C=70p LEN=1 \\
T2 & 3 0 4 0 R=.29 L=.33u G=6.0u C=65p LEN=1 \\
K12 & T1 T2 Lm=.04u Cm=6p
\end{align*}
\]

This fragment leads to the following \([C]\) and \([L]\):

\[
[C] = \begin{bmatrix}
76p & -6p \\
-6p & 71p
\end{bmatrix} \quad [L] = \begin{bmatrix}
0.38u & 0.04u \\
0.04u & 0.33u
\end{bmatrix}
\]

The model used to simulate this system is based on the approach described by Tripathi and Rettig in Reference [1] of References on page 218 and is extended for lossy lines by Roychowdhury and Pederson in Reference [2]. The approach involves computing the system propagation modes by extracting the eigenvalues and eigenvectors of the matrix product \([L][C]\).

Note: This model is not general for lossy lines.

**Lossy lines**

For the lossy line case, the matrix product to be decoupled is actually:

\[
[R+sL][G+sC]
\]

where:
s = the Laplace variable  
R = the resistance per unit length matrix  
G = the conductance per unit length matrix.

The modes obtained from $[L][C]$ represent a high frequency asymptote for this system. Simulation results should be good approximations for low-loss lines. However, as shown in reference [2], the approximation becomes exact for homogeneous, equally-spaced lossy lines, provided that coupling beyond immediately adjacent lines is negligible (i.e., the coupling matrices are tridiagonal and Toeplitz).

Note: Coupled ideal lines can be modeled by setting R and G to zero. The Z0/TD parameter set is not supported for coupled lines.

References

For a further description of the Jiles-Atherton model, refer to:


For more information on transmission line coupling, refer to:


Inductor

General form

\[
L\langle \text{name} \rangle \langle (+) \text{ node} \rangle \langle (-) \text{ node} \rangle [\text{model name}] \langle \text{value} \rangle \\
+ [\text{IC=}\langle \text{initial value} \rangle]
\]

Examples

LLOAD 15 0 20mH
L2 1 2 .2E-6
LCHOKE 3 42 LMOD .03
LSENSE 5 12 2UH IC=2mA

Model form

\[
.\text{MODEL} \langle \text{model name} \rangle \text{ IND} [\text{model parameters}]
\]

Arguments and options

\((+)\) and \((-)\) nodes

Define the polarity when the inductor has a positive voltage across it.

The first node listed (or pin one in Capture), is defined as positive. The voltage across the component is therefore defined as the first node voltage less the second node voltage.

Positive current flows from the \((+)\) node through the inductor to the \((-)\) node. Current flow from the first node through the component to the second node is considered positive.

\[\text{[model name]}\]

If \[\text{[model name]}\] is left out, then the effective value is \langle \text{value} \rangle.

If \[\text{[model name]}\] is specified, then the effective value is given by the model parameters; see Inductance value formula on page 222.

If the inductor is associated with a Core model, then the effective value is the number of turns on the core. Otherwise, the effective value is the inductance. See the Model Form statement for the K device in Coupling on page 202 for more information on the Core model.
Capture parts

For standard L parts, the effective value of the part is set directly by the VALUE property.

In general, inductors should have positive component values (VALUE property). In all cases, components must not be given a value of zero.

However, there are cases when negative component values are desired. This occurs most often in filter designs that analyze an RLC circuit equivalent to a real circuit. When transforming from the real to the RLC equivalent, it is possible to end up with negative component values.

PSpice A/D allows negative component values for bias point, DC sweep, AC, and noise analyses. A transient analysis may fail for a circuit with negative components. Negative inductors may create instabilities in time that the analysis cannot handle.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>L</td>
<td>inductor</td>
<td>VALUE</td>
<td>inductance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IC</td>
<td>initial current through the inductor</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>during bias point calculation</td>
</tr>
<tr>
<td>XFRM_LINEAR</td>
<td>transformer</td>
<td>L1_VALUE</td>
<td>winding inductances in Henries</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L2_VALUE</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>COUPLING</td>
<td>coefficient of mutual coupling (must</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>be between 0 and 1)</td>
</tr>
<tr>
<td>K_LINEAR</td>
<td>transformer</td>
<td>Ln</td>
<td>inductor reference designator</td>
</tr>
</tbody>
</table>
Breakout parts

For non-stock passive and semiconductor devices, Capture provides a set of breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters. Another approach is to use the model editor to derive an instance model and customize this. For example, you could add device and/or lot tolerances to model parameters.

Basic breakout part names consist of the intrinsic PSpice A/D device letter plus the suffix BREAK. By default, the model name is the same as the part name and references the appropriate device model with all parameters set at their default. For instance, the DBREAK part references the DBREAK model, which is derived from the intrinsic PSpice A/D D model (.MODEL DBREAK D).

For breakout part LBREAK, the effective value is computed from a formula that is a function of the specified VALUE property.

<table>
<thead>
<tr>
<th>Device type</th>
<th>Part name</th>
<th>Part library file</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>inductor</td>
<td>LBREAK</td>
<td>breakout.olb</td>
<td>VALUE</td>
<td>inductance</td>
</tr>
<tr>
<td></td>
<td>K</td>
<td></td>
<td>IC</td>
<td>initial current through the inductor during bias point calculation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>MODEL</td>
<td>IND model name</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model parameters¹</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L$</td>
<td>Inductance multiplier</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>$IL1$</td>
<td>Linear current coefficient</td>
<td>amp⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>$IL2$</td>
<td>Quadratic current coefficient</td>
<td>amp⁻²</td>
<td>0.0</td>
</tr>
<tr>
<td>$TC1$</td>
<td>Linear temperature coefficient</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>$TC2$</td>
<td>Quadratic temperature coefficient</td>
<td>°C⁻²</td>
<td>0.0</td>
</tr>
<tr>
<td>$T_{ABS}$</td>
<td>Absolute temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>$T_{MEASURED}$</td>
<td>Measured temperature</td>
<td>°C</td>
<td></td>
</tr>
</tbody>
</table>
Inductor equations

Inductance value formula

If \([\text{model name}]\) is specified, then the effective value is given by:

\[
<\text{value}> \cdot L \cdot (1 + IL1 \cdot I + IL2 \cdot I^2) \cdot (1 + TC1 \cdot (T - T_{\text{nom}}) + TC2 \cdot (T - T_{\text{nom}})^2)
\]

where <value> is normally positive (though it can be negative, but not zero). \(T_{\text{nom}}\) is the nominal temperature (set using TNOM option).

Inductor equation for noise

The inductor does not have a noise model.

Inductor as Winding

General form

\[L\langle\text{name}\rangle \langle+\text{node}\rangle \langle-\text{node}\rangle <\text{TURNS}> [\text{RESIS} = \text{val}] [\text{IC} = \text{val}]\]

Examples

\[L1 \ 2 \ 0 \ 103 \ \text{resis=40m}\]
\[L2 \ 6 \ 0 \ 5 \ \text{resis=40m}\]

Arguments and options

---

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>(T_{\text{REL}}_{\text{GLOBAL}})</td>
<td>Relative to current temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>(T_{\text{REL}}_{\text{LOCAL}})</td>
<td>Relative to AKO model temperature</td>
<td>°C</td>
<td></td>
</tr>
</tbody>
</table>

1. For information on \(T_{\text{MEASURED}}, T_{\text{ABS}}, T_{\text{REL}}_{\text{GLOBAL}}, \) and \(T_{\text{REL}}_{\text{LOCAL}}\), see _MODEL (model definition) on page 58.
Example of windings coupled to a core

A sample of Spice plus level-1 core model with multiple windings is shown below.

```plaintext
L1 2 0 103 resis=40m
L2 6 0 5 resis=40m
K1  L1  L2  1.0  N1
.model N1 core(Level=3 od=2.88 id=0.0 area=1.38 gap=0.04 br=2300 bm=4850 +hc=0.188)
```

To know more about .MODEL, see .MODEL (model definition) on page 58.
MOSFET

General form

\[ M<\text{name}> \text{<drain node> <gate node> <source node>} \]
\[ + \text{<bulk/substrate node> <model name>} \]
\[ + [L=<\text{value}>] [W=<\text{value}>] \]
\[ + [AD=<\text{value}>] [AS=<\text{value}>] \]
\[ + [PD=<\text{value}>] [PS=<\text{value}>] \]
\[ + [NRD=<\text{value}>] [NRS=<\text{value}>] \]
\[ + [NRG=<\text{value}>] [NRB=<\text{value}>] \]
\[ + [M=<\text{value}>] [N=<\text{value}>] \]

Examples

M1 14 2 13 0 PNMOS L=25u W=12u
M13 15 3 0 0 PSTRONG
M16 17 3 0 0 PSTRONG M=2
M28 0 2 100 100 NWEEAK L=33u W=12u
\[ + AD=288\text{p} AS=288\text{p} PD=60\text{u} PS=60\text{u} NRD=14 NRS=24 NRG=10 \]

Model form

\[ .\text{MODEL <model name>} \text{NMOS [model parameters]} \]
\[ .\text{MODEL <model name>} \text{PMOS [model parameters]} \]

Description

The MOSFET is modeled as an intrinsic MOSFET using ohmic resistances in series with the drain, source, gate, and bulk (substrate). There is also a shunt resistance (RDS) in parallel with the drain-source channel.
Arguments and options

$L$ and $W$

are the channel length and width, which are decreased to get the effective channel length and width. They can be specified in the device, .MODEL (model definition) on page 58, or .OPTIONS (analysis options) on page 71 statements. The value in the device statement supersedes the value in the model statement, which supersedes the value in the .OPTIONS statement. Defaults for $L$ and $W$ can be set in the .OPTIONS statement. If $L$ or $W$ defaults are not set, their default value is 100 u.

Note: $[L=<value>]$ $[W=<value>]$ cannot be used in conjunction with Monte Carlo analysis.

$AD$ and $AS$

The drain and source diffusion areas. Defaults for $AD$ and $AS$ can be set in the .OPTIONS statement. If $AD$ or $AS$ defaults are not set, their default value is 0.

$PD$ and $PS$

The drain and source diffusion perimeters. Their default value is 0.

$NRD$, $NRS$, $NRG$, and $NRB$

Multipliers (in units of squares) that can be multiplied by $RSH$ to yield the parasitic (ohmic) resistances of the drain ($RD$), source ($RS$), gate ($RG$), and substrate ($RB$), respectively. $NRD$, $NRS$, $NRG$, and $NRB$ default to 0.

Consider a square sheet of resistive material. Analysis shows that the resistance between two parallel edges of such a sheet depends upon its composition and thickness, but is independent of its size as long as it is square. In other words, the resistance will be the same whether the square’s edge is 2 mm, 2 cm, or 2 m. For this reason, the sheet resistance of such a layer, abbreviated $RSH$ on page 240, has units of ohms per square.
**M (NP)**

A parallel device multiplier (default = 1), which simulates the effect of multiple devices in parallel. (NP is an alias for M.)

The effective width, overlap and junction capacitances, and junction currents of the MOSFET are multiplied by M. The parasitic resistance values (e.g., RD and RS) are divided by M. Note the third example: it shows a device twice the size of the second example.

**N (NS)**

A series device multiplier (default value= 1.0) for the Level 5 model only, which simulates an approximation of the effect of multiple devices in series. NS is an aliased name for N.

There are some things to keep in mind while using this parameter. The parameter N is used to derive the effective length, \( L_{\text{eff}} = N \cdot (L+DL) \), of a transistor drawn as \( N \) elements of width \( W \) and length \( L \) in series (in other words, the drain of element \( [K] \) is the source of element \( [K+1] \), and the gates are tied together). The short-channel effects included in the pinch-off voltage calculation, however, are evaluated using the effective length \( L+DL \) of each element. Except for this, everything is calculated as if the transistor were laid out as a single element of length \( L=L_{\text{eff}}-DL=N \cdot (L+DL)-DL \).

In this compact formulation, the intermediate drain/source diffusions appearing along the channel are ignored (that is, junction capacitance and diffusion resistances are assumed to be zero). As a consequence, DC, AC and transient analyses can yield different results compared with the standard device declaration, particularly at higher frequencies. A closer match is obtained for long devices, or devices with low RS and RD and high UCRIT. Be sure to evaluate the accuracy of this compact formulation and to check the validity of the underlying approximations.

**JS**

Can specify the drain-bulk and source-bulk saturation currents. JS is multiplied by AD and AS.

**IS**

Can also specify the drain-bulk and source-bulk saturation currents. IS is an absolute value.
CJ

Can specify the zero-bias depletion capacitances. CJ is multiplied by AD and AS.

CJSW

Can also specify the zero-bias depletion capacitances. CJSW is multiplied by PD and PS.

CBD and CBS

Can also specify the zero-bias depletion capacitances. CBD and CBS are absolute values.

**Note:** Parameters IS, JS, CJ, CJSW, CBD, and CBS are model parameters. These parameters are specified in the *model parameters* section of the .MODEL statement.

**Comments**

The simulator provides seven MOSFET device models, which differ in the formulation of the I-V characteristic. The LEVEL parameter selects among different models as shown below. For more information, see References on page 270.

- **LEVEL=1** Shichman-Hodges model (see reference [1])
- **LEVEL=2** geometry-based, analytic model (see reference [2])
- **LEVEL=3** semi-empirical, short-channel model (see reference [2])
- **LEVEL=4** BSIM model (see reference [3])
- **LEVEL=5** EKV model version 2.6 (see reference [10])
- **LEVEL=6** BSIM3 model version 2.0 (see reference [7])
- **LEVEL=7** BSIM3 model version 3.2 (see reference [8])
- **LEVEL=8** BSIM4 model version 4.1.0 (see reference [11])
Capture parts

The following table lists the set of MOSFET breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MBREAKN</td>
<td>NMOS</td>
<td>L</td>
<td>channel length</td>
</tr>
<tr>
<td>MBREAKN3</td>
<td></td>
<td>W</td>
<td>channel width</td>
</tr>
<tr>
<td>MBREAKN4</td>
<td></td>
<td>AD</td>
<td>drain diffusion area</td>
</tr>
<tr>
<td>MBREAKP</td>
<td>PMOS</td>
<td>AS</td>
<td>source diffusion area</td>
</tr>
<tr>
<td>MBREAKP3</td>
<td></td>
<td>PD</td>
<td>drain diffusion perimeter</td>
</tr>
<tr>
<td>MBREAKP4</td>
<td></td>
<td>PS</td>
<td>source diffusion perimeter</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NRD</td>
<td>relative drain resistivity (in squares)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NRS</td>
<td>relative source resistivity (in squares)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NRG</td>
<td>relative gate resistivity (in squares)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NRB</td>
<td>relative substrate resistivity (in squares)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>M</td>
<td>device multiplier (simulating parallel devices)</td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
<td></td>
<td>NMOS or PMOS model name</td>
</tr>
</tbody>
</table>

Setting operating temperature

Operating temperature can be set to be different from the global circuit temperature by defining one of the model parameters: T_ABS, T_REL_GLOBAL, or T_REL_LOCAL. Additionally, model parameters can be assigned unique measurement temperatures using the T_MEASURED model parameter. For more information, see MOSFET model parameters on page 229.
MOSFET model parameters

For all model levels

The parameters common to all model levels are primarily parasitic element values such as series resistance, overlap and junction capacitance, and so on.

Model levels 1, 2, and 3

The DC characteristics of the first three model levels are defined by the parameters $V_{TO}$, $K_P$, $\Lambda$, $\Phi$, and $\Gamma$. These are computed by the simulator if process parameters (e.g., $TOX$, and $NSUB$) are given, but the user-specified values always override. $V_{TO}$ is positive (negative) for enhancement mode and negative (positive) for depletion mode of N-channel (P-channel) devices.

Note: The default value for $TOX$ is 0.1 $\mu$m for Levels 2 and 3, but is unspecified for Level 1, which discontinues the use of process parameters.

For MOSFETs the capacitance model has been changed to conserve charge, affecting only the Level 1, 2, and 3 models.

Effective length and width for device parameters are calculated with the formula:

$$P_i = P_0 + P_L/L_e + P_W/W_e$$

where:

$L_e$ = effective length $= L - (LD \cdot 2)$
$W_e$ = effective width $= W - (WD \cdot 2)$

See .MODEL (model definition) on page 58 for more information.

Model level 4

Note: Unlike the other models in PSpice, the BSIM model is designed for use with a process characterization system that provides all parameters. Therefore, there are no defaults specified for the parameters, and leaving one out can cause problems.

The LEVEL=4 (BSIM1) model parameters are all values obtained from process characterization, and can be generated automatically. Reference [4] of References on page 270 describes a means of generating a process file, which must then be converted into .MODEL (model definition) on page 58 statements for inclusion in the Model Library or circuit file. (The simulator does not read process files.)
The level 4 (BSIM) and level 6 (BSIM3 version 2) models have their own capacitance model, which conserves charge and remains unchanged. References [6] and [7] describe the equations for the capacitance due to channel charge.

In the following MOSFET model parameters on page 238 list, parameters marked with a $\zeta$ in the Default column also have corresponding parameters with a length and width dependency. For example, VFB is a basic parameter using units of volts, and LVFB and WVFB also exist and have units of volt$\cdot$\micro. The formula

$$P_i = P_0 + \frac{P_L}{L_e} + \frac{P_W}{W_e}$$

is used to evaluate the parameter for the actual device, where:

$L_e = \text{effective length} = L - DL$

$W_e = \text{effective width} = W - DW$

**Model level 5 (EKV version 2.6)**

The EKV model is a scalable and compact model built on fundamental physical properties of the device. Use this model to design low-voltage, low-current analog, and mixed analog-digital circuits that use sub-micron technologies. The charge-based static, quasi-static dynamic, and noise models are all derived from the normalized transconductance-to-current ratio, which is accurately described for all levels of current, including the moderate inversion region. A single I-V expression preserves the continuity of first- and higher-order derivatives with respect to any terminal voltage in all regions of device operation.

Version 2.6 models the following:

- geometrical and process related aspects of the device (oxide thickness, junction depth, effective channel length and width, and so on)
- effects of doping profile and substrate effects
- weak, moderate, and strong inversion behavior
- mobility effects due to vertical and lateral fields and carrier velocity saturation
- short-channel effects such as channel-length modulation, source and drain charge sharing, and the reverse short channel effect
- thermal and flicker noise modeling
- short-distance geometry and bias-dependent device matching for Monte Carlo analysis.

For more detailed model information, see reference [10] of References on page 270.
Additional notes

- The EKV noise model is used rather than the PSpice noise model. The NLEV parameter is not used with this model.
- The $DL$ and $DW$ parameters usually have a negative value.
- 0 (zero) and O (the letter O) are not interchangeable. For example, use $VTO$, not $VT0$ ($VTO$ is referenced to the bulk); use $E0$, not $EO$; use $Q0$, not $QO$.
- Use the $AVTO$, $AKP$, and $AGAMMA$ model parameters with a DEV tolerance to perform Monte Carlo and Sensitivity/Worst-Case analyses. Their default values cannot be changed. The device-to-device matching of MOSFETs depends on the gate area, $W \cdot L$. Using $AVTO$, $AKP$, and $AGAMMA$ with a DEV tolerance applies the matching scaling law for the model equations and derives the device matching statistics (DEV tolerance) from a single normalized parameter. (Without these parameters, you would need to use a dedicated .MODEL card with a DEV tolerance for $VTO$, $KP$ and $GAMMA$ for each value of the gate area used in your design.)

Do not apply the LOT specification, which is a measure of the ability of the process to control the absolute value of a model parameter, to $AVTO$, $AKP$, and $AGAMMA$, because this would be redundant with the LOT specification for $VTO$, $KP$, and $GAMMA$.

- Use the model parameter $HDIF$ with the device parallel multiplier, $M$, to set default values for $AD$, $AS$, $PD$, and $PS$. Use $HDIF$ only for the MOSEKV (Level 5) model.

When $HDIF$ is specified, the following equations are used.

\[
NRD = \frac{HDIF}{W}
\]
\[
NRS = \frac{HDIF}{W}
\]

For $M = 1$, the following equations are used.

\[
AD = (2 \cdot HDIF) \cdot W
\]
\[
AS = (2 \cdot HDIF) \cdot W
\]
\[
PD = 2 \cdot ((2 \cdot HDIF) + W)
\]
\[
PS = 2 \cdot (2 \cdot HDIF) + W
\]

For $M \geq 2$ and even:

\[
AD = HDIF \cdot W
\]
\[
AS = (HDIF + (2 \cdot HDIF) / M) \cdot W
\]
\[
PD = (2 \cdot HDIF) + W
\]
For $M \geq 2$ and odd:

\[
\begin{align*}
AD &= (HDIF + (HDIF/M)) \cdot W \\
AS &= (HDIF + (HDIF/M)) \cdot W \\
PD &= (2 \cdot HDIF + W + ((2 \cdot HDIF + W)/M) \\
PS &= (2 \cdot HDIF + W + ((2 \cdot HDIF + W)/M)
\end{align*}
\]

If RGSH is specified, the default value for NRG is set to $0.5 \cdot W/L$. 

The model parameters $TOX$, $NSUB$, $VFB$, $UO$, and $VMAX$ accommodate scaling behavior of the process and basic intrinsic model parameters, as well as statistical circuit simulation. These parameters are only used if $COX$, $GAMMA$, and/or $PHI$, $VTO$, $KP$, and $UCRIT$ are not specified, respectively. Furthermore, a simpler mobility reduction model due to vertical field is accessible through the mobility reduction coefficient, $THETA$. $THETA$ is only used if $E0$ is not specified.

**Model level 6 (BSIM3 version 2.0)**

Note: The Level 6 Advanced parameters should not be changed unless the detail structure of the device is known and has specific, meaningful values.

The BSIM3 model is a physical model using extensive built-in dependencies of important dimensional and processing parameters. It includes the major effects that are important to modeling deep-submicrometer MOSFETs, such as threshold voltage reduction, nonuniform doping, mobility reduction due to the vertical field, bulk charge effect, carrier velocity saturation, drain-induced barrier lowering (DIBL), channel length modulation (CLM), hot-carrier-induced output resistance reduction, subthreshold conduction, source/drain parasitic resistance, substrate current induced body effect (SCBE), and drain voltage reduction in LDD structure. For additional, detailed model information, see References on page 270.

**Additional notes**

- The BSIM3v3 noise model (NOIMOD and its parameters) is used rather than the PSpice noise model (NLEV).
- If any of the following BSIM3 version 2.0 model parameters are not explicitly specified, they are calculated using the following equations.

\[
\begin{align*}
VTH0 &= VFB + PHI + K_1/PHI \\
K1 &= GAMMA2 - 2 \cdot K2/((PHI - VBM))
\end{align*}
\]
Default values listed for the BSIM3 version 2.0 parameters \( U_A, U_B, U_C, U_A1, A_B1, \) and \( U_C1 \) are used for simplified mobility modeling.

Model level 7 (BSIM3 version 3.2)

The BSIM3 version 3.2 model was developed by the University of California, Berkeley, as a deep submicron MOSFET model for use in deep-submicron digital and analog circuit designs. The BSIM3 version 3.2 model is an extension of the BSIM3 model, with the following enhancements and improvements:

- a new intrinsic capacitance model (the Charge Thickness Model), considering the finite charge layer thickness determined by quantum effect, is introduced as capMod 3. It is very accurate in all operating regions.
- improved modeling of C-V characteristics at the weak-to-inversion transition
- addition of TOX dependence into the threshold voltage (VTH) model
- addition of flat-band voltage (VFB) as a new model parameter to accurately model MOSFET’s with different gate materials
- improved substrate current scalability with the channel length, controlled through parameter \( \alpha_{1} \)
the non-quasi-static (NQS) model is restructured to improve model accuracy and simulation efficiency

- temperature dependence is added to the diode junction capacitance model where both the unit area junction capacitance and built-in potential are now temperature dependent

- the DC junction diode model now supports a resistance-free diode model and a current limiting feature

- addition of the option of using C-V inversion charge equations of CAPMOD 0, 1, 2 or 3 to calculate the thermal noise when NOIMOD == 2 or 4

- the small negative capacitance of CGS and CGD in the accumulation-depletion regions is eliminated

- a separate set of length/width-dependence parameters is introduced in the C-V model for CV channel length and width to better fit the capacitance data

- parameter checking is added to avoid invalid values for certain parameters

The BSIM3 version 3.2 model has the same physical basis as the BSIM3 version 2.0 model and retains the extensive built-in dependencies of dimensional and processing parameters of BSIM3 version 2.0.

For additional, detailed model information, see Reference [8] of References on page 270.

Additional notes

Note 1

If any of the following BSIM3 version 3.2 model parameters are not explicitly specified, they are calculated using the following equations:

Note: In the model template add the parameter, VERSION = 3.2. By default, version 3.1 is used.

If $VTHO$ is not specified, then:

$$VTHO = VFB + \phi_s + K1 \sqrt{\phi}$$

where:

- $VFB=1.0$, if not specified

If $VTHO$ is specified, then:

$$VFB = VTHO - \phi_s - K1 \sqrt{\phi_s}$$
Note 2

If \( K1 \) AND \( K2 \) are not specified, they are calculated using the following equations:

\[
K1 = \text{GAMMA2} - 2K2\sqrt{\phi_s - \text{VBM}}
\]

\[
2 = \frac{(\text{GAMMA1} - \text{GAMMA2})(\sqrt{\phi_S - \text{VBX}} - \sqrt{\phi_s})}{2\sqrt{\phi_s} (\sqrt{\phi_S - \text{VBM}} - \sqrt{\phi_s}) + \text{VBM}}
\]

where:

\[
\phi_s = 2Vt \cdot \ln \left( \frac{\text{NCH}}{n_i} \right)
\]

\[
Vt = \frac{k \cdot T\text{nom}}{q}
\]

\[
\left( n_i = 1.45 \cdot 10^{10} \left( \frac{T\text{nom}}{300.15} \right)^{1.5} \right) \exp \left( 21.5565981 - \frac{E_{g0}}{2Vt\text{mo}} \right)
\]

where

\[
E_{g0}\text{=the energy bandgap at temperature } T\text{nom=1.16} - \frac{(7.02 \cdot 10^{-4} \cdot T\text{nom}^2)}{(T\text{nom} + 1108)}
\]

Note 3

If \( \text{NCH} \) is not given and \( \text{GAMMA1} \) is given, then:

\[
\text{NCH} = \frac{\text{GAMMA1}^2 \cdot (\text{Cox})^2}{2q \cdot \varepsilon_{si}}
\]

If neither \( \text{GAMMA1} \) nor \( \text{NCH} \) is given, then \( \text{NCH} \) has a default value of \( 1.7e23 \, 1/m^3 \) and \( \text{GAMMA1} \) is calculated from \( \text{NCH} \):

\[
\text{GAMMA1} = \sqrt{\frac{2q \cdot \varepsilon_{si} \cdot \text{NCH}}{\text{Cox}}}
\]

If \( \text{GAMMA2} \) is not given, then:

\[
\text{GAMMA2} = \sqrt{\frac{2q \cdot \varepsilon_{si} \cdot \text{NSUB}}{\text{Cox}}}
\]
Note 4

If \( VBX \) is not given, it is calculated by:

\[
VBX = \phi_s - \frac{q \cdot NCH \cdot XT^2}{2 \cdot \varepsilon_{si}}
\]

Note 5

If \( CGSO \) is not given and \( DLC > 0 \), then:

\[
CGSO = (DLC \cdot Cox) - CGSL
\]

If the previously calculated \( CGSO < 0 \), then:

\( CGSO = 0 \)

Else:

\( CGSO = 0.6 \cdot XJ \cdot Cox \)

Note 6

If \( CGDO \) is not given and \( DLC > 0 \), then:

\[
CGDO = (DLC \cdot Cox) - CGSL
\]

If the previously calculated \( CGDO < 0 \), then

\( CGDO = 0 \)

Else:

\( CGDO = 0.6 \cdot XJ \cdot Cox \)

Note 7

If \( CF \) is not given, it is calculated by:

\[
CF = \left( \frac{2 \varepsilon_{ox}}{\pi} \right) \ln \left( 1 + \frac{4 \times 10^{-7}}{TOX} \right)
\]
Note 8

In BSIM3 version 3.0 and 3.1, \textit{NQSMOD} was a model parameter. From BSIM3 version 3.2, \textit{NQSMOD} is an (element) instance parameter. In the absence of an instance \textit{NQSMOD} parameter, the model parameter \textit{NQSMOD}, if any, will be considered.

Note 9

If the following parameters are not specified, their corresponding parameters, if specified, will be used.

<table>
<thead>
<tr>
<th>If following parameter is not specified</th>
<th>Then its corresponding parameter is used</th>
</tr>
</thead>
<tbody>
<tr>
<td>\textit{LLC}</td>
<td>\textit{LL}</td>
</tr>
<tr>
<td>\textit{LWC}</td>
<td>\textit{LW}</td>
</tr>
<tr>
<td>\textit{LWLC}</td>
<td>\textit{LWL}</td>
</tr>
<tr>
<td>\textit{WLC}</td>
<td>\textit{WL}</td>
</tr>
<tr>
<td>\textit{WWC}</td>
<td>\textit{WW}</td>
</tr>
<tr>
<td>\textit{WWLC}</td>
<td>\textit{WWL}</td>
</tr>
</tbody>
</table>

Note 10

Error or warning messages are reported if invalid values are specified for the following parameters:

- If \textit{PSCBE2} \(\leq 0.0\), a warning message is reported
- If (\textit{MOIN} < 5.0) or (\textit{MOIN} > 25.0), a warning message is reported
- If (\textit{ACDE} < 0.4) or (\textit{ACDE} > 1.6), a warning message is reported
- If (\textit{NOFF} < 0.1) or (\textit{NOFF} > 4.0), a warning message is reported
- If (\textit{VOFFCV} < -0.5) or (\textit{VOFFCV} > 0.5), a warning message is reported
- If (\textit{IJTH} < 0.0), a fatal error message is reported
- If (\textit{TOXM} \(\leq 0.0\)), a fatal error message is reported
Model Level 8 (BSIM4 version 4.1.0)

BSIM4 is an extension of BSIM3 model and provides robust and predictive simulations with increased accuracies in modeling various functions, such as tunneling and thermal noise. As specified by University of California, Berkeley, BSIM4 has the following improvements and enhancements:

- An accurate gate direct tunneling model
- A better model for pocket-implanted devices in Vth, bulk charge effect model, and Rout
- An asymmetrical and bias-dependent source/drain resistance, either internal or external to the intrinsic MOSFET, at the user’s discretion
- An acceptance of either the electrical or physical gate oxide thickness as the model input in a physically accurate manner
- The quantum mechanical charge-layer-thickness model for both IV and CV
- A more accurate mobility model for predictive modeling
- A gate-induced drain leakage (GIDL) current model, available in BSIM for the first time
- Different diode IV and CV characteristics for source and drain junctions
- A junction diode breakdown with or without current limiting
- A dielectric constant of the gate dielectric as a model parameter

Note: Flicker and Thermal Noise models and High-Speed/RF models should not be used as model or instance parameters as they are not included in the current implementation of BSIM4. The High-Speed/RF model parameters are XRCRG1, XRCRG2, GBMIN, RBPB, RBPD, RBPS, RBDB, and RBSB. The Flicker and Thermal Noise model parameters are: NOIA, NOIB, NOIC, EM, EF, KF, NTNOI, TNOIA, and TNOIB.

For additional, detailed model information, see Reference [11] of “References” on page 270.

MOSFET model parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>all levels</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$AF$</td>
<td>flicker noise exponent</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>$CBD$</td>
<td>zero-bias bulk-drain $p-n$ capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>$CBS$</td>
<td>zero-bias bulk-source $p-n$ capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGBO</td>
<td>gate-bulk overlap capacitance/channel length</td>
<td>farad/meter</td>
<td>0</td>
</tr>
<tr>
<td>CGDO</td>
<td>gate-drain overlap capacitance/channel width</td>
<td>farad/meter</td>
<td>0</td>
</tr>
<tr>
<td>CGSO</td>
<td>gate-source overlap capacitance/channel width</td>
<td>farad/meter</td>
<td>0</td>
</tr>
<tr>
<td>CJ</td>
<td>bulk p-n zero-bias bottom capacitance/area</td>
<td>farad/meter$^2$</td>
<td>0</td>
</tr>
<tr>
<td>CJSW</td>
<td>bulk p-n zero-bias sidewall capacitance/length</td>
<td>farad/meter</td>
<td>0</td>
</tr>
<tr>
<td>FC</td>
<td>bulk p-n forward-bias capacitance coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>GDSNOI</td>
<td>channel shot noise coefficient (use with NLEV=3)</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>IS</td>
<td>bulk p-n saturation current</td>
<td>amp</td>
<td>1E-14</td>
</tr>
<tr>
<td>JS</td>
<td>bulk p-n saturation current/area</td>
<td>amp/meter$^2$</td>
<td>0</td>
</tr>
<tr>
<td>JSSW</td>
<td>bulk p-n saturation sidewall current/length</td>
<td>amp/meter</td>
<td>0</td>
</tr>
<tr>
<td>KF</td>
<td>flicker noise coefficient</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>L</td>
<td>channel length</td>
<td>meter</td>
<td>DEFL</td>
</tr>
<tr>
<td>LEVEL</td>
<td>model index</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>MJ</td>
<td>bulk p-n bottom grading coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>MJSW</td>
<td>bulk p-n sidewall grading coefficient</td>
<td></td>
<td>0.33</td>
</tr>
<tr>
<td>N</td>
<td>bulk p-n emission coefficient</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>NLEV</td>
<td>noise equation selector</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>PB</td>
<td>bulk p-n bottom potential</td>
<td>volt</td>
<td>0.8</td>
</tr>
<tr>
<td>PBSW</td>
<td>bulk p-n sidewall potential</td>
<td>volt</td>
<td>PB</td>
</tr>
<tr>
<td>RB</td>
<td>bulk ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>RD</td>
<td>drain ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
</tbody>
</table>
## MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDS</td>
<td>drain-source shunt resistance</td>
<td>ohm</td>
<td>infinite</td>
</tr>
<tr>
<td>RG</td>
<td>gate ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>RS</td>
<td>source ohmic resistance</td>
<td>ohm</td>
<td>0</td>
</tr>
<tr>
<td>RSH</td>
<td>drain, source diffusion sheet resistance</td>
<td>ohm/square</td>
<td>0</td>
</tr>
<tr>
<td>TT</td>
<td>bulk p-n transit time</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>T_ABS</td>
<td>absolute temperature</td>
<td>°C</td>
<td>0</td>
</tr>
<tr>
<td>T_MEASURED</td>
<td>measured temperature</td>
<td>°C</td>
<td>0</td>
</tr>
<tr>
<td>T_REL_GLOB</td>
<td>relative to current temperature</td>
<td>°C</td>
<td>0</td>
</tr>
<tr>
<td>T_REL_LOCA</td>
<td>relative to AKO model temperature</td>
<td>°C</td>
<td>0</td>
</tr>
<tr>
<td>W</td>
<td>channel width</td>
<td>meter</td>
<td>DEFW</td>
</tr>
</tbody>
</table>

### levels 1, 2, and 3

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELTA</td>
<td>width effect on threshold</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>ETA</td>
<td>static feedback (Level 3)</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>GAMMA</td>
<td>bulk threshold parameter</td>
<td>volt$^{1/2}$</td>
<td>see Model levels 1, 2, and 3 on page 229</td>
</tr>
<tr>
<td>KP</td>
<td>transconductance coefficient</td>
<td>amp/volt$^2$</td>
<td>2.0E-5</td>
</tr>
<tr>
<td>KAPPA</td>
<td>saturation field factor (Level 3)</td>
<td></td>
<td>0.2</td>
</tr>
<tr>
<td>LAMBDA</td>
<td>channel-length modulation (Levels 1 and 2)</td>
<td>volt$^{-1}$</td>
<td>0.0</td>
</tr>
<tr>
<td>LD</td>
<td>lateral diffusion (length)</td>
<td>meter</td>
<td>0.0</td>
</tr>
<tr>
<td>NEFF</td>
<td>channel charge coefficient (Level 2)</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>NFS</td>
<td>fast surface state density</td>
<td>1/cm$^2$</td>
<td>0.0</td>
</tr>
<tr>
<td>NSS</td>
<td>surface state density</td>
<td>1/cm$^2$</td>
<td>none</td>
</tr>
<tr>
<td>NSUB</td>
<td>substrate doping density</td>
<td>1/cm$^3$</td>
<td>none</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>PHI</td>
<td>surface potential</td>
<td>volt</td>
<td>0.6</td>
</tr>
<tr>
<td>THETA</td>
<td>mobility modulation (Level 3)</td>
<td>volt^-1</td>
<td>0.0</td>
</tr>
<tr>
<td>TOX</td>
<td>oxide thickness</td>
<td>meter</td>
<td>see Model levels 1, 2, and 3 on page 229</td>
</tr>
<tr>
<td>TPG</td>
<td>Gate material type:</td>
<td></td>
<td>+1</td>
</tr>
<tr>
<td></td>
<td>+1 = opposite of substrate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>-1 = same as substrate</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 = aluminum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCRIT</td>
<td>mobility degradation critical field (Level 2)</td>
<td>volt/cm</td>
<td>1.0E4</td>
</tr>
<tr>
<td>UEXP</td>
<td>mobility degradation exponent (Level 2)</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>UTRA</td>
<td>(not used)</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mobility degradation transverse field coefficient</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UO</td>
<td>surface mobility</td>
<td>cm^2/</td>
<td>600</td>
</tr>
<tr>
<td></td>
<td>(The second character is the letter O, not the numeral zero.)</td>
<td>volt·sec</td>
<td></td>
</tr>
<tr>
<td>VMAX</td>
<td>maximum drift velocity</td>
<td>meter/sec</td>
<td>0</td>
</tr>
<tr>
<td>VTO</td>
<td>zero-bias threshold voltage</td>
<td>volt</td>
<td>0</td>
</tr>
<tr>
<td>WD</td>
<td>lateral diffusion (width)</td>
<td>meter</td>
<td>0</td>
</tr>
<tr>
<td>XJ</td>
<td>metallurgical junction depth (Levels 2 and 3)</td>
<td>meter</td>
<td>0</td>
</tr>
<tr>
<td>XQC</td>
<td>fraction of channel charge attributed to drain</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>DL</td>
<td>Channel shortening</td>
<td>mu-m (1E-6*m)</td>
<td></td>
</tr>
<tr>
<td>DW</td>
<td>Channel narrowing</td>
<td>mu-m (1E-6*m)</td>
<td></td>
</tr>
<tr>
<td>ETA</td>
<td>Zero-bias drain-induced barrier lowering coefficient</td>
<td>ζ</td>
<td></td>
</tr>
</tbody>
</table>

**level 4**

October 2012 241 Product Version 16.6
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>K1</td>
<td>Body effect coefficient</td>
<td>volt$^{1/2}$</td>
<td>ζ</td>
</tr>
<tr>
<td>K2</td>
<td>Drain/source depletion charge sharing coefficient</td>
<td></td>
<td>ζ</td>
</tr>
<tr>
<td>MUS</td>
<td>Mobility at zero substrate bias and Vds=Vdd</td>
<td>cm$^2$/volt·sec</td>
<td>ζ</td>
</tr>
<tr>
<td>MUZ</td>
<td>Zero-bias mobility</td>
<td>cm$^2$/volt·sec</td>
<td>ζ</td>
</tr>
<tr>
<td>N0</td>
<td>Zero-bias subthreshold slope coefficient</td>
<td></td>
<td>ζ</td>
</tr>
<tr>
<td>NB</td>
<td>Sens. of subthreshold slope to substrate bias</td>
<td></td>
<td>ζ</td>
</tr>
<tr>
<td>ND</td>
<td>Sens. of subthreshold slope to drain bias</td>
<td></td>
<td>ζ</td>
</tr>
<tr>
<td>PHI</td>
<td>Surface inversion potential</td>
<td>volt</td>
<td>ζ</td>
</tr>
<tr>
<td>TEMP</td>
<td>Temperature at which parameters were measured</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>TOX</td>
<td>Gate-oxide thickness</td>
<td>mu-m (1E-6*m)</td>
<td></td>
</tr>
<tr>
<td>U0</td>
<td>Zero-bias transverse-field mobility degradation</td>
<td>volt$^{-1}$</td>
<td>ζ</td>
</tr>
<tr>
<td>U1</td>
<td>Zero-bias velocity saturation</td>
<td>μ/volt</td>
<td>ζ</td>
</tr>
<tr>
<td>VDD</td>
<td>Measurement bias range</td>
<td>volts</td>
<td></td>
</tr>
<tr>
<td>VFB</td>
<td>Flat-band voltage</td>
<td>volt</td>
<td>ζ</td>
</tr>
<tr>
<td>WDF</td>
<td>Drain, source junction default width</td>
<td>meter</td>
<td></td>
</tr>
<tr>
<td>X2E</td>
<td>Sens. of drain-induced barrier lowering effect to substrate bias</td>
<td>volt$^{-1}$</td>
<td>ζ</td>
</tr>
<tr>
<td>X2MS</td>
<td>Sens. of mobility to substrate bias @ Vds=0</td>
<td>cm$^2$/volt$^2$·sec</td>
<td>ζ</td>
</tr>
<tr>
<td>X2MZ</td>
<td>Sens. of mobility to substrate bias @ Vds=0</td>
<td>cm$^2$/volt$^2$·sec</td>
<td>ζ</td>
</tr>
<tr>
<td>X2U0</td>
<td>Sens. of transverse-field mobility degradation effect to substrate bias</td>
<td>volt$^2$</td>
<td>ζ</td>
</tr>
<tr>
<td>X2U1</td>
<td>Sens. of velocity saturation effect to substrate bias</td>
<td>μ/volt$^2$</td>
<td>ζ</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>X3E</td>
<td>Sens. of drain-induced barrier lowering effect to drain bias @ Vds = Vdd</td>
<td>volt⁻¹</td>
<td>ζ</td>
</tr>
<tr>
<td>X3MS</td>
<td>Sens. of mobility to drain bias @ Vds = Vdd</td>
<td>cm²/volt²·sec</td>
<td>ζ</td>
</tr>
<tr>
<td>X3U1</td>
<td>Sens. of velocity saturation effect on drain</td>
<td>μ/volt²·sec</td>
<td>ζ</td>
</tr>
<tr>
<td>XPART</td>
<td>Gate-oxide capacitance charge model flag.</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$\text{XPART}=0$ selects a 40/60 drain/source charge partition in saturation, while $\text{XPART}=1$ selects a 0/100 drain/source charge partition.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### level 5: process parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>COX</td>
<td>gate oxide capacitance per unit area</td>
<td>F/m²</td>
<td>0.7E-3</td>
<td></td>
</tr>
<tr>
<td>XJ</td>
<td>junction depth</td>
<td>m</td>
<td>0.1E-6</td>
<td></td>
</tr>
<tr>
<td>DW</td>
<td>channel width correction</td>
<td>m</td>
<td>0.0</td>
<td>see Model level 5 (EKV version 2.6) on page 230</td>
</tr>
<tr>
<td>DL</td>
<td>channel length correction</td>
<td>m</td>
<td>0.0</td>
<td>see Model level 5 (EKV version 2.6) on page 230</td>
</tr>
<tr>
<td>HDIF</td>
<td>length of heavily doped diffusion contact to gate</td>
<td>m</td>
<td>0.0</td>
<td>see Model level 5 (EKV version 2.6) on page 230</td>
</tr>
</tbody>
</table>
## MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level 5: basic intrinsic parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VTO</td>
<td>long-channel threshold voltage</td>
<td>V</td>
<td>0.5</td>
</tr>
<tr>
<td></td>
<td>see Model level 5 (EKV version 2.6) on page 230</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GAMMA</td>
<td>body effect parameter</td>
<td>$\sqrt{V}$</td>
<td>1.0</td>
</tr>
<tr>
<td>PHI</td>
<td>bulk Fermi potential (-2)</td>
<td>V</td>
<td>0.7</td>
</tr>
<tr>
<td>KP</td>
<td>transconductance parameter</td>
<td>A/V$^2$</td>
<td>50.0E-6</td>
</tr>
<tr>
<td>E0</td>
<td>mobility reduction coefficient</td>
<td>V/m</td>
<td>1.0E12</td>
</tr>
<tr>
<td></td>
<td>see Model level 5 (EKV version 2.6) on page 230</td>
<td></td>
<td></td>
</tr>
<tr>
<td>UCRIT</td>
<td>longitudinal critical field</td>
<td>V/m</td>
<td>2.0E6</td>
</tr>
<tr>
<td><strong>level 5: channel length modulation and charge sharing parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LAMBDA</td>
<td>depletion length coefficient (channel length modulation)</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>WETA</td>
<td>narrow-channel effect coefficient</td>
<td></td>
<td>0.25</td>
</tr>
<tr>
<td>LETA</td>
<td>short-channel effect coefficient</td>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td><strong>level 5: impact ionization related parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IBA</td>
<td>first impact ionization coefficient</td>
<td>1/m</td>
<td>0.0</td>
</tr>
<tr>
<td>IBB</td>
<td>second impact ionization coefficient</td>
<td>V/m</td>
<td>3.0E8</td>
</tr>
<tr>
<td>IBN</td>
<td>saturation voltage factor for impact ionization</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td><strong>level 5: intrinsic temperature parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCV</td>
<td>threshold voltage temperature coefficient</td>
<td>V/K</td>
<td>1.0E-3</td>
</tr>
<tr>
<td>BEX</td>
<td>mobility temperature exponent</td>
<td></td>
<td>-1.5</td>
</tr>
<tr>
<td>UCEX</td>
<td>longitudinal critical field temperature exponent</td>
<td></td>
<td>0.8</td>
</tr>
<tr>
<td>IBBT</td>
<td>temperature coefficient for IBB</td>
<td>1/K</td>
<td>9.0E-4</td>
</tr>
</tbody>
</table>
## MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level 5: matching parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AVTO</td>
<td>area related threshold voltage temperature coefficient</td>
<td>V·m</td>
<td>1.0E-6 see Model level 5 (EKV version 2.6) on page 230</td>
</tr>
<tr>
<td>AKP</td>
<td>area related gain mismatch parameter</td>
<td>m</td>
<td>1.0E-6 see Model level 5 (EKV version 2.6) on page 230</td>
</tr>
<tr>
<td>AGAMMA</td>
<td>area related body effect mismatch parameter</td>
<td>V·m</td>
<td>1.0E-6 see Model level 5 (EKV version 2.6) on page 230</td>
</tr>
<tr>
<td><strong>level 5: resistance parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RBC</td>
<td>bulk contact resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>RBSH</td>
<td>bulk layer sheet resistance</td>
<td>ohm/square</td>
<td>0.0</td>
</tr>
<tr>
<td>RDC</td>
<td>drain contact resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>RGC</td>
<td>gate contact resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>RGSH</td>
<td>gate layer sheet resistance</td>
<td>ohm/square</td>
<td>0.0</td>
</tr>
<tr>
<td>RSC</td>
<td>source contact resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
</tbody>
</table>
## MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level 5: temperature parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TR1</td>
<td>first-order temperature coefficient for drain,</td>
<td>°C(^{-1})</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>source series resistance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TR2</td>
<td>second-order temperature coefficient for drain,</td>
<td>°C(^{-2})</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>source series resistance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRB</td>
<td>temperature coefficient for bulk series</td>
<td>°C(^{-1})</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>resistance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TRG</td>
<td>temperature coefficient for gate series</td>
<td>°C(^{-1})</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>resistance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XTI</td>
<td>drain, source junction current temperature</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>exponent</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>level 5: optional parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NSUB</td>
<td>channel doping</td>
<td>meter</td>
<td>see</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
<tr>
<td>THETA</td>
<td>mobility reduction coefficient</td>
<td>volt(^{-1})</td>
<td>see</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
<tr>
<td>TOX</td>
<td>oxide thickness</td>
<td>meter</td>
<td>see</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
<tr>
<td>UO</td>
<td>low-field mobility</td>
<td>(\frac{cm^2}{volt \cdot sec})</td>
<td>see</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
<tr>
<td>VFB</td>
<td>flat-band voltage</td>
<td>volt</td>
<td>see</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
<tr>
<td>VMAX</td>
<td>saturation velocity</td>
<td>meter/sec</td>
<td>see</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
<tr>
<td><strong>level 5: setup parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SATLIM</td>
<td>ratio defining the saturation limit (i_t / i_r)</td>
<td></td>
<td>54.6</td>
</tr>
<tr>
<td></td>
<td>see Additional Notes</td>
<td></td>
<td>Notes</td>
</tr>
</tbody>
</table>

---

October 2012 246 Product Version 16.6
### MOSFET model parameters, *continued*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0</td>
<td>bulk charge effect coefficient NMOS</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>bulk charge effect coefficient PMOS</td>
<td></td>
<td>4.4</td>
</tr>
<tr>
<td>A1</td>
<td>first non-saturation coefficient NMOS</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>first non-saturation coefficient PMOS</td>
<td>1/V</td>
<td>0.23</td>
</tr>
<tr>
<td>A2</td>
<td>second non-saturation coefficient NMOS</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td></td>
<td>second non-saturation coefficient PMOS</td>
<td></td>
<td>0.08</td>
</tr>
<tr>
<td>AT</td>
<td>saturation velocity temperature coefficient</td>
<td>m/sec</td>
<td>3.3E4</td>
</tr>
<tr>
<td>BULKMOD</td>
<td>bulk charge model selector:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>NMOS</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>PMOS</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>CDSC</td>
<td>drain/source and channel coupling capacitance</td>
<td>F/m²</td>
<td>2.4E-4</td>
</tr>
<tr>
<td>CDSCB</td>
<td>body bias sensitivity of CDSC</td>
<td>F/Vm²</td>
<td>0.0</td>
</tr>
<tr>
<td>DL</td>
<td>channel length reduction on one side</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>DROUT</td>
<td>channel length dependent coefficient of the DIBL effect on Rout</td>
<td></td>
<td>0.56</td>
</tr>
<tr>
<td>DSUB</td>
<td>subthreshold DIBL coefficient exponent</td>
<td></td>
<td>DROUT</td>
</tr>
<tr>
<td>DVT0</td>
<td>first coefficient of short-channel effect on threshold voltage</td>
<td></td>
<td>2.2</td>
</tr>
<tr>
<td>DVT1</td>
<td>second coefficient of short-channel effect on threshold voltage</td>
<td></td>
<td>0.53</td>
</tr>
<tr>
<td>DVT2</td>
<td>body bias coefficient of short-channel effect on threshold voltage</td>
<td>1/V</td>
<td>-0.032</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DW</td>
<td>channel width reduction on one side</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>ETA0</td>
<td>DIBL coefficient in subthreshold region</td>
<td>1/V</td>
<td>0.08</td>
</tr>
<tr>
<td>ETAB</td>
<td>body bias coefficient for the subthreshold DIBL coefficient</td>
<td>1/V</td>
<td>-0.07</td>
</tr>
<tr>
<td>K1</td>
<td>first-order body effect coefficient</td>
<td>1/V</td>
<td>see Model level 6 (BSIM3 version 2.0) on page 232</td>
</tr>
<tr>
<td>K2</td>
<td>second-order body effect coefficient</td>
<td>1/V</td>
<td>see Model level 6 (BSIM3 version 2.0) on page 232</td>
</tr>
<tr>
<td>K3</td>
<td>narrow width effect coefficient</td>
<td>1/V</td>
<td>80.0</td>
</tr>
<tr>
<td>K3B</td>
<td>body effect coefficient of K3</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>KETA</td>
<td>body bias coefficient of the bulk charge effect</td>
<td>1/V</td>
<td>-0.047</td>
</tr>
<tr>
<td>KT1</td>
<td>temperature coefficient for threshold voltage</td>
<td>V</td>
<td>-0.11</td>
</tr>
<tr>
<td>KT1L</td>
<td>channel length sensitivity of temperature coefficient for threshold voltage</td>
<td>V-m</td>
<td>0.0</td>
</tr>
<tr>
<td>KT2</td>
<td>body bias coefficient of the threshold voltage temperature effect</td>
<td>V-m</td>
<td>0.022</td>
</tr>
<tr>
<td>NFACTOR</td>
<td>subthreshold swing coefficient</td>
<td>1/cm³</td>
<td>1.0</td>
</tr>
<tr>
<td>NGATE</td>
<td>poly gate doping concentration</td>
<td>1/cm³</td>
<td>1.74E-7</td>
</tr>
<tr>
<td>NLX</td>
<td>lateral nonuniform doping coefficient</td>
<td>m</td>
<td>1.74E-7</td>
</tr>
<tr>
<td>NPEAK</td>
<td>peak doping concentration near interface</td>
<td>1/cm³</td>
<td>1.7E17</td>
</tr>
<tr>
<td>NSUB</td>
<td>substrate doping concentration</td>
<td>1/cm³</td>
<td>6.0E16</td>
</tr>
<tr>
<td>PCLM</td>
<td>channel length modulation coefficient</td>
<td>1/cm³</td>
<td>1.3</td>
</tr>
<tr>
<td>PDIBL1</td>
<td>first output resistance DIBL effect coefficient</td>
<td>1/cm³</td>
<td>0.39</td>
</tr>
</tbody>
</table>
## MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDIBL2</td>
<td>second output resistance DIBL effect coefficient</td>
<td></td>
<td>0.0086</td>
</tr>
<tr>
<td>PSCBE1</td>
<td>first substrate current body effect coefficient</td>
<td>V/m</td>
<td>4.24E8</td>
</tr>
<tr>
<td>PSCBE2</td>
<td>second substrate current body effect coefficient</td>
<td>m/V</td>
<td>1.0E-5</td>
</tr>
<tr>
<td>PVAG</td>
<td>gate dependence of Early voltage</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>RDS0</td>
<td>contact resistance</td>
<td>ohms</td>
<td>0.0</td>
</tr>
<tr>
<td>RDSW</td>
<td>parasitic resistance per unit width</td>
<td>ohms/μm</td>
<td>0.0</td>
</tr>
<tr>
<td>SATMOD</td>
<td>saturation model selector:</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>For semi-empirical output:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>resistance model 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For physical output:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>resistance model 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBTHMOD</td>
<td>subthreshold model selector:</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>no subthreshold model 0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BSIM1 subthreshold model 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BSIM3 subthreshold model 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>BSIM3 subthreshold model using log current</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TNOM</td>
<td>temperature at which parameters are extracted.</td>
<td>deg. C</td>
<td>27</td>
</tr>
<tr>
<td>TOX</td>
<td>gate oxide thickness</td>
<td>m</td>
<td>1.5E-8</td>
</tr>
<tr>
<td>UA</td>
<td>first-order mobility degradation coefficient</td>
<td>m/V</td>
<td>2.25E-9</td>
</tr>
<tr>
<td>UA1</td>
<td>temperature coefficient for UA</td>
<td>m/V</td>
<td>4.31E-9</td>
</tr>
<tr>
<td>UB</td>
<td>second-order mobility degradation coefficient</td>
<td>(m/V)^2</td>
<td>5.87E-19</td>
</tr>
<tr>
<td>UB1</td>
<td>temperature coefficient for UB</td>
<td>(m/V)^2</td>
<td>-7.61E-18</td>
</tr>
<tr>
<td>UC</td>
<td>body effect mobility degradation coefficient</td>
<td>1/V</td>
<td>0.0465</td>
</tr>
<tr>
<td>UC1</td>
<td>temperature coefficient for UC</td>
<td>1/V</td>
<td>-0.056</td>
</tr>
<tr>
<td>UTE</td>
<td>mobility temperature exponent</td>
<td></td>
<td>-1.5</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOFF</td>
<td>offset voltage in subthreshold region</td>
<td>V</td>
<td>-0.11</td>
</tr>
<tr>
<td>VSAT</td>
<td>saturation velocity at Temp=TNOM</td>
<td>cm/sec</td>
<td>8.0E6</td>
</tr>
<tr>
<td>VTH0</td>
<td>threshold voltage at Vbs=0 for large channel length</td>
<td>V</td>
<td>see Additional notes on page 232</td>
</tr>
<tr>
<td>W0</td>
<td>narrow width effect parameter</td>
<td>m</td>
<td>2.5E-6</td>
</tr>
<tr>
<td>XJ</td>
<td>junction depth</td>
<td>m</td>
<td>1.5E-7</td>
</tr>
<tr>
<td>XPART</td>
<td>charge partitioning coefficient:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>no charge model &lt; 0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>40/60 partition = 0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>50/50 partition = 0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0/100 partition = 1.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIT</td>
<td>capacitance due to interface trapped charge</td>
<td>F/m²</td>
<td>0.0</td>
</tr>
<tr>
<td>EM</td>
<td>critical electrical field in channel</td>
<td>V/m</td>
<td>4.1E7</td>
</tr>
<tr>
<td>ETA</td>
<td>drain voltage reduction coefficient due to LDD</td>
<td></td>
<td>0.3</td>
</tr>
<tr>
<td>GAMMA1</td>
<td>body effect coefficient near the interface</td>
<td>( \sqrt{V} )</td>
<td>see Additional notes on page 232</td>
</tr>
<tr>
<td>GAMMA2</td>
<td>body effect coefficient in the bulk</td>
<td>( \sqrt{V} )</td>
<td>see Additional notes on page 232</td>
</tr>
<tr>
<td>LDD</td>
<td>total length of the LDD region</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>LITL</td>
<td>characteristic length related to current depth</td>
<td>m</td>
<td>see Additional notes on page 232</td>
</tr>
<tr>
<td>PHI</td>
<td>surface potential under strong inversion</td>
<td>V</td>
<td>see Additional notes on page 232</td>
</tr>
<tr>
<td>U0</td>
<td>mobility at Temp=TNOM:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NMOS</td>
<td></td>
<td>cm²/V-sec</td>
<td>670.0</td>
</tr>
<tr>
<td>PMOS</td>
<td></td>
<td>cm²/V-sec</td>
<td>250.0</td>
</tr>
<tr>
<td>VBM</td>
<td>maximum applied body bias</td>
<td>V</td>
<td>-5.0</td>
</tr>
<tr>
<td>VBX</td>
<td>vbs at which the depletion width equals XT</td>
<td>V</td>
<td>see Additional notes on page 232</td>
</tr>
</tbody>
</table>
**MOSFET model parameters, continued**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>VFB</td>
<td>flat-band voltage</td>
<td>V</td>
<td>see Additional notes on page 232</td>
</tr>
<tr>
<td>VGHIGH</td>
<td>voltage shift of the higher bound of the transition region</td>
<td>V</td>
<td>0.12</td>
</tr>
<tr>
<td>VGLOW</td>
<td>voltage shift of the lower bound of the transition region</td>
<td>V</td>
<td>-0.12</td>
</tr>
<tr>
<td>XT</td>
<td>doping depth</td>
<td>m</td>
<td>1.55E-7</td>
</tr>
</tbody>
</table>

**level 7: control parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAPMOD</td>
<td>flag for the short-channel capacitance model</td>
<td>none</td>
<td>2</td>
</tr>
<tr>
<td>MOBMOD</td>
<td>mobility model selector</td>
<td>none</td>
<td>1</td>
</tr>
<tr>
<td>NOIMOD</td>
<td>flag for noise model</td>
<td>none</td>
<td>1</td>
</tr>
<tr>
<td>NQSMOD</td>
<td>flag for NQS model</td>
<td>none</td>
<td>0</td>
</tr>
<tr>
<td>PARAMCHK</td>
<td>flag for model parameter checking</td>
<td>none</td>
<td>0</td>
</tr>
</tbody>
</table>

**level 7: AC and capacitance parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACDE</td>
<td>exponential coefficient for charge thickness in CAPMOD=3 model for accumulation and depletion regions</td>
<td>m/V</td>
<td>1.0</td>
</tr>
<tr>
<td>CF</td>
<td>fringing field capacitance</td>
<td>F/m</td>
<td>see Note 1 on page 234</td>
</tr>
<tr>
<td>CKAPPA</td>
<td>coefficient for lightly doped region overlap capacitance fringing field capacitance</td>
<td>F/m</td>
<td>0.6</td>
</tr>
<tr>
<td>CLC</td>
<td>constant term for the short-channel model</td>
<td>m</td>
<td>0.1E-6</td>
</tr>
<tr>
<td>CLE</td>
<td>exponential term for the short-channel model</td>
<td>none</td>
<td>0.6</td>
</tr>
<tr>
<td>CGBO</td>
<td>gate-bulk overlap capacitance per unit channel length</td>
<td>F/m</td>
<td>0.0</td>
</tr>
<tr>
<td>CGDL</td>
<td>light-doped drain-gate region overlap capacitance</td>
<td>F/m</td>
<td>0.0</td>
</tr>
<tr>
<td>CGDO</td>
<td>non-LDD region drain-gate overlap capacitance per channel length</td>
<td>F/m</td>
<td>see Note 6 on page 236</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CGSL</td>
<td>light-doped source-gate region overlap capacitance</td>
<td>F/m</td>
<td>0.0</td>
</tr>
<tr>
<td>CGSO</td>
<td>non-LDD region source-gate overlap capacitance per channel length</td>
<td>F/m</td>
<td>see Note 5 on page 236</td>
</tr>
<tr>
<td>CJ</td>
<td>bottom junction capacitance per unit area</td>
<td>F/m^2</td>
<td>5.0E-4</td>
</tr>
<tr>
<td>CJSW</td>
<td>source/drain side junction capacitance per unit periphery</td>
<td>F/m</td>
<td>5.0E-10</td>
</tr>
<tr>
<td>CJSWG</td>
<td>source/drain gate sidewall junction capacitance per unit width</td>
<td>F/m</td>
<td>CJSW</td>
</tr>
<tr>
<td>DLC</td>
<td>length offset fitting parameter from C-V</td>
<td>m</td>
<td>LINT</td>
</tr>
<tr>
<td>DWC</td>
<td>width offset fitting parameter from C-V</td>
<td>m</td>
<td>WINT</td>
</tr>
<tr>
<td>MJ</td>
<td>bottom junction capacitance grading coefficient</td>
<td>none</td>
<td>0.5</td>
</tr>
<tr>
<td>MJSW</td>
<td>source/drain side junction capacitance grading coefficient</td>
<td>none</td>
<td>0.33</td>
</tr>
<tr>
<td>MJSWG</td>
<td>source/drain gate sidewall junction capacitance grading coefficient</td>
<td>none</td>
<td>MJSW</td>
</tr>
<tr>
<td>MOIN</td>
<td>coefficient for the gate-bias dependent surface potential</td>
<td>V^{0.5}</td>
<td>15.0</td>
</tr>
<tr>
<td>NOFF</td>
<td>CV parameter in V gsteff, CV for weak to strong inversion region</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>PB</td>
<td>bottom built-in potential</td>
<td>V</td>
<td>1.0</td>
</tr>
<tr>
<td>PBSW</td>
<td>source/drain side junction built-in potential</td>
<td>V</td>
<td>1.0</td>
</tr>
<tr>
<td>PBSWUG</td>
<td>source/drain gate sidewall junction built-in potential</td>
<td>V</td>
<td>PBSW</td>
</tr>
<tr>
<td>VFBCV</td>
<td>flat-band voltage parameter (for CAPMOD = 0 only)</td>
<td>V</td>
<td>-1.0</td>
</tr>
<tr>
<td>VOFFCV</td>
<td>CV parameter in V gsteff, CV for weak to strong inversion region</td>
<td>volt</td>
<td>0.0</td>
</tr>
<tr>
<td>XPART</td>
<td>charge partitioning rate flag</td>
<td>none</td>
<td>0.0</td>
</tr>
</tbody>
</table>
## MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level 7: bin description parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BINUNIT</td>
<td>bin unit scale selector</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>LMAX</td>
<td>maximum channel length</td>
<td>m</td>
<td>1.0</td>
</tr>
<tr>
<td>LMIN</td>
<td>minimum channel length</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>WMAX</td>
<td>maximum channel width</td>
<td>m</td>
<td>1.0</td>
</tr>
<tr>
<td>WMIN</td>
<td>minimum channel width</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td><strong>level 7: DC parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A0</td>
<td>bulk charge effect coefficient for channel length</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>A1</td>
<td>first non-saturation effect parameter</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>A2</td>
<td>second non-saturation factor</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>AGS</td>
<td>gate-bias coefficient of Abulk</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>ALPHA0</td>
<td>first parameter of impact-ionization current</td>
<td>m/V</td>
<td>0.0</td>
</tr>
<tr>
<td>ALPHA1</td>
<td>Isub parameter for length scaling</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>B0</td>
<td>bulk charge effect coefficient for channel width</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>B1</td>
<td>bulk charge effect width offset</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>BETA0</td>
<td>second parameter of impact-ionization current</td>
<td>V</td>
<td>30.0</td>
</tr>
<tr>
<td>CDSC</td>
<td>drain/source to channel coupling capacitance</td>
<td>F/m²</td>
<td>2.4E-4</td>
</tr>
<tr>
<td>CDSCB</td>
<td>body-bias sensitivity of CDSC</td>
<td>F/Vm²</td>
<td>0.0</td>
</tr>
<tr>
<td>CDSCD</td>
<td>drain-bias sensitivity of CDSC</td>
<td>F/Vm²</td>
<td>0.0</td>
</tr>
<tr>
<td>CIT</td>
<td>interface trap capacitance</td>
<td>F/m²</td>
<td>0.0</td>
</tr>
<tr>
<td>DELTA</td>
<td>effective Vds parameter</td>
<td>V</td>
<td>0.01</td>
</tr>
<tr>
<td>DROUT</td>
<td>L-dependence coefficient of the DIBL correction parameter in Rout</td>
<td>none</td>
<td>0.56</td>
</tr>
<tr>
<td>DSUB</td>
<td>DIBL coefficient exponent in subthreshold region</td>
<td>none</td>
<td>DROUT</td>
</tr>
<tr>
<td>DVT0</td>
<td>first coefficient of short-channel effect on threshold voltage</td>
<td>none</td>
<td>2.2</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, *continued*

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DVT0W</td>
<td>first coefficient of narrow-width effect on threshold voltage for small-channel length</td>
<td>1/m</td>
<td>0.0</td>
</tr>
<tr>
<td>DVT1</td>
<td>second coefficient of short-channel effect on threshold voltage</td>
<td>none</td>
<td>0.53</td>
</tr>
<tr>
<td>DVT2</td>
<td>body-bias coefficient of short-channel effect on threshold voltage</td>
<td>1/V</td>
<td>-0.032</td>
</tr>
<tr>
<td>DVT1W</td>
<td>second coefficient of narrow-width effect on threshold voltage for small channel length</td>
<td>1/m</td>
<td>5.3E6</td>
</tr>
<tr>
<td>DVT2W</td>
<td>body-bias coefficient of narrow-width effect for small channel length</td>
<td>1/V</td>
<td>-0.032</td>
</tr>
<tr>
<td>DWB</td>
<td>coefficient of substrate body bias dependence of Weff</td>
<td>m/V(^{1/2})</td>
<td>0.0</td>
</tr>
<tr>
<td>DWG</td>
<td>coefficient of gate dependence of Weff</td>
<td>m/V</td>
<td>0.0</td>
</tr>
<tr>
<td>ETA0</td>
<td>DIBL coefficient in subthreshold region</td>
<td>none</td>
<td>0.08</td>
</tr>
<tr>
<td>ETAB</td>
<td>body-bias coefficient for the subthreshold DIBL effect</td>
<td>1/V</td>
<td>-0.07</td>
</tr>
<tr>
<td>IJTH</td>
<td>diode limiting current</td>
<td>amp</td>
<td>0.1</td>
</tr>
<tr>
<td>JS</td>
<td>source-drain junction saturation current per unit area</td>
<td>A/m(^2)</td>
<td>1.0E-4</td>
</tr>
<tr>
<td>JSW</td>
<td>sidewall saturation current per unit length</td>
<td>A/m</td>
<td>0.0</td>
</tr>
<tr>
<td>K1</td>
<td>first-order body effect coefficient</td>
<td>V(^{1/2})</td>
<td>0.5</td>
</tr>
<tr>
<td>K2</td>
<td>second-order body effect coefficient</td>
<td>none</td>
<td>0.0</td>
</tr>
<tr>
<td>K3</td>
<td>narrow width coefficient</td>
<td>none</td>
<td>80.0</td>
</tr>
<tr>
<td>K3B</td>
<td>body effect coefficient of K3</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>KETA</td>
<td>body-bias coefficient of bulk charge effect</td>
<td>1/V</td>
<td>-0.047</td>
</tr>
<tr>
<td>LINT</td>
<td>length offset fitting parameter from I-V without bias</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>NFACTOR</td>
<td>subthreshold swing factor</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>NGATE</td>
<td>poly gate doping concentration</td>
<td>cm(^{-3})</td>
<td>0.0</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter1</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLX</td>
<td>lateral non-uniform doping parameter</td>
<td>m</td>
<td>1.74E-7</td>
</tr>
<tr>
<td>PCLM</td>
<td>channel length modulation parameter</td>
<td>none</td>
<td>1.3</td>
</tr>
<tr>
<td>PDIBLC1</td>
<td>first output resistance DIBL effect correction parameter</td>
<td>none</td>
<td>0.39</td>
</tr>
<tr>
<td>PDIBLC2</td>
<td>second output resistance DIBL effect correction parameter</td>
<td>none</td>
<td>0.0086</td>
</tr>
<tr>
<td>PDIBLCB</td>
<td>body effect coefficient of DIBL correction parameter</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>PRWB</td>
<td>body effect coefficient of $R_{DSW}$</td>
<td>$1/V^{1/2}$</td>
<td>0.0</td>
</tr>
<tr>
<td>PRWG</td>
<td>gate-bias effect coefficient of $R_{DSW}$</td>
<td>1/V</td>
<td>0.0</td>
</tr>
<tr>
<td>PSCBE1</td>
<td>first substrate current body effect parameter</td>
<td>V/m</td>
<td>4.24E8</td>
</tr>
<tr>
<td>PSCBE2</td>
<td>second substrate current body effect parameter</td>
<td>V/m</td>
<td>1.0E-5</td>
</tr>
<tr>
<td>PVAG</td>
<td>gate dependence of Early voltage</td>
<td>none</td>
<td>0.0</td>
</tr>
<tr>
<td>RDSW</td>
<td>parasitic resistance per unit width</td>
<td>$\Omega \cdot \mu m^{WR}$</td>
<td>0.0</td>
</tr>
<tr>
<td>RSH</td>
<td>source-drain sheet resistance</td>
<td>$\Omega$/square</td>
<td>0.0</td>
</tr>
<tr>
<td>U0</td>
<td>mobility at Temp=$T_{NOM}$</td>
<td>NMOS</td>
<td>670.0 cm$^2/(V\cdot$sec)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>PMOS</td>
<td>250.0</td>
</tr>
<tr>
<td>UA</td>
<td>first-order mobility degradation coefficient</td>
<td>m/V</td>
<td>2.25E-9</td>
</tr>
<tr>
<td>UB</td>
<td>second-order mobility degradation coefficient</td>
<td>(m/V)$^2$</td>
<td>5.87E-19</td>
</tr>
<tr>
<td>UC</td>
<td>body effect of mobility degradation coefficient</td>
<td>m/V$^2$</td>
<td>-4.65E-11</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>when $MOBMOD=1$ or $2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>-0.046 when $MOBMOD=3$</td>
</tr>
<tr>
<td>VBM</td>
<td>maximum applied body-bias in threshold voltage calculation</td>
<td>V</td>
<td>-3.0</td>
</tr>
<tr>
<td>VFB</td>
<td>DC flatband voltage</td>
<td>volt</td>
<td></td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOFF</td>
<td>offset voltage in the subthreshold region at large W and L</td>
<td>V</td>
<td>-0.08</td>
</tr>
<tr>
<td>VSAT</td>
<td>saturation velocity at Temp=TNOM</td>
<td>m/sec</td>
<td>8.0E 4</td>
</tr>
<tr>
<td>VTH0</td>
<td>threshold voltage @Vbs=0 for large L</td>
<td>V</td>
<td>0.7 (NMOS) -0.7 (PMOS) see Model level 7 (BSIM3 version 3.2) on page 233</td>
</tr>
<tr>
<td>W0</td>
<td>narrow-width parameter</td>
<td>m</td>
<td>2.5E-6</td>
</tr>
<tr>
<td>WINT</td>
<td>width-offset fitting parameter from I-V without bias</td>
<td>m</td>
<td>0.0</td>
</tr>
<tr>
<td>WR</td>
<td>width-offset from Weff for Rds calculation</td>
<td>none</td>
<td>1.0</td>
</tr>
</tbody>
</table>

#### Level 7: flicker noise parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>frequency exponent</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>EF</td>
<td>flicker exponent</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>EM</td>
<td>saturation field</td>
<td>V/m</td>
<td>4.1E7</td>
</tr>
<tr>
<td>KF</td>
<td>flicker noise parameter</td>
<td>none</td>
<td>0.0</td>
</tr>
<tr>
<td>NOIA</td>
<td>noise parameter A</td>
<td>none</td>
<td>1.0E20 (NMOS) 9.9E18 (PMOS)</td>
</tr>
<tr>
<td>NOIB</td>
<td>noise parameter B</td>
<td>none</td>
<td>5.0E4 (NMOS) 2.4E3 (PMOS)</td>
</tr>
<tr>
<td>NOIC</td>
<td>noise parameter C</td>
<td>none</td>
<td>-1.4E-12 (NMOS) 1.4E-12 (PMOS)</td>
</tr>
</tbody>
</table>

#### level 7: NQS parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELM</td>
<td>Elmore constant of the channel</td>
<td>none</td>
<td>5.0</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>level 7: process parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GAMMA1</td>
<td>body effect coefficient near the surface</td>
<td>V(^{1/2})</td>
<td>see Note 3 on page 235</td>
</tr>
<tr>
<td>GAMMA2</td>
<td>body effect coefficient in the bulk</td>
<td>V(^{1/2})</td>
<td>see Note 3 on page 235</td>
</tr>
<tr>
<td>NCH</td>
<td>channel doping concentration</td>
<td>1/cm(^3)</td>
<td>1.7E17</td>
</tr>
<tr>
<td>NSUB</td>
<td>substrate doping concentration</td>
<td>1/cm(^3)</td>
<td>6.0E16</td>
</tr>
<tr>
<td>TOX</td>
<td>gate-oxide thickness</td>
<td>m</td>
<td>1.5E-8</td>
</tr>
<tr>
<td>TOXM</td>
<td>gate-oxide thickness at which parameters are extracted</td>
<td>m</td>
<td>TOX</td>
</tr>
<tr>
<td>VBX</td>
<td>Vbs at which the depletion region = XT</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>XJ</td>
<td>junction depth</td>
<td>m</td>
<td>1.5E-7</td>
</tr>
<tr>
<td>XT</td>
<td>doping depth</td>
<td>m</td>
<td>1.55E-7</td>
</tr>
<tr>
<td><strong>level 7: temperature parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AT</td>
<td>temperature coefficient for saturation velocity</td>
<td>m/sec</td>
<td>3.3E4</td>
</tr>
<tr>
<td>KT1</td>
<td>temperature coefficient for threshold voltage</td>
<td>V</td>
<td>-0.11</td>
</tr>
<tr>
<td>KT1L</td>
<td>channel length dependence of the temperature coefficient for threshold voltage</td>
<td>V*m</td>
<td>0.0</td>
</tr>
<tr>
<td>KT2</td>
<td>body-bias coefficient of threshold voltage temperature effect</td>
<td>none</td>
<td>0.022</td>
</tr>
<tr>
<td>NJ</td>
<td>emission coefficient of junction</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>PRT</td>
<td>temperature coefficient for RDSW</td>
<td>Ω·μm</td>
<td>0.0</td>
</tr>
<tr>
<td>TNOM</td>
<td>temperature at which parameters are extracted</td>
<td>°C</td>
<td>27.0</td>
</tr>
<tr>
<td>TCJ</td>
<td>temperature coefficient of CJ</td>
<td>1/K</td>
<td>0.0</td>
</tr>
<tr>
<td>TCJSW</td>
<td>temperature coefficient of CJSW</td>
<td>1/K</td>
<td>0.0</td>
</tr>
<tr>
<td>TCJSWG</td>
<td>temperature coefficient of CJSWG</td>
<td>1/K</td>
<td>0.0</td>
</tr>
<tr>
<td>TPB</td>
<td>temperature coefficient of PB</td>
<td>V/K</td>
<td>0.0</td>
</tr>
<tr>
<td>TPBSW</td>
<td>temperature coefficient of PBSW</td>
<td>V/K</td>
<td>0.0</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPBSWG</td>
<td>temperature coefficient of PBSWG</td>
<td>V/K</td>
<td>0.0</td>
</tr>
<tr>
<td>UA1</td>
<td>temperature coefficient for UA</td>
<td>m/V</td>
<td>4.31E-9</td>
</tr>
<tr>
<td>UB1</td>
<td>temperature coefficient for UB</td>
<td>(m/V)^2</td>
<td>-7.61E-18</td>
</tr>
<tr>
<td>UC1</td>
<td>temperature coefficient for UC</td>
<td>m/V^2</td>
<td>-5.6E-11 when MOBMOD=1 or 2, -0.056 when MOBMOD=3</td>
</tr>
<tr>
<td>UTE</td>
<td>mobility temperature exponent</td>
<td>none</td>
<td>-1.5</td>
</tr>
<tr>
<td>XTI</td>
<td>junction current temperature exponent</td>
<td>none</td>
<td>3.0</td>
</tr>
</tbody>
</table>

#### level 7: W and L parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>LL</td>
<td>coefficient of length dependence for length offset</td>
<td>m^{LLN}</td>
<td>0.0</td>
</tr>
<tr>
<td>LLC</td>
<td>coefficient for length dependence for CV channel length offset</td>
<td>LI</td>
<td>m^{Lln}</td>
</tr>
<tr>
<td>LLN</td>
<td>power of length dependence for length offset</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>LW</td>
<td>coefficient of width dependence for length offset</td>
<td>m^{LWN}</td>
<td>0.0</td>
</tr>
<tr>
<td>LWC</td>
<td>coefficient for width dependence for CV channel length offset</td>
<td>m^{Lwn}</td>
<td>Lw</td>
</tr>
<tr>
<td>LWL</td>
<td>coefficient of length and width cross term for length offset</td>
<td>m^{LWN+LLN}</td>
<td>0.0</td>
</tr>
<tr>
<td>LWLC</td>
<td>coefficient for length and width dependence for CV channel length offset</td>
<td>m^{Lln+Lwn}</td>
<td>Lwl</td>
</tr>
<tr>
<td>LWN</td>
<td>power of width dependence for length offset</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>WL</td>
<td>coefficient of length dependence for width offset</td>
<td>m^{WLN}</td>
<td>0.0</td>
</tr>
<tr>
<td>WLC</td>
<td>coefficient of length dependence for CV channel width offset</td>
<td>m^{Wln}</td>
<td>Wl</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>WLN</td>
<td>power of length dependence of width offset</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>WW</td>
<td>coefficient of width dependence for width offset</td>
<td>$m^W_{WWN}$</td>
<td>0.0</td>
</tr>
<tr>
<td>WWC</td>
<td>coefficient for width dependence for CV channel width offset</td>
<td>$m^W_{Wwn}$</td>
<td>Ww</td>
</tr>
<tr>
<td>WWL</td>
<td>coefficient of length and width cross term for width offset</td>
<td>$m^W_{WN+LN}$</td>
<td>0.0</td>
</tr>
<tr>
<td>WWLC</td>
<td>coefficient for length and width dependence for CV channel width offset</td>
<td>$m^W_{Wln+Wwn}$</td>
<td>Wwl</td>
</tr>
<tr>
<td>WWN</td>
<td>power of width dependence of width offset</td>
<td>none</td>
<td>1.0</td>
</tr>
</tbody>
</table>

#### Level 8: Gate-Induced Drain Leakage Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGIDL</td>
<td>Pre-exponential coefficient for GIDL</td>
<td>Mho</td>
<td>0.0</td>
</tr>
<tr>
<td>BGIDL</td>
<td>Exponential coefficient for GIDL</td>
<td>V/m</td>
<td>2.3e9</td>
</tr>
<tr>
<td>CGIDL</td>
<td>Parameter for body-bias effect on GIDL</td>
<td>V$^3$</td>
<td>0.5</td>
</tr>
<tr>
<td>DGIDL</td>
<td>Fitting parameter for band bending for GIDL</td>
<td>V</td>
<td>0.8</td>
</tr>
</tbody>
</table>

#### Level 8: Gate Dielectric Tunneling Current Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIGBACC</td>
<td>Parameter for $I_{gb}$ in accumulation</td>
<td>$(F_{s}/g)^{0.5}m^{-1}$</td>
<td>0.43</td>
</tr>
<tr>
<td>BIGBACC</td>
<td>Parameter for $I_{gb}$ in accumulation</td>
<td>$(F_{s}/g)^{0.5}m^{-1}V^{-1}$</td>
<td>0.054</td>
</tr>
<tr>
<td>CIGBACC</td>
<td>Parameter for $I_{gb}$ in accumulation</td>
<td>V$^{-1}$</td>
<td>0.075</td>
</tr>
<tr>
<td>NIGBACC</td>
<td>Parameter for $I_{gb}$ in accumulation</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>AIGBINV</td>
<td>Parameter for $I_{gb}$ in inversion</td>
<td>$(F_{s}/g)^{0.5}m^{-1}$</td>
<td>0.35 (Fs2/g)0.5m-1</td>
</tr>
<tr>
<td>BIGBINV</td>
<td>Parameter for $I_{gb}$ in inversion</td>
<td>$(F_{s}/g)^{0.5}m^{-1}V^{-1}$</td>
<td>0.03</td>
</tr>
<tr>
<td>CIGBINV</td>
<td>Parameter for $I_{gb}$ in inversion</td>
<td>V$^{-1}$</td>
<td>0.006</td>
</tr>
<tr>
<td>EIGBINV</td>
<td>Parameter for $I_{gb}$ in inversion</td>
<td>V</td>
<td>1.1</td>
</tr>
<tr>
<td>NIGBINV</td>
<td>Parameter for $I_{gb}$ in inversion</td>
<td>none</td>
<td>3.0</td>
</tr>
</tbody>
</table>
### MOSFET model parameters, continued

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIGC</td>
<td>Parameter for $I_{gcs}$ and $I_{gcd}$</td>
<td>(F$s^2$/g)$^{0.5}$m$^{-1}$</td>
<td>0.054 (NMOS) and 0.31 (PMOS)</td>
</tr>
</tbody>
</table>

#### Level 8: Charge and Capacitance Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CKAPPAS</td>
<td>Coefficient of bias-dependent overlap capacitance for the source side</td>
<td>V</td>
<td>0.6</td>
</tr>
<tr>
<td>CKAPPAD</td>
<td>Coefficient of bias-dependent overlap capacitance for the drain side</td>
<td>CKAPPAS</td>
<td></td>
</tr>
</tbody>
</table>

#### Level 8: Asymmetric Source/Drain Junction Diode Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>IJTHSREV</td>
<td>Limiting current in reverse bias region</td>
<td>A</td>
<td>IJTHSREV=0.1</td>
</tr>
<tr>
<td>IJTHDREV</td>
<td></td>
<td></td>
<td>IJTHDREV=IJTHSREV</td>
</tr>
<tr>
<td>IJTHSFWD</td>
<td>Limiting current in forward bias region</td>
<td>A</td>
<td>IJTHSFWD=0.1</td>
</tr>
<tr>
<td>IJTHDFWD</td>
<td></td>
<td></td>
<td>IJTHDFWD=IJTHSFWD</td>
</tr>
<tr>
<td>XJBVS</td>
<td>Fitting parameter for diode breakdown</td>
<td>none</td>
<td>XJBVS=1.0</td>
</tr>
<tr>
<td>XJBVD</td>
<td></td>
<td></td>
<td>XJBVD=XJBVS</td>
</tr>
<tr>
<td>BVS BVD</td>
<td>Breakdown voltage</td>
<td>V</td>
<td>BVS=10.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>BVD=BVS</td>
</tr>
<tr>
<td>JSS JSD</td>
<td>Bottom junction reverse saturation current density</td>
<td>A/m$^2$</td>
<td>JSS=1.0e-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>JSD=JSS</td>
</tr>
<tr>
<td>JSWS JSWD</td>
<td>Isolation-edge sidewall reverse saturation current density</td>
<td>A/m</td>
<td>JSWS=0.0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>JSWD=JSWS</td>
</tr>
<tr>
<td>JSWGGS</td>
<td>Gate-edge sidewall reverse saturation current density</td>
<td>A/m</td>
<td>JSWGGS=0.0</td>
</tr>
<tr>
<td>JSWGGD</td>
<td></td>
<td></td>
<td>JSWGGD=JSWGGS</td>
</tr>
<tr>
<td>CJS CJD</td>
<td>Bottom junction capacitance per unit area at zero bias</td>
<td>F/m$^2$</td>
<td>CJS=5.0e-4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CJD=CJS</td>
</tr>
</tbody>
</table>
### Level 8: Temperature Dependence Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Unit</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>NJS, NJD</td>
<td>Emission coefficients of junction for source and drain junctions, respectively</td>
<td>none</td>
<td>NJS=1.0; NJD=NJS</td>
</tr>
<tr>
<td>XTIS, XTID</td>
<td>Junction current temperature exponents for source and drain junctions, respectively</td>
<td>none</td>
<td>XTIS=3.0; XTID=XTIS</td>
</tr>
</tbody>
</table>

1. See [MODEL (model definition)](page 58) on page 58.
2. A $\zeta$ in the Default column indicates that the parameter may have corresponding parameters exhibiting length and width dependence. See Model level 4 on page 229.

† For information on T_MEASURED, T_ABS, T_REL_GLOBAL, and T_REL_LOCAL, see .MODEL (model definition) on page 58.
**MOSFET Equations**

These equations describe an N-channel MOSFET. For P-channel devices, reverse the signs of all voltages and currents.

In the following equations:

- $V_{bs}$ = intrinsic substrate-intrinsic source voltage
- $V_{bd}$ = intrinsic substrate-intrinsic drain voltage
- $V_{ds}$ = intrinsic drain-intrinsic source voltage
- $V_{dsat}$ = saturation voltage
- $V_{gs}$ = intrinsic gate-intrinsic source voltage
- $V_{gd}$ = intrinsic gate-intrinsic drain voltage
- $V_t$ = $k \cdot T/q$ (thermal voltage)
- $V_{th}$ = threshold voltage
- $C_{ox}$ = the gate oxide capacitance per unit area.
- $f$ = noise frequency
- $k$ = Boltzmann’s constant
- $q$ = electron charge
- $L_{eff}$ = effective channel length
- $W_{eff}$ = effective channel width
- $T$ = analysis temperature (°K)
- $T_{nom}$ = nominal temperature (set using TNOM option)

Other variables are from [MOSFET model parameters](#) on page 229.

Note: Positive current is current flowing into a terminal (for example, positive drain current flows from the drain through the channel to the source).

**MOSFET equations for DC current**

- **All levels**
Ig = gate current = 0
Ib = bulk current = Ibs+Ibd

where

Ibs = bulk-source leakage current = Iss·(e^{Vbs/(N·Vt)}-1)
Ibd = bulk-drain leakage current = Ids·(e^{Vbd/(N·Vt)}-1)

where

if:
    JS = 0, or AS = 0, or AD = 0
then:
    Iss = IS
    Ids = IS
else
    Iss = AS·JS + PS·JSSW
    Ids = AD·JS + PD·JSSW

Id = drain current = Idrain-Ibd
Is = source current = -Idrain-Ibs

Level 1: Idrain
Normal mode: Vds > 0

Case 1
for cutoff region: Vgs-Vto < 0
    then: Idrain = 0

Case 2
for linear region: Vds < Vgs-Vto
    then:
        Idrain = (W/L)·(KP/2)·(1+LAMBDA·Vds)·Vds·(2·(Vgs-Vto)-Vds)

Case 3
for saturation region: 0 ≤ Vgs-Vto ≤ Vds
    then: Idrain = (W/L)·(KP/2)·(1+LAMBDA·Vds)·(Vgs-Vto)^2
        where
        Vto = VTO+GAMMA·((PHI-Vbs)^{1/2}-PHI^{1/2})
Inverted mode: \( V_{ds} < 0 \)

Switch the source and drain in the normal mode equations above.

**Levels 2 and 3: I\text{drain}**


**MOSFET equations for capacitance**

Note: All capacitances are between terminals of the intrinsic MOSFET, in other words, to the inside of the ohmic drain and source resistances. For levels 1, 2, and 3, the capacitance model has been changed to conserve charge.

**Levels 1, 2, and 3**

\[
C_{bs} = \text{bulk-source capacitance} = \text{area cap.} + \text{sidewall cap.} + \text{transit time cap.}
\]

\[
C_{bd} = \text{bulk-drain capacitance} = \text{area cap.} + \text{sidewall cap.} + \text{transit time cap.}
\]

where:

if: \( CBS = 0 \) AND \( CBD = 0 \)

then

\[
\begin{align*}
C_{bs} &= A_S \cdot C_j \cdot C_{bsj} + P_S \cdot C_{JSW} \cdot C_{bss} + TT \cdot G_{bs} \\
C_{bd} &= A_D \cdot C_j \cdot C_{bdj} + P_D \cdot C_{JSW} \cdot C_{bds} + TT \cdot G_{ds}
\end{align*}
\]

else

\[
\begin{align*}
C_{bs} &= CBS \cdot C_{bsj} + P_S \cdot C_{JSW} \cdot C_{bss} + TT \cdot G_{bs} \\
C_{bd} &= CBD \cdot C_{bdj} + P_D \cdot C_{JSW} \cdot C_{bds} + TT \cdot G_{ds}
\end{align*}
\]

where:

\[
G_{bs} = \text{DC bulk-source conductance} = \frac{dI_{bs}}{dV_{bs}}
\]

\[
G_{bd} = \text{DC bulk-drain conductance} = \frac{dI_{bd}}{dV_{bd}}
\]

if: \( V_{bs} \leq FC \cdot PB \)

then

\[
\begin{align*}
C_{bsj} &= (1-V_{bs}/PB)^{MJ} \\
C_{bss} &= (1-V_{bs}/PBSW)^{MJSW}
\end{align*}
\]

if: \( V_{bs} > FC \cdot PB \)
then
\[
C_{bsj} = (1-F_C)^{-(1+M_J)} \cdot (1-F_C \cdot (1+M_J) + M_J \cdot V_{bs}/P_B)
\]
\[
C_{bss} = (1-F_C)^{-(1+M_{JSW})} \cdot (1-F_C \cdot (1+M_{JSW}) + M_{JSW} \cdot V_{bs}/P_{BSW})
\]

if: \( V_{bd} \leq F_C \cdot P_B \)

then
\[
C_{bdj} = (1-V_{bd}/P_B)^{-M_J}
\]
\[
C_{bds} = (1-V_{bd}/P_{BSW})^{-M_{JSW}}
\]

if: \( V_{bd} > F_C \cdot P_B \)

then
\[
C_{bdj} = (1-F_C)^{-(1+M_J)} \cdot (1-F_C \cdot (1+M_J) + M_J \cdot V_{bd}/P_B)
\]
\[
C_{bds} = (1-F_C)^{-(1+M_{JSW})} \cdot (1-F_C \cdot (1+M_{JSW}))
\]

C_{gs} = gate-source overlap capacitance = C_{GSO} \cdot W

C_{gd} = gate-drain overlap capacitance = C_{GDO} \cdot W

C_{gb} = gate-bulk overlap capacitance = C_{GBO} \cdot L

**Levels 4 and 6**


**MOSFET equations for temperature effects**

**Note:** The ohmic (parasitic) resistances have no temperature dependence.

**All Levels**

\[
I_S(T) = I_S \cdot e^{(E_g(T_{nom}) \cdot T/T_{nom} - E_g(T))/V_t}
\]

\[
J_S(T) = J_S \cdot e^{(E_g(T_{nom}) \cdot T/T_{nom} - E_g(T))/V_t}
\]

\[
J_{SSW}(T) = J_{SSW} \cdot e^{(E_g(T_{nom}) \cdot T/T_{nom} - E_g(T))/V_t}
\]

\[
P_B(T) = P_B \cdot T/T_{nom} - 3 \cdot V_t \cdot \ln(T/T_{nom}) - E_g(T_{nom}) \cdot T/T_{nom} + E_g(T)
\]

\[
P_{BSW}(T) = P_{BSW} \cdot T/T_{nom} - 3 \cdot V_t \cdot \ln(T/T_{nom}) - E_g(T_{nom}) \cdot T/T_{nom} + E_g(T)
\]

\[
PHI(T) = PHI \cdot T/T_{nom} - 3 \cdot V_t \cdot \ln(T/T_{nom}) - E_g(T_{nom}) \cdot T/T_{nom} + E_g(T)
\]

where

\[
E_g(T) = \text{silicon bandgap energy} = 1.16 - 0.000702 \cdot T^2/(T+1108)
\]

\[
CBD(T) = CBD \cdot (1+M_J) \cdot (.0004 \cdot (T-T_{nom})+(1-P_B(T)/PB))
\]
MOSFET equations for noise

Noise is calculated assuming a 1.0-hertz bandwidth, using the following spectral power densities (per unit bandwidth).

The model parameter NLEV is used to select the form of shot and flicker noise, and GDSNOI is the channel shot noise coefficient model parameter. When NLEV<3, the original SPICE2 shot noise equation is used in both the linear and saturation regions, but the use of this equation may produce inaccurate results in the linear region. When NLEV=3, a different equation is used that is valid in both linear and saturation regions.

Note: For level 7 BSIM3 version 3.1 devices, the noise model NOIMOD (and its parameters) is used rather than the PSpice noise model NLEV.

The model parameters AF and KF are used in the small-signal AC noise analysis to determine the equivalent MOSFET flicker noise.

For more information, see reference [5] of References on page 270.

MOSFET channel shot and flicker noise

\[ I_{\text{chan}}^2 = I_{\text{shot}}^2 + I_{\text{flick}}^2 \]

Intrinsic MOSFET flicker noise

for \( NLEV = 0 \)

\[ I_{\text{flick}}^2 = \frac{KF \cdot Idrain^{AF}}{COX \cdot Leff^2 \cdot f} \]

for \( NLEV = 1 \)
Intrinsic MOSFET shot noise

For $NLEV < 3$

$\text{I}_{\text{shot}}^2 = \frac{8 \cdot k \cdot T \cdot g_m}{3}$

For $NLEV = 3$

And

For $NOIMOD < 3$ (BSIM3 level 7)

$\text{I}_{\text{shot}}^2 = \frac{8 \cdot k \cdot T}{3} \times \beta \times \left(V_{gs} - V_{th}\right) \frac{1 + a + a^2}{1 + a} \times GDSNOI$

Where:

For linear region: $a = 1 - \left(\frac{V_{ds}}{V_{dsat}}\right)$

For saturation region: $a = 0$

Parasitic resistance thermal noise

$\text{I}_{\text{d}}^2 = 4 \cdot k \cdot T / \text{RD}$

$\text{I}_{\text{g}}^2 = 4 \cdot k \cdot T / \text{RG}$

$\text{I}_{\text{s}}^2 = 4 \cdot k \cdot T / \text{RS}$

$\text{I}_{\text{b}}^2 = 4 \cdot k \cdot T / \text{RB}$
References

For a more complete description of the MOSFET models, refer to:


For more information on References [2] and [4], contact:
Bipolar transistor

General form

\[ \text{Q<name> < collector node> <base node> <emitter node>} \]
+ [substrate node] <model name> [area value]

Examples

Q1 14 2 13 PNP NOM
Q13 15 3 0 1 NPNSTRONG 1.5
Q7 VC 5 12 [SUB] LATPNP

Model form

.MODEL <model name> NPN [model parameters]
.MODEL <model name> PNP [model parameters]
.MODEL <model name> LPNP [model parameters]

Arguments and options

[substrate node]

is optional, and if not specified, the default is the ground.

Because the simulator allows alphanumeric names for nodes, and because there is no easy way to distinguish these from the model names, the name (not a number) used for the substrate node needs to be enclosed with square brackets [ ]. Otherwise, nodes would be interpreted as model names. See the third example.

[area value]

is the relative device area and has a default value of 1.
The simulator supports the following two models for a bipolar transistor:

Level 1: Gummel-Poon model
Level 2: Mextram model

Mextram is an extended model that can describe various features of the modern down-scaled transistor, such as avalanche, collector epilayer current, and overlap capacitances. The Mextram model supported by this simulator is level 504. For more information about Mextram 504, you can visit [http://www.semiconductors.philips.com/Philips_Models/bipolar/mextram/](http://www.semiconductors.philips.com/Philips_Models/bipolar/mextram/).

**Note:** Simulations might take more time for circuit involving the Mextram model in comparison to Gummel-Poon due to the complex nature of the equations. The convergence issues might also be more.

Following is a list of effects that are better modelled by Mextram:

- Temperature
- Charge storage
- Substrate
- Parasitic PNP
- Low-level, non-ideal base currents
- Hard- and quasi-saturation
- Weak avalanche
- Hot carrier effects in the collector epilayer
- Explicit modelling of inactive regions
- Split base-collector depletion capacitance
- Current crowding and conductivity modulation for base resistance
- Distributed high frequency effects in the intrinsic base (high frequency current crowding and excess phase shift)
- High-injection
- Built-in electric field in base region
- Bias-dependent Early effect
Note: The self heating effect of Mextram model level 504 is not supported in release 10.5. As a result, the self heating effect equations and parameters are not implemented in this simulation.

**Description**

The bipolar transistor is modeled as an intrinsic transistor using ohmic resistances in series with the collector \((RC/\text{area})\), with the base (value varies with current, see Bipolar transistor equations on page 288), and with the emitter \((RE/\text{area})\).

**Model Level 1**

The equivalent circuit for model level 2 shows the intrinsic part of the transistor and the base, emitter, and the collector or epilayer resistance.

**Model Level 2**

The equivalent circuit for model level 2 shows the intrinsic part of the transistor and the base, emitter, and the collector or epilayer resistance.

Note: Positive current is current flowing into a terminal.
You can use two flags, EXMOD and EXPHI, to introduce additional elements to the schematic of a transistor in model level 2.
The small signal equivalent circuit is shown by the following figure.
The small signal model uses the following small-signal parameters:

Note: The conductances are derivatives with respect to three different biases, namely, base-emitter denoted by the subscript \(x\), internal base-collector denoted by the subscript \(y\), and base-collector denoted by the subscript \(z\).

The transconductance, \(g_m\), is given by the following equation:

\[
g_m = \frac{g_{Rcv,y}(g_x - g_{\mu,x} + g_z - g_{\mu,z}) - (g_{Rcv,x} + g_{Rcv,z})(g_y - g_{\mu,y})}{g_{Rcv,y} + g_{\mu,y} - g_y}
\]

The base conductance, \(g_\pi\), is given by the following equation:

\[
g_\pi = g_S\pi + g_{\pi,x} + g_{\mu,x} + g_{\pi,z} + g_{\mu,z} + (g_{\pi,y} + g_{\mu,y}) \left[ \frac{dy}{dx} + \frac{dy}{dz} \right]
\]

The current amplification, \(\beta\), is given by the following equation:

\[
\beta = \frac{g_m}{g_\pi}
\]

The output conductance, \(g_{out}\), is given by the following equation:

\[
g_{out} = \frac{(g_y - g_{\mu,y})g_{Rcv,z} - (g_z - g_{\mu,z})g_{Rcv,y}}{g_{Rcv,y} + g_{\mu,y} - g_y}
\]

The feedback transconductance, \(g_\mu\), is given by the following equation:

\[
g_\mu = g_{\pi,z} + g_{\mu,z} + (g_{\pi,y} + g_{\mu,y}) \cdot \frac{dy}{dz} + g_{\mu ex} + X g_{\mu ex}
\]

The base-emitter capacitance, \(C_{BE}\), is given by the following equation:

\[
C_{BE} = C_{BE,x} + C_{BE}^S + C_{BC,x} + (C_{BE,y} + C_{BC,y}) \cdot \frac{dy}{dx} + C_{BEO}
\]
The base-collector capacitance, $C_{BC}$, is given by the following equation:

$$C_{BC} = (C_{BE,y} + C_{BC,y}) \cdot \frac{dy}{dz} + C_{BC,z} + C_{BCex} + XC_{BCex} + C_{BCO}$$

In addition to the listed parameters, the cut-off frequency $f_T$ is another important design parameter. The cut-off frequency is a compound small-signal quantity and can be represented in terms of the total transit time, as given by the following equation:

$$f_T = \frac{1}{(2\pi \tau_T)}$$

The total transit time, $\tau_T$, is given by the following equation:

$$\tau_T = C_{BE} S \cdot (r_x + r_{bb}) + (C_{BE,x} + C_{BC,x}) \cdot r_x$$

$$(C_{BE,y} + C_{BC,y}) \cdot r_y + (C_{BE,z} + C_{BC,z}) \cdot r_z + C_{BCex} r_{ex}$$

$$XC_{BCex} Xr_{ex} + (C_{BEO} + C_{BCO})(Xr_{ex} - R_C)$$

For model parameters with alternate names, such as $VAF$ and $VA$ (the alternate name is shown by using parentheses), either name can be used.

For model types NPN and PNP, the isolation junction capacitance is connected between the intrinsic-collector and substrate nodes. This is the same as in SPICE2, or SPICE3, and works well for vertical IC transistor structures. For lateral IC transistor structures there is a third model, LPNP, where the isolation junction capacitance is connected between the intrinsic-base and substrate nodes.
Capture parts

The following table lists the set of bipolar transistor breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QBREAKL</td>
<td>LPNP</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td></td>
<td></td>
<td>MODEL</td>
<td>LNP model name</td>
</tr>
<tr>
<td>QBREAKN</td>
<td>NPN</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td>QBREAKN3</td>
<td>NPN</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td>QBREAKN4</td>
<td>NPN</td>
<td>MODEL</td>
<td>NPN model name</td>
</tr>
<tr>
<td>QBREAKP</td>
<td>PNP</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td>QBREAKP3</td>
<td>PNP</td>
<td>AREA</td>
<td>area scaling factor</td>
</tr>
<tr>
<td>QBREAKP4</td>
<td>PNP</td>
<td>MODEL</td>
<td>PNP model name</td>
</tr>
</tbody>
</table>

Setting operating temperature

Operating temperature can be set to be different from the global circuit temperature by defining one of the model parameters: T_ABS, T_REL_GLOBAL, or T_REL_LOCAL. Additionally, model parameters can be assigned unique measurement temperatures using the T_MEASURED model parameter. See Bipolar transistor model parameters on page 280 for more information.
Bipolar transistor model parameters

Model level 1

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>flicker noise exponent</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>BF</td>
<td>ideal maximum forward beta</td>
<td>100.0</td>
<td></td>
</tr>
<tr>
<td>BR</td>
<td>ideal maximum reverse beta</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>CJC</td>
<td>base-collector zero-bias p-n capacitance</td>
<td>farad</td>
<td>0.0</td>
</tr>
<tr>
<td>CJE</td>
<td>base-emitter zero-bias p-n capacitance</td>
<td>farad</td>
<td>0.0</td>
</tr>
<tr>
<td>CJS (CCS)</td>
<td>substrate zero-bias p-n capacitance</td>
<td>farad</td>
<td>0.0</td>
</tr>
<tr>
<td>CN</td>
<td>quasi-saturation temperature coefficient for hole mobility</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CN</td>
<td>NPN</td>
<td>2.42</td>
<td></td>
</tr>
<tr>
<td>CN</td>
<td>PNP</td>
<td>2.20</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>quasi-saturation temperature coefficient for scattering-limited hole carrier velocity</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>NPN</td>
<td>0.87</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>PNP</td>
<td>0.52</td>
<td></td>
</tr>
<tr>
<td>EG</td>
<td>bandgap voltage (barrier height)</td>
<td>eV</td>
<td>1.11</td>
</tr>
<tr>
<td>FC</td>
<td>forward-bias depletion capacitor coefficient</td>
<td></td>
<td>0.5</td>
</tr>
<tr>
<td>GAMMA</td>
<td>epitaxial region doping factor</td>
<td></td>
<td>1E-11</td>
</tr>
<tr>
<td>IKF (IK)</td>
<td>corner for forward-beta high-current roll-off</td>
<td>amp</td>
<td>infinite</td>
</tr>
<tr>
<td>IKR</td>
<td>corner for reverse-beta high-current roll-off</td>
<td>amp</td>
<td>infinite</td>
</tr>
<tr>
<td>IRB</td>
<td>current at which Rb falls halfway to</td>
<td>amp</td>
<td>infinite</td>
</tr>
<tr>
<td>IS</td>
<td>transport saturation current</td>
<td>amp</td>
<td>1E-16</td>
</tr>
<tr>
<td>ISC (C4) †</td>
<td>base-collector leakage saturation current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>ISE (C2) †</td>
<td>base-emitter leakage saturation current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>ISS</td>
<td>substrate p-n saturation current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>ITF</td>
<td>transit time dependency on Ic</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>KF</td>
<td>flicker noise coefficient</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>Model parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>MJC (MC)</td>
<td>base-collector p-n grading factor</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>MJE (ME)</td>
<td>base-emitter p-n grading factor</td>
<td>0.33</td>
<td></td>
</tr>
<tr>
<td>MJS (MS)</td>
<td>substrate p-n grading factor</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>NC</td>
<td>base-collector leakage emission coefficient</td>
<td>2.0</td>
<td></td>
</tr>
<tr>
<td>NE</td>
<td>base-emitter leakage emission coefficient</td>
<td>1.5</td>
<td></td>
</tr>
<tr>
<td>NF</td>
<td>forward current emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>NK</td>
<td>high-current roll-off coefficient</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>NR</td>
<td>reverse current emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>NS</td>
<td>substrate p-n emission coefficient</td>
<td>1.0</td>
<td></td>
</tr>
<tr>
<td>PTF</td>
<td>excess phase @ 1/(2π·TF)Hz</td>
<td>degree</td>
<td>0.0</td>
</tr>
<tr>
<td>QCO</td>
<td>epitaxial region charge factor</td>
<td>coulomb</td>
<td>0.0</td>
</tr>
<tr>
<td>QUASIMOD</td>
<td>quasi-saturation model flag for temperature dependence</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>RB</td>
<td>zero-bias (maximum) base resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>RBM</td>
<td>minimum base resistance ohm</td>
<td>RB</td>
<td></td>
</tr>
<tr>
<td>RC</td>
<td>collector ohmic resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>RCO ‡</td>
<td>epitaxial region resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>RE</td>
<td>emitter ohmic resistance</td>
<td>ohm</td>
<td>0.0</td>
</tr>
<tr>
<td>TF</td>
<td>ideal forward transit time</td>
<td>sec</td>
<td>0.0</td>
</tr>
<tr>
<td>TR</td>
<td>ideal reverse transit time</td>
<td>sec</td>
<td>0.0</td>
</tr>
<tr>
<td>TRB1</td>
<td>RB temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>TRB2</td>
<td>RB temperature coefficient (quadratic)</td>
<td>°C⁻²</td>
<td>0.0</td>
</tr>
<tr>
<td>TRC1</td>
<td>RC temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>TRC2</td>
<td>RC temperature coefficient (quadratic)</td>
<td>°C⁻²</td>
<td>0.0</td>
</tr>
</tbody>
</table>
### Model parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRE1 RE temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>TRE2 RE temperature coefficient (quadratic)</td>
<td>°C⁻²</td>
<td>0.0</td>
</tr>
<tr>
<td>TRM1 RBM temperature coefficient (linear)</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>TRM2 RBM temperature coefficient (quadratic)</td>
<td>°C⁻²</td>
<td>0.0</td>
</tr>
<tr>
<td>T_ABS absolute temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_MEASURED measured temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_GLOBAL relative to current temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_LOCAL relative to AKO model temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>VAF (VA) forward Early voltage</td>
<td>volt</td>
<td>infinite</td>
</tr>
<tr>
<td>VAR (VB) reverse Early voltage</td>
<td>volt</td>
<td>infinite</td>
</tr>
<tr>
<td>VG quasi-saturation extrapolated bandgap voltage at 0° K</td>
<td>V</td>
<td>1.206</td>
</tr>
<tr>
<td>VJC (PC) base-collector built-in potential</td>
<td>volt</td>
<td>0.75</td>
</tr>
<tr>
<td>VJE (PE) base-emitter built-in potential</td>
<td>volt</td>
<td>0.75</td>
</tr>
<tr>
<td>VJS (PS) substrate p-n built-in potential</td>
<td>volt</td>
<td>0.75</td>
</tr>
<tr>
<td>VO carrier mobility knee voltage</td>
<td>volt</td>
<td>10.0</td>
</tr>
<tr>
<td>VTF transit time dependency on Vbc</td>
<td>volt</td>
<td>infinite</td>
</tr>
<tr>
<td>XCJC fraction of CJC connected internally to Rb</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>XCJC2 fraction of CJC connected internally to Rb</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>XCJS fraction of CJS connected internally to Rc</td>
<td></td>
<td></td>
</tr>
<tr>
<td>XTB forward and reverse beta temperature coefficient</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>XTF transit time bias dependence coefficient</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>XTI (PT) IS temperature effect exponent</td>
<td></td>
<td>3.0</td>
</tr>
</tbody>
</table>

1. For information on T_MEASURED, T_ABS, T_REL_GLOBAL, and T_REL_LOCAL, see MODEL (model definition) on page 58.

† The parameters ISE (C2) and ISC (C4) can be set to be greater than one. In this case, they are interpreted as multipliers of IS instead of absolute currents: that is, if ISE is greater than one, then it is replaced by ISE·IS. Likewise for ISC.

‡ If the model parameter RCO is specified, then quasi-saturation effects are included.
Distribution of the CJC capacitance

The distribution of the CJC capacitance is specified by $XCJC$ and $XCJC2$. The model parameter $XCJC2$ is used like $XCJC$. The differences between the two parameters are as follows.

<table>
<thead>
<tr>
<th>Branch</th>
<th>$XCJC$</th>
<th>$XCJC2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>intrinsic base to intrinsic collector</td>
<td>$XCJC*XCJC$</td>
<td>$XCJC2*XCJC$</td>
</tr>
<tr>
<td>extrinsic base to <strong>intrinsic</strong> collector</td>
<td>$(1.0 - XCJC)*XCJC$</td>
<td>not applicable</td>
</tr>
<tr>
<td>extrinsic base to <strong>extrinsic</strong> collector</td>
<td>not applicable</td>
<td>$(1.0 - XCJC2)*XCJC$</td>
</tr>
</tbody>
</table>

When $XCJC2$ is specified in the range $0 < XCJC2 < 1.0$, $XCJC$ is ignored. Also, the extrinsic base to extrinsic collector capacitance ($Cbx2$) and the gain-bandwidth product ($Ft2$) are included in the operating point information (in the output listing generated during a Bias Point Detail analysis, `.OP (bias point)` on page 70). For backward compatibility, the parameter $XCJC$ and the associated calculation of $Cbx$ and $Ft$ remain unchanged. $Cbx$ and $Ft$ appears in the output listing only when $XCJC$ is specified.

The use of $XCJC2$ produces more accurate results because $Cbx2$ (the fraction of $CJC$ associated with the intrinsic collector node) now equals the ratio of the device’s emitter area-to-base area. This results in a better correlation between the measured data and the gain bandwidth product ($Ft2$) calculated by PSpice.

$XCJS$, which is valid in the range $0 \leq XCJS \leq 1.0$, specifies a portion of the $CJS$ capacitance to be between the external substrate and external collector nodes instead of between the external substrate and internal collector nodes. When $XCJS$ is 1, $CJS$ is applied totally between the external substrate and internal collector nodes. When $XCJS$ is 0, $CJS$ is applied totally between the external substrate and external collector codes.

Model level 2

<table>
<thead>
<tr>
<th>Model Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Level 2: general parameters</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXAVL</td>
<td>flag for the extended modelling of avalanche currents</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>EXMOD</td>
<td>flag for the extended modelling of the external regions</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>Model Parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default Value</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------</td>
<td>-------</td>
<td>---------------</td>
</tr>
<tr>
<td>EXPHI</td>
<td>flag for the extended modelling of distributed HF effects in transients</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>MULT</td>
<td>number of parallel transistors modelled together</td>
<td></td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Level 2: intrinsic and extrinsic charge and current split parameters**

<table>
<thead>
<tr>
<th>Model Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCJC</td>
<td>sidewall fraction collector-base depletion capacitance that is under the emitter</td>
<td>farad</td>
<td>32E-03</td>
</tr>
<tr>
<td>XCJE</td>
<td>sidewall fraction of the emitter-base depletion capacitance</td>
<td>farad</td>
<td>0.4</td>
</tr>
<tr>
<td>XEXT</td>
<td>fraction of external charges between B and C1</td>
<td>coulomb</td>
<td>0.63</td>
</tr>
<tr>
<td>XIBI</td>
<td>sidewall fraction of the ideal base current</td>
<td>amp</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Level 2: current parameters**

<table>
<thead>
<tr>
<th>Model Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF</td>
<td>current gain of ideal forward base current</td>
<td></td>
<td>215.0</td>
</tr>
<tr>
<td>BRI (BR)</td>
<td>current gain of ideal reverse base current</td>
<td></td>
<td>7.0</td>
</tr>
<tr>
<td>IBF</td>
<td>saturation current of the non-ideal forward base current</td>
<td></td>
<td>2.7E-15</td>
</tr>
<tr>
<td>IBR</td>
<td>saturation current of the non-ideal reverse base current</td>
<td>amp</td>
<td>1.0E-15</td>
</tr>
<tr>
<td>IK (IKF)</td>
<td>intrinsic transistor high-injection knee current</td>
<td>amp</td>
<td>0.1</td>
</tr>
<tr>
<td>IKS</td>
<td>parasitic PNP transistor high-knee current</td>
<td>amp</td>
<td>250.0E-6</td>
</tr>
<tr>
<td>IS</td>
<td>intrinsic transistor saturation current</td>
<td>amp</td>
<td>22.0E-18</td>
</tr>
<tr>
<td>ISS</td>
<td>parasitic PNP transistor saturation current</td>
<td>amp</td>
<td>48.0E-18</td>
</tr>
<tr>
<td>MLF</td>
<td>non-ideality factor of the non-ideal forward base current</td>
<td></td>
<td>2.0</td>
</tr>
<tr>
<td>SFH</td>
<td>Voltage describing the curvature of the avalanche current</td>
<td>volt</td>
<td>0.3</td>
</tr>
</tbody>
</table>
## Model Parameters Description | Units | Default Value
--- | --- | ---
VAVL | Voltage for the curvature of the avalanche current | volt | 3.0
VEF (VAF) | forward early voltage of the intrinsic transistor | volt | 44.0
VER (VAR) | reverse early voltage of the intrinsic transistor | volt | 2.5
VLR | non-ideal base current cross-over voltage | volt | 0.2
WAVL | effective width of epilayer for avalanche current | m | 1.1E-6

### Level 2: resistance parameters (variable and constant)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AXI</td>
<td>smoothing parameters for the epilayer model</td>
<td></td>
<td>0.3</td>
</tr>
<tr>
<td>IHC</td>
<td>epilayer critical current for hot-carriers</td>
<td>amp</td>
<td>4.0E-3</td>
</tr>
<tr>
<td>RBC</td>
<td>external base constant resistance</td>
<td>ohm</td>
<td>23.0</td>
</tr>
<tr>
<td>RBV</td>
<td>pinched base low current resistance (under the emitter)</td>
<td>ohm</td>
<td>18.0</td>
</tr>
<tr>
<td>RCC (RC)</td>
<td>external collector constant resistance</td>
<td>ohm</td>
<td>12</td>
</tr>
<tr>
<td>RCV</td>
<td>epilayer low current resistance</td>
<td>ohm</td>
<td>150.0</td>
</tr>
<tr>
<td>RE</td>
<td>external emitter constant resistance</td>
<td>ohm</td>
<td>5.0</td>
</tr>
<tr>
<td>SCRCV</td>
<td>epilayer space charge resistance</td>
<td>ohm</td>
<td>1250.0</td>
</tr>
</tbody>
</table>

### Level 2: depletion capacitance parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBCO</td>
<td>base-collector overlap capacitance</td>
<td>farad</td>
<td>0.0</td>
</tr>
<tr>
<td>CBEO</td>
<td>base-emitter overlap capacitance</td>
<td>farad</td>
<td>0.0</td>
</tr>
<tr>
<td>CJC</td>
<td>collector-base junction depletion capacitance at zero bias</td>
<td>farad</td>
<td>78.0E-15</td>
</tr>
<tr>
<td>CJE</td>
<td>emitter-base junction depletion capacitance at zero bias</td>
<td>farad</td>
<td>73.0E-15</td>
</tr>
<tr>
<td>CJS</td>
<td>collector-substrate junction depletion capacitance at zero bias</td>
<td>farad</td>
<td>315.0E-15</td>
</tr>
<tr>
<td>Model Parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default Value</td>
</tr>
<tr>
<td>------------------</td>
<td>------------------------------------------------------------------</td>
<td>-------</td>
<td>---------------</td>
</tr>
<tr>
<td>MC (MJC)</td>
<td>collector depletion charge current modulation factor</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>PC (VJC)</td>
<td>collector-base depletion capacitance grading coefficient</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>PE (VJE)</td>
<td>emitter-base depletion capacitance grading coefficient</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>PS (VJS)</td>
<td>collector-substrate depletion capacitance grading coefficient</td>
<td>0.34</td>
<td></td>
</tr>
<tr>
<td>VDC</td>
<td>built-in diffusion voltage collector-base</td>
<td>volt</td>
<td>0.68</td>
</tr>
<tr>
<td>VDE</td>
<td>built-in diffusion voltage emitter-base</td>
<td>volt</td>
<td>0.95</td>
</tr>
<tr>
<td>VDS</td>
<td>built-in diffusion voltage emitter-substrate</td>
<td>volt</td>
<td>0.62</td>
</tr>
<tr>
<td>XP (XC)</td>
<td>constant fraction of collector-base depletion capacitance</td>
<td>farad</td>
<td>0.35</td>
</tr>
</tbody>
</table>

**Level 2: transit time parameters (diffusion charges)**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAUB</td>
<td>base transmit time</td>
<td>sec</td>
<td>4.2E-12</td>
</tr>
<tr>
<td>TAUE</td>
<td>emitter charge transmit time</td>
<td>sec</td>
<td>2.0E-12</td>
</tr>
<tr>
<td>TEPI</td>
<td>collector epilayer transmit time</td>
<td>sec</td>
<td>41.0E-12</td>
</tr>
<tr>
<td>TAUR</td>
<td>reverse transmit time</td>
<td>sec</td>
<td>520.0E-12</td>
</tr>
<tr>
<td>MTAU</td>
<td>emitter charge non-ideality factor</td>
<td></td>
<td>1.0</td>
</tr>
</tbody>
</table>

**Level 2: noise parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>flicker noise exponent</td>
<td></td>
<td>2.0</td>
</tr>
<tr>
<td>KF</td>
<td>ideal base current flicker noise coefficient</td>
<td></td>
<td>2.0E-11</td>
</tr>
<tr>
<td>KFN</td>
<td>non-ideal base current flicker noise coefficient</td>
<td></td>
<td>2.0E-11</td>
</tr>
</tbody>
</table>

**Level 2: temperature parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB</td>
<td>temperature coefficient of RB (pinched base low current resistance)</td>
<td>$^\circ C^{-1}$</td>
<td>1.0</td>
</tr>
<tr>
<td>AC</td>
<td>temperature coefficient of RCC (external collector constant resistance)</td>
<td>$^\circ C^{-1}$</td>
<td>2.0</td>
</tr>
<tr>
<td>Model Parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default Value</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------------------------------------</td>
<td>-------</td>
<td>---------------</td>
</tr>
<tr>
<td>AE</td>
<td>temperature coefficient of RE (external emitter constant resistance)</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>AEPI</td>
<td>temperature coefficient of RCV (epilayer low current resistance)</td>
<td>°C⁻¹</td>
<td>2.5</td>
</tr>
<tr>
<td>AEX</td>
<td>temperature coefficient of RBC (external base constant resistance)</td>
<td>°C⁻¹</td>
<td>0.62</td>
</tr>
<tr>
<td>AQBO</td>
<td>zero bias base charge temperature coefficient</td>
<td>°C⁻¹</td>
<td>0.3</td>
</tr>
<tr>
<td>AS</td>
<td>temperature coefficient of the mobility related to the substrate currents</td>
<td>°C⁻¹</td>
<td>1.58</td>
</tr>
<tr>
<td>DVGBF</td>
<td>band-gap voltage difference for forward current gain</td>
<td>volt</td>
<td>0.05</td>
</tr>
<tr>
<td>DVGBR</td>
<td>band-gap voltage difference for reverse current gain</td>
<td>volt</td>
<td>0.045</td>
</tr>
<tr>
<td>DVGTE</td>
<td>band-gap voltage difference for emitter charge</td>
<td>volt</td>
<td>0.05</td>
</tr>
<tr>
<td>DTA</td>
<td>difference between device and ambient temperature</td>
<td>°C</td>
<td>0.0</td>
</tr>
<tr>
<td>TREF</td>
<td>reference temperature</td>
<td>°C</td>
<td>25.0</td>
</tr>
<tr>
<td></td>
<td>if a value is defined for .temp, it will override the value specified in the TREF parameter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VGB</td>
<td>base band-gap voltage</td>
<td>volt</td>
<td>1.17</td>
</tr>
<tr>
<td>VGC</td>
<td>collector band-gap voltage</td>
<td>volt</td>
<td>1.18</td>
</tr>
<tr>
<td>VGJ</td>
<td>base-emitter junction recombination band-gap voltage</td>
<td>volt</td>
<td>1.15</td>
</tr>
<tr>
<td>VGS</td>
<td>substrate band-gap voltage</td>
<td>volt</td>
<td>1.20</td>
</tr>
</tbody>
</table>

Level 2: SiGe parameters

<table>
<thead>
<tr>
<th>Model Parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEG</td>
<td>base band-gap difference</td>
<td></td>
<td>0.0</td>
</tr>
<tr>
<td>XREC</td>
<td>base recombination prefactor</td>
<td></td>
<td>0.0</td>
</tr>
</tbody>
</table>
Bipolar transistor equations

Model level 1

The equations in this section describe an NPN transistor. For the PNP and LPNP devices, reverse the signs of all voltages and currents.

The following variables are used:

\[
\begin{align*}
V_{be} &= \text{intrinsic base-intrinsic emitter voltage} \\
V_{bc} &= \text{intrinsic base-intrinsic collector voltage} \\
V_{bs} &= \text{intrinsic base-substrate voltage} \\
V_{bw} &= \text{intrinsic base-extrinsic collector voltage (quasi-saturation only)} \\
V_{bx} &= \text{extrinsic base-intrinsic collector voltage} \\
V_{ce} &= \text{intrinsic collector-intrinsic emitter voltage} \\
V_{js} &= (\text{NPN}) \text{ intrinsic collector-substrate voltage} \\
&= (\text{PNP}) \text{ intrinsic substrate-collector voltage} \\
&= (\text{LPNP}) \text{ intrinsic base-substrate voltage} \\
V_t &= k \cdot T/q \text{ (thermal voltage)} \\
k &= \text{Boltzmann's constant} \\
q &= \text{electron charge} \\
T &= \text{analysis temperature (°K)} \\
T_{nom} &= \text{nominal temperature (set using the TNOM option)}
\end{align*}
\]

Other variables are listed in Bipolar transistor model parameters on page 280.

Note: Positive current is current flowing into a terminal.

Bipolar transistor equations for DC current

\[
\begin{align*}
I_b &= \text{base current} = \textit{area} \cdot (I_{be1}/BF + I_{be2} + I_{bc1}/BR + I_{bc2}) \\
I_c &= \text{collector current} = \textit{area} \cdot (I_{be1}/Kqb - I_{bc1}/Kqb - I_{bc1}/BR - I_{bc2}) \\
I_{be1} &= \text{forward diffusion current} = I_S \cdot (e^{V_{be}/(NF \cdot V_t)} - 1) \\
I_{be2} &= \text{non-ideal base-emitter current} = I_{SE} \cdot (e^{V_{be}/(NE \cdot V_t)} - 1)
\end{align*}
\]
Bipolar transistor equations for DC current

\[ I_{bc1} = \text{reverse diffusion current} = I_S \cdot (e^{\frac{V_{bc}}{(N_R \cdot V_t)}} - 1) \]

\[ I_{bc2} = \text{non-ideal base-collector current} = I_S \cdot (e^{\frac{V_{bc}}{(N_C \cdot V_t)}} - 1) \]

\[ K_{qb} = \text{base charge factor} = K_{q1} \cdot \frac{1 + (1 + 4 \cdot K_{q2}) N_K}{2} \]

\[ K_{q1} = \frac{1}{1 - \frac{V_{bc}}{V_{AF}} - \frac{V_{be}}{V_{AR}}} \]

\[ K_{q2} = \frac{I_{be1}}{I_KF} + \frac{I_{bc1}}{I_KR} \]

\[ I_S = \text{substrate current} = area \cdot I_{SS} \cdot (e^{\frac{V_{js}}{(N_S \cdot V_t)}} - 1) \]

\[ R_b = \text{actual base parasitic resistance} \]

**Case 1**

for: \( IR_B = \text{infinite (default value)} \)

then: \( R_b = \frac{(R_{BM} + (R_B - R_{BM})/K_{qb})}{area} \)

**Case 2**

For: \( IR_B > 0 \)

then:
\[ R_b = \frac{(R_{BM} + 3 \cdot (R_B - R_{BM}) \cdot \tan(x) - x)}{x \cdot (\tan(x))^2} \cdot \text{area} \]

where:
\[ x = \frac{(1 + (144/\pi^2) \cdot \frac{1b}{(area \cdot IR_B)})^{1/2} - 1}{(24/\pi^2) \cdot (\frac{1b}{(area \cdot IR_B)})^{1/2}} \]

Bipolar transistor equations for capacitance

All capacitances, except \( C_{bx} \), are between terminals of the intrinsic transistor which is inside of the collector, base, and emitter parasitic resistances. \( C_{bx} \) is between the intrinsic collector and the extrinsic base.

**base-emitter capacitance**

\[ C_{be} = \text{base-emitter capacitance} = C_{tbe} + area \cdot C_{jbe} \]

\[ C_{tbe} = \text{transit time capacitance} = t_f \cdot G_{be} \]
Bipolar transistor equations for quasi-saturation effect

Quasi-saturation is an operating region where the internal base-collector metallurgical junction is forward biased, while the external base-collector terminal remains reverse biased.

This effect is modeled by extending the intrinsic Gummel-Poon model, adding a new internal node, a controlled current source, $I_{epi}$, and two controlled capacitances, represented by the equations:

$$\text{tf} = \text{effective } TF = TF \cdot (1 + XTF \cdot (I_{be1}/(I_{be1} + area \cdot TF))^2 \cdot e^{V_{bc}/(1.44 \cdot V_{TF})})$$

$$G_{be} = \text{DC base-emitter conductance} = (dI_{be})/(dV_{b})$$

$$I_{be} = I_{be1} + I_{be2}$$

$$C_{jbe} = C_{JE} \cdot (1 - V_{be}/V_{JE})^{MJE} \quad IF \quad V_{be} \leq FC \cdot V_{JE}$$

$$C_{jbe} = C_{JE} \cdot (1-FC) \cdot (1-FC \cdot (1+MJE) + MJE \cdot V_{be}/V_{JE}) \quad IF \quad V_{be} > FC \cdot V_{JE}$$

**base-collector capacitance**

$$C_{bc} = \text{base-collector capacitance} = C_{tc} + area \cdot X_{JC} \cdot C_{jbc}$$

$$C_{tc} = \text{transit time capacitance} = TR \cdot G_{bc}$$

$$G_{bc} = \text{DC base-collector conductance} = (dI_{bc})/(dV_{bc})$$

$$C_{jbc} = C_{JC} \cdot (1 - V_{bc}/V_{JC})^{MJC} \quad IF \quad V_{bc} < FC \cdot V_{JC}$$

$$C_{jbc} = C_{JC} \cdot (1-FC) \cdot (1-FC \cdot (1+MJC) + MJC \cdot V_{bc}/V_{JC}) \quad IF \quad V_{bc} > FC \cdot V_{JC}$$

**extrinsic-base to intrinsic-collector capacitance**

$$C_{bx} = \text{extrinsic-base to intrinsic-collector capacitance} = area \cdot (1-X_{JC}) \cdot C_{jbx}$$

$$C_{jbx} = C_{JC} \cdot (1 - V_{bx}/V_{JC})^{MJC} \quad IF \quad V_{bx} \leq FC \cdot V_{JC}$$

$$C_{jbx} = C_{JC} \cdot (1-FC) \cdot (1-FC \cdot (1+MJC) + MJC \cdot V_{bx}/V_{JC}) \quad IF \quad V_{bx} > FC \cdot V_{JC}$$

**substrate junction capacitance**

$$C_{js} = \text{substrate junction capacitance} = area \cdot C_{jjs}$$

$$C_{jjs} = C_{JS} \cdot (1-V_{js}/V_{JS})^{MS}(assumes FC = 0) \quad IF \quad V_{js} \leq 0$$

$$C_{jjs} = C_{JS} \cdot (1+MJS \cdot V_{js}/V_{JS}) \quad IF \quad V_{js} > 0$$
charges \( Q_o \) and \( Q_w \). These additions are only included if the model parameter \( RCO \) is specified. See reference [3] of Model level 2 on page 292 for the derivation of this extension.

\[
\text{i}_{\text{epi}} = \text{area} \cdot (V_O \cdot (K(V_{bc}) - K(V_{bn}) - \ln((1+K(V_{bc}))/(1+K(V_{bn})))) + V_{bc} - V_{bn})/RCO \cdot (|V_{bc} - V_{bn}| + V_O)
\]

\[
Q_o = \text{area} \cdot QCO \cdot (K(V_{bc}) - 1 - \Gamma/2)
\]

\[
Q_w = \text{area} \cdot QCO \cdot (K(V_{bn}) - 1 - \Gamma/2)
\]

where \( K(v) = (1 + \Gamma \cdot e^{(v/V_t)})^{1/2} \)

\[\text{Bipolar transistor equations for temperature effect}\]

\[
\text{IS}(T) = IS \cdot e^{(V_{bc}/V_t-1) \cdot (E_g(T)/N \cdot V_t)} \cdot (T/T_{nom})^{XT/B}
\]

where \( N = 1 \)

\[
\text{ISE}(T) = (ISE/(T/T_{nom})^{XT/B}) \cdot e^{(V_{bc}/V_t-1) \cdot (E_g(T)/N \cdot V_t)} \cdot (T/T_{nom})^{XT/NE}
\]

\[
\text{ISC}(T) = (ISC/(T/T_{nom})^{XT/B}) \cdot e^{(V_{bc}/V_t-1) \cdot (E_g(T)/N \cdot V_t)} \cdot (T/T_{nom})^{XT/NE}
\]

\[
\text{ISS}(T) = (ISS/(T/T_{nom})^{XT/B}) \cdot e^{(V_{bc}/V_t-1) \cdot (E_g(T)/N \cdot V_t)} \cdot (T/T_{nom})^{XT/NS}
\]

\[
\text{BF}(T) = BF \cdot (T/T_{nom})^{XT/B}
\]

\[
\text{BR}(T) = BR \cdot (T/T_{nom})^{XT/B}
\]

\[
\text{RE}(T) = RE \cdot (1 + TRE1 \cdot (T - T_{nom}) + TRE2 \cdot (T - T_{nom})^2)
\]

\[
\text{RB}(T) = RB \cdot (1 + TRB1 \cdot (T - T_{nom}) + TRB2 \cdot (T - T_{nom})^2)
\]

\[
\text{RBM}(T) = RBM \cdot (1 + TRM1 \cdot (T - T_{nom}) + TRM2 \cdot (T - T_{nom})^2)
\]

\[
\text{RC}(T) = RC \cdot (1 + TRC1 \cdot (T - T_{nom}) + TRC2 \cdot (T - T_{nom})^2)
\]

\[
\text{VJE}(T) = VJE \cdot T/T_{nom} - 3 \cdot V_t \cdot ln(T/T_{nom}) - Eg(T_{nom}) \cdot T/T_{nom} + Eg(T)
\]

\[
\text{VJC}(T) = VJC \cdot T/T_{nom} - 3 \cdot V_t \cdot ln(T/T_{nom}) - Eg(T_{nom}) \cdot T/T_{nom} + Eg(T)
\]

\[
\text{VJS}(T) = VJS \cdot T/T_{nom} - 3 \cdot V_t \cdot ln(T/T_{nom}) - Eg(T_{nom}) \cdot T/T_{nom} + Eg(T)
\]

where \( Eg(T) = \text{silicon bandgap energy} = 1.16 - .000702 \cdot T^2/(T+1108) \)

\[
\text{CJE}(T) = CJE \cdot (1 + MJJE \cdot (.0004 \cdot (T - T_{nom}) + (1 - VJE(T)/VJE)))
\]

\[
\text{CJC}(T) = CJC \cdot (1 + MJJC \cdot (.0004 \cdot (T - T_{nom}) + (1 - VJC(T)/VJC)))
\]

\[
\text{CJS}(T) = CJS \cdot (1 + MJJS \cdot (.0004 \cdot (T - T_{nom}) + (1 - VJS(T)/VJS)))
\]
Note: The development of the temperature dependencies for the quasi-saturation model parameters GAMMA, RCO, and VO are described in Model level 2 on page 292, (reference [3]). These temperature dependencies are only used when the model parameter QUASIMOD = 1.0.

\[
\begin{align*}
\text{GAMMA(T)} &= \text{GAMMA(T}_{\text{nom}}) \cdot \left(\frac{T}{T_{\text{nom}}}\right)^3 \cdot \exp(-qV_G/k \cdot (1/T - 1/T_{\text{nom}})) \\
\text{RCO(T)} &= \text{RCO(T}_{\text{nom}}) \cdot \left(\frac{T}{T_{\text{nom}}}\right)^{CN} \\
\text{VO(T)} &= \text{VO(T}_{\text{nom}}) \cdot \left(\frac{T}{T_{\text{nom}}}\right)^{CN - D}
\end{align*}
\]

**Bipolar transistor equations for noise**

Noise is calculated assuming a 1.0-hertz bandwidth, using the following spectral power densities (per unit bandwidth):

<table>
<thead>
<tr>
<th>Parasitic resistances thermal noise</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RC</td>
<td>$I_c^2 = 4 \cdot k \cdot T / (R C / \text{area})$</td>
</tr>
<tr>
<td>RB</td>
<td>$I_b^2 = 4 \cdot k \cdot T / R B$</td>
</tr>
<tr>
<td>RE</td>
<td>$I_e^2 = 4 \cdot k \cdot T / (R E / \text{area})$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Base and collector currents shot and flicker noise</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>IB</td>
<td>$I_b^2 = 2 \cdot q \cdot I_b + K F \cdot I_b^{\text{AF/FREQUENCY}}$</td>
</tr>
<tr>
<td>IC</td>
<td>$I_c^2 = 2 \cdot q \cdot I_c$</td>
</tr>
</tbody>
</table>

**Model level 2**

The equations in this section describe a NPN transistor and use the following variables:

- $I_{c1c2}$ = epilayer current
- $I_{b1b2}$ = pinched-base current
- $I_{b1}$ = ideal forward base current
- $I_{b2}$ = non-ideal forward base current
- $I_{sb1}$ = ideal side-wall base current
Main current

The Early effect current due to the variation in the width of the base is given by the following equations.

Forward current

\[ I_f = I_s \cdot e^{(V_{b2e1})/(V_t)} \]

Reverse current

\[ I_r = I_s \cdot e^{(V_{b2c2})/(V_t)} \]

Main current

\[ I_n = \frac{I_f - I_r}{Q_b} \]

The base currents are given by the following equations.

Ideal forward base current
Non-ideal forward base current

\[ Ib1 = \frac{Is}{Bf} \cdot (e^{(Vb2e1)/(Vt)} - 1) \]

Ideal reverse base current

\[ Ib2 = Ib \cdot (e^{(Vb2e1)/(Mel \cdot Vt)} - 1) \]

Non-ideal reverse base current

\[ Ib3 = \frac{I_{br} \cdot e^{(Vb1c1)/(Vt)} - 1}{e^{(Vb1c1)/(2 \cdot Vt)} + e^{(Vlr)/(2 \cdot Vt)}} \]

In addition to main and base current, this model has an avalanche current, given by the following equation.

\[ Iavl = Ic1c2 \times G \cdot (Vb1c1, Ic1c2) \]

where G is the generation factor.

The substrate current, Isub models the parasitic PNP main current in reverse bias.

\[ Isub = \frac{2 \cdot Iss \cdot (e^{(Vb1c1)/(Vt)} - 1)}{1 + \sqrt{1 + (Is \cdot e^{(Vb1c1)/(Vt)})/(Ik)}} \]

The base resistance is modeled as an extrinsic part, RBC, and a variable intrinsic part, RBV. The current through the base resistance is a function of the applied voltage and is given by the following equation.

\[ Ib1b2 = \frac{Qb}{3 \cdot RbV} \cdot [2 \cdot Vt \cdot (e^{(Vb1b2)/(Vt)} - 1) + Vb1b2] \]

**Depletion capacitance**

The depletion capacitance at the emitter-base junction is given by the following equation

\[ C_{je} = (C_{je}) \left( \frac{Vde}{V_{deT}} \right)^{PE} \]
The depletion capacitance at the collector-substrate junction is given by the following equation:

\[ C_{js,T} = C_{js} \left( \frac{V_{ds}}{V_{ds,T}} \right)^{PS} \]

The depletion capacitance at the collector-base junction capacitance is given by the following equation:

\[ C_{jc,T} = C_{jc} \left[ (1 - Xp) \left( \frac{V_{dc}}{V_{dc,T}} \right)^{PC} + Xp \right] \]

**Diffusion charges**

Equations for diffusion charges depend upon the current transit time. In low current, the base and the emitter contributions are modelled by the following equations.

**Base contribution**

\[ \frac{Q_{be}}{2} = Q_{1} T_{b} \cdot I_{f} \cdot \frac{2}{1 + \sqrt{1 + 4 \cdot (I_{f})/(I_{k})}} \]

**Emitter contribution**

\[ Q_{e} = T_{e} \cdot I_{s} \cdot e^{(V_{b2e1})/(M_{t} \cdot V_{t})} \]

The high current contributions are due to a finite voltage drop in the collector epilayer and base widening given by the following equations.

\[ Q_{bc} = q_{1} \cdot T_{b} \cdot I_{r} \cdot \frac{2}{1 + \sqrt{1 + 4 \cdot (I_{r})/(I_{k})}} \]

\[ Q_{epi} = T_{epi} \cdot \left( \frac{X_{i}}{W_{epi}} \right)^{2} \cdot I_{epi} \]

**Excess phase shift**

The excess phase shift is an optional effect in Mextram and is modelled only if EXPHI is 1. Both the collector and emitter contributes to the phase shift.

The phase shift is given by the following equations:

\[ Q_{be} = q \cdot A_{em} \int_{0}^{W_{b}} X_{n}(1 - x/(W_{b}))dx \]
Where,

\[ n(X) = n(0) \cdot \left( \sinh[\lambda(1 - x/(Wb))] / (\sinh \lambda) \right) \]

and

\[ \lambda = (j\omega Wb^2) / (Dn) \]

The current to the emitter and the collector are given by:

\[
I(0) = Idc + j\omega \frac{2}{3} Q_{tot} = \left\{ Idc + \frac{d\left(\frac{2}{3} Q_{tot}\right)}{dt} \right\}
\]

\[
I(Wb) = Idc - j\omega \frac{1}{3} Q_{tot} = \left\{ Idc - \frac{d\left(\frac{1}{3} Q_{tot}\right)}{dt} \right\}
\]

The AC current crowding or the extra effect in the lateral direction is modelled by the following equation:

\[ Q_{bb1b2} = \frac{1}{5} \cdot V_{b1b2} \cdot (Cte + Cbe + Ce) \]

**Noise model equations**

The two types of noise, thermal noise due to parasitic resistance and flicker noise due to base and collector currents, are modelled by the following equations.

**Parasitic resistances thermal noise**

\[
Ic = 4 \cdot k \cdot T / ((Rc)/(MULT))
\]

\[
Ib = 4 \cdot k \cdot T / (Rb)
\]

\[
Ie = 4 \cdot k \cdot T / ((Re)/(MULT))
\]

**Base and collector currents shot and flicker noise**

\[
Ib = 2 \cdot q \cdot Ib + (Kf \cdot (lb)^{A_f}) / (FREQUENCY)
\]

\[
Ic = 2 \cdot q \cdot Ic
\]
Bipolar transistor equations for temperature effect

\[ I_{st} = I_s(T_n) e^{-\frac{A_b - A_{bq} 0}{V_{gb}} V_{\Delta t}} \]

\[ I_{bft} = I_{bf}(T_n) e^{-\frac{M_{lf} V_{gf}}{(M_{lf})(V_{\Delta t})}} \]

\[ I_{brt} = I_{br}(T_n) e^{-\frac{V_{GC}}{(2V_{\Delta T})}} \]

\[ I_{sst} = I_{ss}(T_n) e^{-\frac{V_{GS}}{V_{\Delta T}}} \]

For power gains, the model uses bandgap difference between emitter and base \((dV\beta F)\) or base and collector \((dV\beta R)\).

\[ \beta_{ft} = \beta_{f}(T_n) e^{-\frac{A_e - A_b - A_{qb} 0}{V_{\Delta T}}} \]

\[ \beta_{rit} = \beta_{rI} e^{-\frac{dV\beta R}{V_{\Delta T}}} \]

Resistances are not constant over temperature. As a result, the resistances have parameters linked to the temperature dependence.

\[ R_{et} = R_{e}(T_n) e^{A_e} \]

\[ R_{bvt} = R_{bv}(T_n) e^{-A_{qb} 0} \]

\[ R_{bct} = R_{bc}(T_n) e^{A_{ex}} \]

\[ R_{cct} = R_{cc}(T_n) e^{A_c} \]

\[ R_{cvt} = R_{cv}(T_n) e^{A_{epi}} \]

\[ V_{dt} = (-3V_t)\ln(T_n) + (V_d)(T_n) + (1 - T_n)V_g \]

The following equation gives the scaling factor of capacitances after temperature scaling of
the diffusion voltages is done.

\[ C_{jt} = C_j \left( \frac{V_d}{V_{dt}} \right)^\beta \]

where \( \rho \) is the grading coefficient.

### Quasi saturation/high injection effect equations

Quasi saturation or high injection effect can occur due to ohmic resistance or space-charge limited resistance in the epilayer region. If the resistance is due to space-charge, the effect is also known as Kirk effect.

The quasi saturation voltage drop is given by the following equation:

\[ V_{qs} = V_{dc} - Vb2c1 = \int_0^{W_{epi}} E(X) dx \]

The current is given by the following equation:

\[ I_{qs} = \frac{(V_{qs})}{(R_{cv})} \]

For higher currents, the equation is given by:

\[ I_{qs} = \frac{(V_{qs})}{(SCR_{cv})} \]

### Current crowding equations

Following is the general DC current crowding equation:

\[ I(x) = \frac{2 \cdot V_t \cdot L_{em}}{\rho \cdot H_{em}} Z \tan[Z(1 - x/(H_{em}))] \]

where,

- \( L_{em} \) = emitter length
- \( H_{em} \) = emitter width
- \( \rho \) = pinch resistance
- \( Z \) = integration constant

For the boundary condition, \( I(x=H_{em})=0 \), the equation is:
The voltage is given by the following equation:

\[ e^{V/(Vt)} = \frac{Z}{\tan Z \cos^2 [Z(1-x/(Hem))]} \]

In the low current limit \( Z \) is small and the equation is:

\[ \frac{Vb1b2}{Ib} = \frac{\rho Hem}{3Lem} = Rbv \]

In the high current limit \( Z \to \pi/2 \) and the equation is:

\[ (e^{(Vb1b2)/(Vt)}) = \frac{Z\tan Z}{\sin Z} \to (Z\tan Z = Ib\frac{\rho Hem}{2VtLem}) \]

The current is given by:

\[ Ib = \frac{2Vt}{3Rbv} e^{(Vb1b2)/(Vt)} \]

By interpolating between the high and low current limits, we can derive the following equation:

\[ Ib = \frac{1}{3Rbv} [2Vt(e^{(Vb1b2)/(Vt)} - 1) + Vb1b2] \]

The resistance seen by the current is given by the following equations:

\[ Rb2 = \frac{3Rbv}{qb} \]

\[ Ib1b2 = \frac{1}{Rb2}[2Vt(e^{(Vb1b2)/(Vt)} - 1) + Vb1b2] \]
Bipolar transit time equations

The transit time for the base for closely related knee current is given by the following equation:

\[ TAU_{b} = TAU_{b} \cdot tn^{A_{b}0 + A_{b}^{-1}} \]

Similarly, the transit time for the epilayer is given by the following equation:

\[ TAU_{epit} = TAU_{epi} \cdot tn^{A_{epi}^{-1}} \]

The reverse transmit time is given by the following equation:

\[ TAU_{rt} = TAU_{r} \cdot \frac{TAU_{bt} + TAU_{epit}}{TAU_{b} + TAU_{epi}} \]

The emitter charge is given by the following equation:

\[ Q_{e} = TAU_{e} \cdot \sqrt{IsIk} \cdot e^{(V_{b}2e1)/(2V_{t})} \]

Therefore, the emitter transit time is given by the following equation:

\[ TAU_{et} = TAU_{e} \cdot tn^{A_{b}^{-2}} \cdot \exp\left[(-dV_{gtaue})/(V_{A}t)\right] \]

References

For more information on bipolar transistor models, refer to:


For a generally detailed discussion of the U.C. Berkeley SPICE models, including the bipolar transistor, refer to:


For a description of the extension for the quasi-saturation effect, refer to:


For more information on the Mextram model, refer to:

For a comparison of Mextram and the Gummel-Poon model, refer to:

Resistor

General form

\[ R<\text{name}> <(+) \text{ node}> <(-) \text{ node}> [\text{model name}] <\text{value}> + [\text{TC} = <\text{TC1}> [, <\text{TC2}>]] \]

Examples

RLOAD 15 0 2K
R2 1 2 2.4E4 TC=.015,-.003
RFDBCK 3 33 RMOD 10K

Model form

\[
\text{.MODEL } <\text{model name}> \text{ RES [model parameters]}\
\]

Arguments and options

(+ ) and (-) nodes

Define the polarity when the resistor has a positive voltage across it.

[model name]

Affects the resistance value; see Resistor value formulas on page 306.

Comments

The first node listed (or pin 1 in Capture) is defined as positive. The voltage across the component is therefore defined as the first node voltage minus the second node voltage.

Positive current flows from the (+) node through the resistor to the (-) node. Current flow from the first node through the component to the second node is considered positive.

Temperature coefficients for the resistor can be specified in-line, as in the second example. If the resistor has a model specified, then the coefficients from the model are used for the temperature updates; otherwise, the in-line values are used. In both cases the temperature coefficients have default values of zero. Expressions cannot be used for the in-line coefficients.

Capture parts

For standard R parts, the effective value of the part is set directly by the VALUE property. For the variable resistor, R_VAR, the effective value is the product of the base value (VALUE) and
multiplier (SET).

In general, resistors should have positive component values (VALUE property). In all cases, components must not be given a value of zero.

However, there are cases when negative component values are desired. This occurs most often in filter designs that analyze an RLC circuit equivalent to a real circuit. When transforming from the real to the RLC equivalent, it is possible to end up with negative component values.

PSpice A/D allows negative component values for bias point, DC sweep, AC, and noise analyses. In the case of resistors, the noise contribution from negative component values come from the absolute value of the component (components are not allowed to generate negative noise). A transient analysis may fail for a circuit with negative components. Negative components may create instabilities in time that the analysis cannot handle.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>resistor</td>
<td>VALUE</td>
<td>resistance</td>
</tr>
<tr>
<td>TC</td>
<td></td>
<td></td>
<td>linear and quadratic temperature coefficients</td>
</tr>
<tr>
<td>TOLERANCE</td>
<td></td>
<td></td>
<td>device tolerance</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>(see <a href="#">tolerance specification</a> on page 61)</td>
</tr>
<tr>
<td>R_VAR</td>
<td>variable resistor</td>
<td>VALUE</td>
<td>base resistance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>SET</td>
<td>multiplier</td>
</tr>
</tbody>
</table>

Note: The RBREAK part must be used if you want a LOT tolerance. In that case, use the Model Editor to edit the RBREAK instance.

**Breakout parts**

For non-stock passive and semiconductor devices, Capture has a set of breakout parts designed for customizing model parameters for simulation. These are useful for setting up
Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

Basic breakout part names consist of the intrinsic PSpice A/D device letter plus the suffix BREAK. By default, the model name is the same as the part name and references the appropriate device model with all parameters set at their default. For instance, the DBREAK part references the DBREAK model, which is derived from the intrinsic PSpice A/D D model (.MODEL DBREAK D). Another approach is to use the model editor to derive an instance model and customize this. For example, you could add device and/or lot tolerances to model parameters.

For breakout part RBREAK, the effective value is computed from a formula that is a function of the specified VALUE property.

<table>
<thead>
<tr>
<th>Device type</th>
<th>Part name</th>
<th>Part library file</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>resistor</td>
<td>RBREAK</td>
<td>breakout.olb</td>
<td>VALUE</td>
<td>resistance</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>MODEL</td>
<td>RES model name</td>
</tr>
</tbody>
</table>
## Resistor model parameters

<table>
<thead>
<tr>
<th>Model parameters¹</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>resistance multiplier</td>
<td></td>
<td>1.0</td>
</tr>
<tr>
<td>TC1</td>
<td>linear temperature coefficient</td>
<td>°C⁻¹</td>
<td>0.0</td>
</tr>
<tr>
<td>TC2</td>
<td>quadratic temperature coefficient</td>
<td>°C⁻²</td>
<td>0.0</td>
</tr>
<tr>
<td>TCE</td>
<td>exponential temperature coefficient</td>
<td>%/°C</td>
<td>0.0</td>
</tr>
<tr>
<td>T_ABS</td>
<td>absolute temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_MEASURED</td>
<td>measured temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_GLOBAL</td>
<td>relative to current temperature</td>
<td>°C</td>
<td></td>
</tr>
<tr>
<td>T_REL_LOCAL</td>
<td>relative to AKO model temperature</td>
<td>°C</td>
<td></td>
</tr>
</tbody>
</table>

¹ For information on T_MEASURED, T_ABS, T_REL_GLOBAL, and T_REL_LOCAL, see .MODEL (model definition) on page 58.
Resistor equations

Resistor value formulas

One
If [model name] is included and TCE is specified, then the resistance is given by:

\[ <\text{value}> \cdot R \cdot 1.01^{\text{TCE} \cdot (T - T_{\text{nom}})} \]

where \(<\text{value}>\) is normally positive (though it can be negative, but not zero). \(T_{\text{nom}}\) is the nominal temperature (set using TNOM option).

Two
If [model name] is included and TCE is not specified, then the resistance is given by:

\[ <\text{value}> \cdot R \cdot (1 + TC1 \cdot (T - T_{\text{nom}}) + TC2 \cdot (T - T_{\text{nom}})^2) \]

where \(<\text{value}>\) is usually positive (though it can be negative, but not zero).

Resistor equation for noise
Noise is calculated assuming a 1.0-hertz bandwidth. The resistor generates thermal noise using the following spectral power density (per unit bandwidth):

\[ i^2 = 4 \cdot k \cdot T \text{/resistance} \]
Voltage-Controlled switch

**General form**

\[ S<name> <(+) switch node> <(-) switch node> \\
+ <(+) controlling node> <(-) controlling node> \\
+ <model name> \]

**Examples**

- S12 13 17 2 0 SMOD
- SESET 5 0 15 3 RELAY

**Model form**

```
.MODEL <model name> VSWITCH [model parameters]
```

**Description**

The voltage-controlled switch can function either as a variable-resistance switch or a short-transition switch. The type of switching characteristic is determined by the specific model parameters used. Under most circumstances it is recommended that the variable-resistance mode be used. The switch model was designed to minimize numerical problems. However, there are a few things to consider; see **Special considerations** on page 309.

![Switch Diagram]

**Comments**

The resistance between the <(+) switch node> and <(-) switch node> depends on the voltage between the <(+) controlling node> and <(-) controlling node>. In the variable-resistance mode, the resistance varies continuously between \( R_{ON} \) and \( R_{OFF} \) during the switching transition. For the short-transition switch, the resistance switches between \( R_{ON} \) and \( R_{OFF} \) in the shortest possible time or voltage increment.

A resistance of \( 1/G\text{MIN} \) is connected between the controlling nodes to keep them from floating. See the **.OPTIONS (analysis options)** on page 71 statement for setting GMIN.

Although very little computer time is required to evaluate switches, during transient analysis the simulator must step through the transition region using a fine enough step size to get an accurate waveform. Applying many transitions can produce long run times when evaluating the other devices in the circuit at each time step.
Capture parts

Ideal switches

Summarized below are the available voltage-controlled switch part types in the analog library. To create a time-controlled switch, connect the switch control pins to a voltage source with the appropriate voltage vs. time values (transient specification).

<table>
<thead>
<tr>
<th>Part type</th>
<th>Part Name</th>
<th>Model type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage-Controlled Switch</td>
<td>S, S_ST</td>
<td>VSWITCH</td>
</tr>
</tbody>
</table>

The S part defines the on/off resistance and the on/off control voltage thresholds for the variable-resistance switch. This switch has a finite on resistance and off resistance, and it changes smoothly between the two as its control voltage changes. This behavior is important because it allows PSpice A/D to find a continuous set of solutions for the simulation. You can make the on resistance very small in relation to the other circuit impedances, and you can make the off resistance very large in relation to the other circuit impedances.

The S_ST part defines the on/off resistance, the threshold and hysteresis control voltage, and the time delay for the short-transition switch. This switch transitions rapidly between states. As a result, the on and off resistance should have as small a dynamic range as practical.

Variable-Resistance switch model parameters

<table>
<thead>
<tr>
<th>Model Parameters 1</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROFF 2</td>
<td>off resistance</td>
<td>ohm</td>
<td>1E+6</td>
</tr>
<tr>
<td>RON</td>
<td>on resistance</td>
<td>ohm</td>
<td>1.0</td>
</tr>
<tr>
<td>VOFF</td>
<td>control voltage for off state</td>
<td>volt</td>
<td>0.0</td>
</tr>
<tr>
<td>VON</td>
<td>control voltage for on state</td>
<td>volt</td>
<td>1.0</td>
</tr>
</tbody>
</table>

1.  See MODEL (model definition) on page 58.
2.  RON and ROFF must be greater than zero and less than 1/GMIN.
Short-Transition switch model parameters

<table>
<thead>
<tr>
<th>Model Parameters ¹</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROFF ²</td>
<td>off resistance</td>
<td>ohm</td>
<td>1E+12</td>
</tr>
<tr>
<td>RON</td>
<td>on resistance</td>
<td>ohm</td>
<td>1.0</td>
</tr>
<tr>
<td>VT</td>
<td>threshold control voltage</td>
<td>volt</td>
<td>0.0</td>
</tr>
<tr>
<td>VH</td>
<td>hysteresis control voltage</td>
<td>volt</td>
<td>0.0</td>
</tr>
<tr>
<td>TD ³</td>
<td>time delay</td>
<td>sec</td>
<td>0.0</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58.
2. RON and ROFF must be greater than zero and less than 1/GMIN.
3. TD shifts the switching transition to a later time.

Special considerations

- Using double precision numbers, the simulator can only handle a dynamic range of about 12 decades. Making the ratio of ROFF to RON greater than 1E+12 is not recommended.

- For the variable-resistance switch, it is not recommended to make the transition region too narrow. Remember that in the transition region the switch has gain. The narrower the region, the higher the gain and the greater the potential for numerical problems. The smallest allowed value for |VON-VOFF| is RELTOL*(MAX(|VON|, |VOFF|))+ VNTOL.

- The short-transition switch is highly non-linear and can cause large discontinuities to occur in the circuit node voltages and branch currents. A rapid change such as that associated with a switch changing state can cause tolerance problems, leading to erroneous results or time step difficulties. Use switch resistances that are close to ideal, setting them only high and low enough to be negligible with respect to other circuit elements.
Switch equations

In the following equations:

\[ V_c = \text{voltage across control nodes} \]
\[ L_m = \log\text{-mean of resistor values} = \ln((R_{ON} \cdot R_{OFF})^{1/2}) \]
\[ L_r = \log\text{-ratio of resistor values} = \ln(R_{ON}/R_{OFF}) \]
\[ V_m = \text{mean of control voltages} = (V_{ON} + V_{OFF})/2 \]
\[ V_d = \text{difference of control voltages} = V_{ON} - V_{OFF} \]
\[ k = \text{Boltzmann's constant} \]
\[ T = \text{analysis temperature (°K)} \]
\[ S_s = \text{switch state} \]
\[ R_s = \text{switch resistance} \]

Variable-Resistance equations for switch resistance

For: \( V_{ON} > V_{OFF} \)
if:
\[ V_c > V_{ON} \]
then:
\[ R_s = R_{ON} \]
if:
\[ V_c \leq V_{OFF} \]
then:
\[ R_s = R_{OFF} \]
if:
\[ V_{OFF} < V_c < V_{ON} \]
then:
\[ R_s = \exp(L_m + 3 \cdot L_r \cdot (V_c - V_m)/(2 \cdot V_d) - 2 \cdot L_r \cdot (V_c - V_m)^3/V_d^3) \]

For: \( V_{ON} < V_{OFF} \)
if:
\[ V_c < V_{ON} \]
then:
\[ R_s = R_{ON} \]
if:
\[ V_c > V_{OFF} \]
then:
Rs = ROFF

if:
VOFF > Vc > VON
then:
Rs = exp(Lm - 3·Lr·(Vc-Vm)/(2·Vd) + 2·Lr·(Vc-Vm)^3/Vd^2)

Short-Transition equations for switch resistance

If: Ss = off  
for:  
Vc ≥ VT + VH  
then:  
Rs = RON  
Ss = on

Else if: Ss = on  
for:  
Vc < VT - VH  
then:  
Rs = ROFF  
Ss = off

Else: (use the current state)  
Rs = Rs  
Ss = Ss

Voltage-Controlled switch equation for noise

Noise is calculated assuming a 1.0-hertz bandwidth. The voltage-controlled switch generates thermal noise as if it were a resistor having the same resistance that the switch has at the bias point, using the following spectral power density (per unit bandwidth):

\[ \dot{i}^2 = 4·k·T/Rs \]
Transmission line

Description  The transmission line device is a bidirectional delay line with two ports, A and B. The (+) and (-) nodes define the polarity of a positive voltage at a port.

Comments  During transient (.TRAN (transient analysis) on page 113) analysis, the internal time step is limited to be no more than one-half the smallest transmission delay, so short transmission lines cause long run times.

The simulation status window displays the properties of the three shortest transmission lines in a circuit if a transient run’s time step ceiling is set more frequently by one of the transmission lines. This is helpful when you have a large number of transmission lines. The properties displayed are:

- % loss: percent attenuation at the characteristic delay (i.e., the degree to which the line is lossy)
- Time step ceiling: induced by the line
- % of line delay: time step size at percentage of characteristic delay

These transmission line properties are displayed only if they are slowing down the simulation.

For a line that uses a model, the electrical length is given after the model name. Example T5 of Lossy line Examples uses TMOD to specify the line parameters and has an electrical length of one unit.

All of the transmission line parameters from either the ideal or lossy parameter set can be expressions. In addition, R and G can be general Laplace expressions. This allows the user to model frequency dependent effects, such as skin effect and dielectric loss. However, this adds to the computation time for transient analysis, since the impulse responses must be obtained by an inverse FFT instead of analytically.
Ideal line

**General form**

```
T<name> <A port (+) node> <A port (-) node>
+ <B port (+) node> <B port (-) node>
+ [model name]
+ Z0=<value> [TD=<value>] [F=<value> [NL=<value>]]
+ IC= <near voltage> <near current> <far voltage> <far current>
```

**Description**

As shown below, port A’s (+) and (-) nodes are 1 and 2, and port B’s (+) and (-) nodes are 3 and 4, respectively.

**Comments**

For the ideal line, IC sets the initial guess for the voltage or current across the ports. The `<near voltage>` value is the voltage across A(+) and A(-) and the `<far voltage>` is the voltage across B(+) and B(-). The `<near current>` is the current through A(+) and A(-) and the `<far current>` is the current through B(+) and B(-).

For the ideal case, Z0 is the characteristic impedance. The transmission line’s length can be specified either by TD, a delay in seconds, or by F and NL, a frequency and a relative wavelength at F. NL has a default value of 0.25 (F is the quarter-wave frequency). Although TD and F are both shown as optional, one of the two must be specified.

**Note:** Both Z0 (Z-zero) and ZO (Z-O) are accepted by the simulator.
Lossy line

General form

```
T<name> <A port (+) node> <A port (-) node>
+ <B port (+) node> <B port (-) node>
+ [ <model name> [electrical length value] ]
+ LEN=<value> R=<value> L=<value>
+ G=<value> C=<value>
```

Examples

```
T1 1 2 3 4 Z0=220 TD=115ns
T2 1 2 3 4 Z0=220 F=2.25MEG
T3 1 2 3 4 Z0=220 F=4.5MEG NL=0.5
T4 1 2 3 4 LEN=1 R=.311 L=.378u G=6.27u C=67.3p
T5 1 2 3 4 TMOD 1
```

Model form

```
.MODEL <model name> TRN [model parameters]
```

Description

The simulator uses a distributed model to represent the properties of a lossy transmission line. That is, the line resistance, inductance, conductance, and capacitance are all continuously apportioned along the line's length. A common approach to simulating lossy lines is to model these characteristics using discrete passive elements to represent small sections of the line.

This is the lumped model approach, and it involves connecting a set of many small subcircuits in series as shown below:

![Lumped line segment](image)

This method requires that there is enough lumps to adequately represent the distributed character of the line, and this often results in the need for a large netlist and correspondingly long simulation times. The method also produces spurious oscillations near the natural frequencies of the lumped elements.

An additional extension allows systems of coupled transmission lines to be simulated. Transmission line coupling is specified using the K device. This is done in much the same way that coupling is specified for inductors. See the description of Transmission line coupling on page 216 for further details.

The distributed model allows freedom from having to determine how many lumps are sufficient, and eliminates the spurious oscillations. It also allows lossy lines to be simulated in a fraction of the time necessary when using the lumped approach, for the same accuracy.
Comments

For a lossy line, LEN is the electrical length. R, L, G, and C are the per unit length values of resistance, inductance, conductance, and capacitance, respectively.

Example T4 specifies a lossy line one meter long. The lossy line model is similar to that of the ideal case, except that the delayed voltage and current values include terms which vary with frequency. These terms are computed in transient analysis using an impulse response convolution method, and the internal time step is limited by the time resolution required to accurately model the frequency characteristics of the line. As with ideal lines, short lossy lines cause long run times.
Capture parts

Ideal and lossy transmission lines

Listed below are the properties that you can set per instance of an ideal (T) or lossy (TLOSSY) transmission line. The parts contained in the TLINE.OLB part library contain a variety of transmission line types. Their part properties vary.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>transmission line</td>
<td>Z0</td>
<td>characteristic impedance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>TD</td>
<td>transmission delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>frequency for NL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>NL</td>
<td>number of wavelengths or wave number</td>
</tr>
<tr>
<td>TLOSSY1</td>
<td>transmission line</td>
<td>LEN</td>
<td>electrical length</td>
</tr>
<tr>
<td></td>
<td></td>
<td>R</td>
<td>per unit length resistance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L</td>
<td>per unit length inductance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>G</td>
<td>per unit length conductance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C</td>
<td>per unit length capacitance</td>
</tr>
</tbody>
</table>

1. PSpiçe A/D uses a distributed model to represent the properties of a lossy transmission line. That is, the line resistance, inductance, conductance, and capacitance are all continuously apportioned along the line’s length.

A common approach to simulating lossy lines is to model these characteristics using discreet passive elements to represent small sections of the line. This is the lumped model approach, and it involves connecting a set of many small subcircuits in series. This method requires that enough lumps exist to adequately represent the distributed characteristic of the line. This often results in the need for a large netlist and correspondingly long simulation time. The method also produces spurious oscillations near the natural frequencies of the lumped elements.
The distributed model used in PSpice A/D frees you from having to determine how many lumps are sufficient, and eliminates the spurious oscillations. It also allows lossy lines to be simulated with the same accuracy in a fraction of the time required by the lumped approach.

In addition, you can make R and G general Laplace expressions. This allows frequency dependent effects to be modeled, such as skin effect and dielectric loss.

**Coupled transmission lines**

Listed below are the properties that you can set per instance of a coupled transmission line part. The part library provides parts that can accommodate up to five coupled transmission lines. You can also create new parts that have up to ten coupled lines.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>T2COUPLED</td>
<td>coupled transmission line—symmetric</td>
<td>LEN</td>
<td>electrical length</td>
</tr>
<tr>
<td>T3COUPLED</td>
<td></td>
<td>R</td>
<td>per unit length resistance</td>
</tr>
<tr>
<td>T4COUPLED</td>
<td></td>
<td>L</td>
<td>per unit length inductance</td>
</tr>
<tr>
<td>T5COUPLED</td>
<td></td>
<td>G</td>
<td>per unit length conductance</td>
</tr>
<tr>
<td>T2COUPLEDX</td>
<td>coupled transmission line—asymmetric</td>
<td>LEN</td>
<td>electrical length</td>
</tr>
<tr>
<td>T3COUPLEDX</td>
<td></td>
<td>R</td>
<td>per unit length resistance</td>
</tr>
<tr>
<td>T4COUPLEDX</td>
<td></td>
<td>L</td>
<td>per unit length inductance</td>
</tr>
<tr>
<td>T5COUPLEDX</td>
<td></td>
<td>G</td>
<td>per unit length conductance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C</td>
<td>per unit length capacitance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LM</td>
<td>per unit length mutual inductance</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CM</td>
<td>per unit length mutual capacitance</td>
</tr>
<tr>
<td>KCOUPLE2</td>
<td>transmission line coupling matrix</td>
<td>T1</td>
<td>name of first coupled line</td>
</tr>
<tr>
<td></td>
<td></td>
<td>T2</td>
<td>name of second coupled line</td>
</tr>
<tr>
<td></td>
<td></td>
<td>LM</td>
<td>per unit length mutual inductance</td>
</tr>
</tbody>
</table>
Simulating coupled lines

Use the K device to simulate coupling between transmission lines. Each of the coupled transmission line parts provided in the standard part library translate to K device and T device declarations in the netlist. PSpice A/D compiles a system of coupled lines by assembling capacitive and inductive coupling matrices from all of the K devices involving transmission lines. Though the maximum order for any one system is ten lines, there is no explicit limitation on the number of separate systems that may appear in one simulation.

The simulation model is accurate for:

- ideal lines
- low-loss lossy lines
- systems of homogeneous, equally spaced high-loss lines

For more information, see Transmission line coupling on page 216.

Simulation considerations

When simulating, transmission lines with short delays can create performance bottlenecks by setting the time step ceiling to a very small value.
If one transmission line sets the time step ceiling frequently, PSpice A/D reports the three lines with the shortest time step. The status window displays the percentage attenuation, step ceiling, and step ceiling as percentage of transmission line delay.

If your simulation is running reasonably fast, you can ignore this information and let the simulation proceed. If the simulation is slowed significantly, you may want to cancel the simulation and modify your design. If the line is lossy and shows negligible attenuation, model the line as ideal instead.
## Transmission line model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>for all transmission lines</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IC</td>
<td>Sets the initial condition and all four values must be entered. Four values are expected when IC is specified: the near-end voltage, the near-end current, the far-end voltage, and the far-end current, given in that order.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>for ideal transmission lines</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ZO</td>
<td>characteristic impedance</td>
<td>ohms</td>
<td>none</td>
</tr>
<tr>
<td>TD</td>
<td>transmission delay</td>
<td>seconds</td>
<td>none</td>
</tr>
<tr>
<td>F</td>
<td>frequency for NL</td>
<td>Hz</td>
<td>none</td>
</tr>
<tr>
<td>NL</td>
<td>relative wavelength</td>
<td>none</td>
<td>0.25</td>
</tr>
<tr>
<td><strong>for lossy transmission lines</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>R</td>
<td>per unit length resistance</td>
<td>ohms/unit length</td>
<td>none</td>
</tr>
<tr>
<td>L</td>
<td>per unit length inductance</td>
<td>henries/unit length</td>
<td>none</td>
</tr>
<tr>
<td>G</td>
<td>per unit length conductance</td>
<td>mhos/unit length</td>
<td>none</td>
</tr>
<tr>
<td>C</td>
<td>per unit length capacitance</td>
<td>farads/unit length</td>
<td>none</td>
</tr>
<tr>
<td>LEN***</td>
<td>physical length</td>
<td>agrees with RLGC *</td>
<td>none</td>
</tr>
</tbody>
</table>

1. See MODEL (model definition) on page 58. The order is from the most commonly used to the least commonly used parameter.
2. Any length units can be used, but they must be consistent. For instance, if LEN is in feet, then the units of R must be in ohms/foot.

*** A lossy line with $R=G=0$ and $LEN=1$ is equivalent to an ideal line with $Z_0 = \sqrt{\frac{L}{C}}$ and $TD = LEN \cdot \sqrt{\frac{L}{C}}$.

References

For more information on how the lossy transmission line is implemented, refer to:

Independent voltage source & stimulus

The Independent Current Source & Stimulus (I) and the Independent Voltage Source & Stimulus (V) devices have the same syntax. See Independent current source & stimulus Independent voltage source & stimulus on page 180.

Current-Controlled switch

**General form**

W<name> <(+) switch node> <(-) switch node>
+ <controlling V device name> <model name>

**Examples**

W12 13 17 VC WMOD
WRESET 5 0 VRESET RELAY

**Model form**

.MODEL <model name> ISWITCH [model parameters]

**Description**

The current-controlled switch can function either as a variable-resistance switch or a short-transition switch. The type of switching characteristic is determined by the specific model parameters used. Under most circumstances it is recommended that the variable-resistance mode be used. The switch model was designed to minimize numerical problems. However, there are a few things to consider; see Special considerations on page 325.
The resistance between the <+> switch node> and <+(-) switch node> depends on the current through the <+controlling V device name> source. In the variable-resistance mode, the resistance varies continuously between RON and ROFF during the switching transition. For the short-transition switch, the resistance switches between RON and ROFF in the shortest possible time or voltage increment.

A resistance of 1/GMIN is connected between the controlling nodes to keep them from floating. See .OPTIONS (analysis options) on page 71 for information on setting GMIN.

Although very little computer time is required to evaluate switches, during transient analysis the simulator must step through the transition region using a fine enough step size to get an accurate waveform. Having many transitions can produce long run times when evaluating the other devices in the circuit for many times.
Capture parts

Ideal switches

Summarized below are the available current-controlled switch part types in the analog library. To create a time-controlled switch, connect the switch control pins to a voltage source with the appropriate voltage vs. time values (transient specification).

<table>
<thead>
<tr>
<th>Part type</th>
<th>Part name</th>
<th>Model type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Current-controlled switch</td>
<td>W, W_ST</td>
<td>ISWITCH</td>
</tr>
</tbody>
</table>

The W part defines the on/off resistance and the on/off control current thresholds for the variable-resistance switch. This switch has a finite on resistance and off resistance, and it changes smoothly between the two as its control current changes. This behavior is important because it allows PSpice A/D to find a continuous set of solutions for the simulation. You can make the on resistance very small in relation to the other circuit impedances, and you can make the off resistance very large in relation to the other circuit impedances.

The W_ST part defines the on/off resistance, the threshold and hysteresis control current, and the time delay for the short-transition switch. This switch transitions rapidly between states. As a result, the on and off resistance should have as small a dynamic range as practical.

As with current-controlled sources (F, FPOLY, H, and HPOLY), the W part and the W_ST part contain a current-sensing voltage source, which when netlisted, generate two device declarations to the circuit file set:

- one for the controlled switch
- one for the independent current-sensing voltage source

If you want to create a new part for a current-controlled switch (with, for example, different on/off resistance and current threshold settings in the ISWITCH model), the TEMPLATE property must account for the additional current-sensing voltage source.
Variable-Resistance switch model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>IOff</td>
<td>control current for off state</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>ION</td>
<td>control current for on state</td>
<td>amp</td>
<td>1E-3</td>
</tr>
<tr>
<td>ROff^2</td>
<td>off resistance</td>
<td>ohm</td>
<td>1E+6</td>
</tr>
<tr>
<td>RON</td>
<td>on resistance</td>
<td>ohm</td>
<td>1.0</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58.  
2. RON and Roff must be greater than zero and less than 1/GMIN

Short-Transition switch model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>IT</td>
<td>threshold control current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>IH</td>
<td>hysteresis control current</td>
<td>amp</td>
<td>0.0</td>
</tr>
<tr>
<td>ROff^2</td>
<td>off resistance</td>
<td>ohm</td>
<td>1E+12</td>
</tr>
<tr>
<td>RON</td>
<td>on resistance</td>
<td>ohm</td>
<td>1.0</td>
</tr>
<tr>
<td>TD^3</td>
<td>time delay</td>
<td>ohm</td>
<td>0.0</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58.  
2. RON and Roff must be greater than zero and less than 1/GMIN  
3. TD shifts the switching transition to a later time

Special considerations

Using double precision numbers, the simulator can handle only a dynamic range of about 12 decades. Therefore, it is not recommended making the ratio of Roff to RON greater than 1.0E+12.
For the variable-resistance switch, it is not recommended making the transition region too narrow. Remember that in the transition region the switch has gain. The narrower the region, the higher the gain and the greater the potential for numerical problems. The smallest allowed value for $|I_{ON} - I_{OFF}|$ is $\text{RELTOL} \cdot \left( \max( |I_{ON}|, |I_{OFF}| ) \right) + \text{ABSTOL}$.

The short-transition switch is highly non-linear and can cause large discontinuities to occur in the circuit node voltages and branch currents. A rapid change such as that associated with a switch changing state can cause tolerance problems, leading to erroneous results or time step difficulties. Use switch resistances that are close to ideal, setting them only high and low enough to be negligible with respect to other circuit elements.

**Switch equations**

In the following equations:

- $I_c$ = controlling current
- $L_m$ = log-mean of resistor values = $\ln(R_{ON} \cdot R_{OFF})^{1/2}$
- $L_r$ = log-ratio of resistor values = $\ln(R_{ON}/R_{OFF})$
- $I_m$ = mean of control currents = $(I_{ON} + I_{OFF})/2$
- $I_d$ = difference of control currents = $I_{ON} - I_{OFF}$
- $k$ = Boltzmann’s constant
- $T$ = analysis temperature (°K)
- $S_s$ = switch state
- $R_s$ = switch resistance
Variable-Resistance equations for switch resistance

For: $I_{ON} > I_{OFF}$

if:
   $I_{c} > I_{ON}$
then:
   $R_{s} = R_{ON}$
if:
   $I_{c} < I_{OFF}$
then:
   $R_{s} = R_{OFF}$
if:
   $I_{OFF} < I_{c} < I_{ON}$
then:
   $R_{s} = \exp\left(L_{m} + 3 \cdot L_{r} \cdot (I_{c} - I_{m})/(2 \cdot I_{d}) - 2 \cdot L_{r} \cdot (I_{c} - I_{m})^{3}/I_{d}^{3}\right)$

For: $I_{ON} < I_{OFF}$

if:
   $I_{c} < I_{ON}$
then:
   $R_{s} = R_{ON}$
if:
   $I_{c} > I_{OFF}$
then:
   $R_{s} = R_{OFF}$
if:
   $I_{OFF} > I_{c} > I_{ON}$
then:
   $R_{s} = \exp\left(L_{m} - 3 \cdot L_{r} \cdot (I_{c} - I_{m})/(2 \cdot I_{d}) + 2 \cdot L_{r} \cdot (I_{c} - I_{m})^{3}/I_{d}^{3}\right)$

Short-Transition equations for switch resistance

If: $SS = \text{off}$

for:
   $V_{c} \geq I_{T} + I_{H}$
then:
   $R_{s} = R_{ON}$
   $S_{s} = \text{on}$
Current-Controlled switch equation for noise

Noise is calculated assuming a 1.0-hertz bandwidth. The current-controlled switch generates thermal noise as if it were a resistor using the same resistance as the switch has at the bias point, using the following spectral power density (per unit bandwidth):

\[ \dot{i}^2 = 4 \cdot k \cdot T / R_s \]
Subcircuit instantiation

Purpose
This statement causes the referenced subcircuit to be inserted into the circuit using the given nodes to replace the argument nodes in the definition. It allows a block of circuitry to be defined once and then used in several places.

General form
X<name> [node]* <subcircuit name> [PARAMS: <<name> = <value>>]*
+ [TEXT: < <name> = <text value> >* ]

Examples
X12 100 101 200 201 DIFFAMP
XBUFF 13 15 UNITAMP
XFOLLOW IN OUT VCC VEE OUT OPAMP
XFELT 1 2 FILTER PARAMS: CENTER=200kHz
X27 A1 A2 A3 Y PLD PARAMS: MNTYMXDLY=1
+ TEXT: JEDEC_FILE=MYJEDEC.JED
XNANDI 25 28 7 MYPWR MYGND PARAMS: IO_LEVEL=2

Arguments and options

<subcircuit name>

The name of the subcircuit's definition. See .SUBCKT (subcircuit) on page 105.

PARAMS:

Passes values into subcircuits as arguments and into expressions inside the subcircuit.

TEXT:

Passes text values into subcircuits and into text expressions inside the subcircuit.

Comments

There must be the same number of nodes in the call as in the subcircuit's definition.

Subcircuit references can be nested; that is, a call can be given to subcircuit A, whose definition contains a call to subcircuit B. The nesting can be to any level, but must not be circular: for example, if subcircuit A's definition contains a call to subcircuit B, then subcircuit B's definition must not contain a call to subcircuit A.
## Operational Amplifiers (OpAmp) Model Parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>BF1</td>
<td>input stage gain</td>
<td></td>
<td>75</td>
</tr>
<tr>
<td>BF2</td>
<td>output stage gain</td>
<td></td>
<td>75</td>
</tr>
<tr>
<td>C1</td>
<td>phase control capacitor</td>
<td>F</td>
<td>8.6e-012</td>
</tr>
<tr>
<td>C2</td>
<td>compensation capacitor</td>
<td>F</td>
<td>3e-011</td>
</tr>
<tr>
<td>CEE</td>
<td>slew-rate limiting capacitor</td>
<td>F</td>
<td>0</td>
</tr>
<tr>
<td>GA</td>
<td>interstage transconductance</td>
<td>H</td>
<td>0.00018</td>
</tr>
<tr>
<td>GB</td>
<td>output stage transconductance</td>
<td></td>
<td>424</td>
</tr>
<tr>
<td>GCM</td>
<td>common-mode transconductance</td>
<td></td>
<td>1.88e-009</td>
</tr>
<tr>
<td>IEE</td>
<td>input stage current</td>
<td>A</td>
<td>1.5e-005</td>
</tr>
<tr>
<td>IS1</td>
<td>saturation current</td>
<td>A</td>
<td>8e-016</td>
</tr>
<tr>
<td>IS2</td>
<td>saturation current</td>
<td>A</td>
<td>8e-016</td>
</tr>
<tr>
<td>RC</td>
<td>series collector resistance</td>
<td>Ohm</td>
<td>5300</td>
</tr>
<tr>
<td>RE</td>
<td>input stage emitter resistance</td>
<td>Ohm</td>
<td>1800</td>
</tr>
<tr>
<td>REE</td>
<td>input stage current source output resistance</td>
<td>Ohm</td>
<td>130000 00</td>
</tr>
<tr>
<td>RO1</td>
<td>output resistor #1</td>
<td>Ohm</td>
<td>50</td>
</tr>
<tr>
<td>RO2</td>
<td>output resistor #2</td>
<td>Ohm</td>
<td>25</td>
</tr>
<tr>
<td>Model parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default</td>
</tr>
<tr>
<td>------------------</td>
<td>---------------------------------------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>RP</td>
<td>power dissipation</td>
<td></td>
<td>18000</td>
</tr>
<tr>
<td>VC</td>
<td>output limiter offset (to Vcc)</td>
<td></td>
<td>2</td>
</tr>
<tr>
<td>VE</td>
<td>output limiter offset (to Vee)</td>
<td></td>
<td>2</td>
</tr>
</tbody>
</table>
IGBT

General form

\[ Z\langle \text{name} \rangle \langle \text{collector} \rangle \langle \text{gate} \rangle \langle \text{emitter} \rangle \langle \text{model name} \rangle \]
\[ + \left[ \text{AREA} = \langle \text{value} \rangle \right] \left[ \text{WB} = \langle \text{value} \rangle \right] \left[ \text{AGD} = \langle \text{value} \rangle \right] \]
\[ + \left[ \text{KP} = \langle \text{value} \rangle \right] \left[ \text{TAU} = \langle \text{value} \rangle \right] \]

Examples

ZDRIVE 1 4 2 IGBT A AREA=10.1u WB=91u AGD=5.1u KP=0.381
Z231 3 2 9 IGBT27

Model form

.Model <model name> NIGBT [model parameters]

Description

The equivalent circuit for the IGBT is shown below. It is modeled as an intrinsic device (not as a subcircuit) and contains five DC current components and six charge (capacitive) components. An overview of the model equations is included below. For a more detailed description of the defining equations see references [1] through [4] of References.
Capture parts

The following table lists the set of IGBT breakout parts designed for customizing model parameters for simulation. These are useful for setting up Monte Carlo and worst-case analyses with device and/or lot tolerances specified for individual model parameters.

<table>
<thead>
<tr>
<th>Part name</th>
<th>Model type</th>
<th>Property</th>
<th>Property description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZBREAKN</td>
<td>IGBT</td>
<td>AGD</td>
<td>gate-drain overlap area</td>
</tr>
<tr>
<td>AREA</td>
<td></td>
<td>area of the device</td>
<td></td>
</tr>
<tr>
<td>KP</td>
<td></td>
<td>MOS transconductance</td>
<td></td>
</tr>
<tr>
<td>TAU</td>
<td></td>
<td>ambipolar recombination lifetime</td>
<td></td>
</tr>
<tr>
<td>WB</td>
<td></td>
<td>Metallurgical base width</td>
<td></td>
</tr>
<tr>
<td>MODEL</td>
<td></td>
<td>NIGBT model name</td>
<td></td>
</tr>
</tbody>
</table>

Setting operating temperature

Operating temperature can be set to be different from the global circuit temperature by defining one of the model parameters: T_ABS, T_REL_GLOBAL, or T_REL_LOCAL. Additionally, model parameters can be assigned unique measurement temperatures using the T_MEASURED model parameter. For more information, see IGBT model parameters on page 334.
IGBT device parameters

The general form of the IGBT syntax allows for the specification of five device parameters.

These device parameters and their associated default values are defined in previous table. The IGBT model parameters and their associated default values are defined in the table that follows. Model parameters can be extracted from data sheet information by using the model editor. Also, a library of model parameters for commercially available IGBTs is supplied with the software.

The parameters AGD, AREA, KP, TAU, and WB are specified as both device and model parameters, and they cannot be used in a Monte Carlo analysis.

When specified as device parameters, the assigned values take precedence over those which are specified as model parameters. Also, as device parameters (but not as model parameters), they can be assigned a parameter value and used in conjunction with a .DC or .STEP analysis.

<table>
<thead>
<tr>
<th>Device parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGD</td>
<td>gate-drain overlap area</td>
<td>m²</td>
<td>5.0E-6</td>
</tr>
<tr>
<td>AREA</td>
<td>area of the device</td>
<td>m²</td>
<td>1.0E-5</td>
</tr>
<tr>
<td>KP</td>
<td>MOS transconductance</td>
<td>A/V²</td>
<td>0.38</td>
</tr>
<tr>
<td>TAU</td>
<td>ambipolar recombination lifetime</td>
<td>sec</td>
<td>7.1E-6</td>
</tr>
<tr>
<td>WB</td>
<td>metallurgical base width</td>
<td>m</td>
<td>9.0E-5</td>
</tr>
</tbody>
</table>

IGBT model parameters

<table>
<thead>
<tr>
<th>Model parameters¹</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AGD</td>
<td>gate-drain overlap area</td>
<td>m²</td>
<td>5.0E-6</td>
</tr>
<tr>
<td>AREA</td>
<td>area of the device</td>
<td>m²</td>
<td>1.0E-5</td>
</tr>
<tr>
<td>BVF</td>
<td>avalanche uniformity factor</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>BVN</td>
<td>avalanche multiplication exponent</td>
<td>none</td>
<td>4.0</td>
</tr>
<tr>
<td>CGS</td>
<td>gate-source capacitance per unit area</td>
<td>F/cm²</td>
<td>1.24E-8</td>
</tr>
</tbody>
</table>
## Model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>JSNE</td>
<td>emitter saturation current density</td>
<td>A/cm²</td>
<td>6.5E-13</td>
</tr>
<tr>
<td>KF</td>
<td>triode region factor</td>
<td>none</td>
<td>1.0</td>
</tr>
<tr>
<td>KP</td>
<td>MOS transconductance</td>
<td>A/V²</td>
<td>0.38</td>
</tr>
<tr>
<td>NB</td>
<td>base doping</td>
<td>1/cm³</td>
<td>2.E14</td>
</tr>
<tr>
<td>TAU</td>
<td>ambipolar recombination lifetime</td>
<td>sec</td>
<td>7.1E-6</td>
</tr>
<tr>
<td>THETA</td>
<td>transverse field factor</td>
<td>1/V</td>
<td>0.02</td>
</tr>
<tr>
<td>VT</td>
<td>threshold voltage</td>
<td>V</td>
<td>4.7</td>
</tr>
<tr>
<td>VTD</td>
<td>gate-drain overlap depletion threshold</td>
<td>V</td>
<td>1.E-3</td>
</tr>
<tr>
<td>WB</td>
<td>metallurgical base width</td>
<td>m</td>
<td>9.0E-5</td>
</tr>
</tbody>
</table>

1. See `.MODEL (model definition)` on page 58 statement.
IGBT equations

In the following equations:

\[ I_{\text{mos}} = \text{MOSFET channel current} \]

\[ I_T = \text{anode current} \]

\[ I_{\text{css}} = \text{steady-state (bipolar) collector current} \]

\[ I_{\text{bss}} = \text{Steady-state base current} \]

\[ I_{\text{mult}} = \text{avalanche multiplication current} \]

\[ R_b = \text{conductivity modulated base resistance} \]

\[ b = \text{ambipolar mobility ratio} \]

\[ D_p = \text{diffusion coefficient for holes} \]

\[ W = \text{quasi-neutral base width} \]

\[ Q_{eb} = \text{instantaneous excess carrier base charge} \]

\[ Q_b = \text{background mobile carrier charge} \]

\[ n_i = \text{intrinsic carrier concentration} \]

\[ M = \text{avalanche multiplication factor} \]

\[ I_{\text{gen}} = \text{(bipolar)collector-base thermally generated current} \]

\[ \varepsilon_{\text{si}} = \text{dielectric permittivity of silicon} \]

\[ q = \text{electron charge} \]

\[ W_{bcj} = \text{base (bipolar) to collector depletion width} \]
IGBT equations for DC current

MOSFET channel current

\[
IMOS = \begin{cases} 
0 & \text{For } V_{gs} < V_T \\
\frac{K_F \cdot K_P \cdot \left((V_{gs} - V_T) \cdot V_{ds} - \frac{K_F \cdot V_{ds}^2}{2}\right)}{1 + \theta \cdot (V_{gs} - V_T)} & \text{For } V_{ds} \leq \frac{(V_{gs} - V_T)}{K_F} \\
\frac{K_P \cdot (V_{gs} - V_T)^2}{2 \cdot (1 + \theta \cdot (V_{gs} - V_T))} & \text{For } V_{ds} > \frac{(V_{gs} - V_T)}{K_F}
\end{cases}
\]

anode current: current through the resistor \( R_b \)

\[
I_T = \frac{V_{Ce}}{R_b}
\]

steady-state collector current

\[
I_{c_{ss}} = \begin{cases} 
0 & \text{For } V_{eb} \leq 0 \\
\left(\frac{1}{1 + b}\right) \cdot I_T + \left(\frac{b}{1 + b}\right) \cdot \left(\frac{4 \cdot D_p}{W^2}\right) \cdot Q_{eb} & \text{For } V_{eb} > 0
\end{cases}
\]

steady-state base current

\[
l_{b_{ss}} = \begin{cases} 
0 & \text{For } V_{eb} \leq 0 \\
\frac{Q_{eb}}{\tau} + \left(\frac{Q_{eb}^2}{Q_B}\right) \cdot \left(\frac{4 \cdot N_B^2}{n_i^2}\right) \cdot (JSNE \cdot AREA) & \text{For } V_{eb} > 0
\end{cases}
\]

avalanche multiplication current
\[ I_{mult} = (M - 1) \cdot (I_{mos} + I_{css}) + M \cdot I_{gen} \]

**IGBT equations for capacitance**

**gate source**

\[ C_{gs} = CGS \]

\[ Q_{gs} = CGS \cdot V_{gs} \]

**drain source**

\[ C_{ds} = \frac{(\text{AREA} - \text{AGD}) \cdot \varepsilon_{si}}{W_{dsj}} \]

\[ Q_{ds} = q \cdot (\text{AREA} - \text{AGD}) \cdot \text{NB} \cdot W_{dsj} \]

where:

\[ W_{dsj} = \sqrt{\frac{2 \cdot \varepsilon_{si} \cdot (V_{ds} + 0.6)}{q \cdot \text{NB}}} \]

**gate drain**

For \( V_{ds} < V_{gs} - V_{TD} \)

\[ C_{dg} = \text{COXD} \]

\[ Q_{dg} = \text{COXD} \cdot V_{dg} \]

For \( V_{ds} \geq V_{gs} - V_{TD} \)

\[ C_{dg} = \frac{C_{dgj} \cdot \text{COXD}}{C_{dgj} + \text{COXD}} \]

\[ Q_{dg} = \frac{q \cdot \text{NB} \cdot \varepsilon_{si} \cdot \text{AGD}^2}{\text{COXD}} \left( \frac{\text{COXD} \cdot W_{dgj}}{\varepsilon_{si} \cdot \text{AGD}} - \log \left( 1 + \frac{\text{COXD} \cdot W_{dgj}}{\varepsilon_{si} \cdot \text{AGD}} \right) \right) - \text{COXD} \cdot V_{TD} \]

where:
\[ C_{dgj} = \frac{AGD \cdot \varepsilon_{si}}{W_{dgj}} \]

\[ W_{dgj} = \sqrt{\frac{2 \cdot \varepsilon_{si} \cdot (V_{dg} + V_{TD})}{q \cdot NB}} \]

**Ccer**

\[ C_{cer} = \frac{Q_{eb} \cdot C_{bcj}}{3 \cdot Q_B} \quad \text{and} \quad C_{bcj} = \frac{\varepsilon_{si} \cdot \text{AREA}}{W_{bcj}} \]

**Cmult**

\[ C_{mult} = (M - 1) \cdot C_{cer} \quad \text{and} \quad Q_{mult} = (M - 1) \cdot Q_{cer} \]

**emitter base**

\[ C_{eb} = \frac{dQ_{eb}}{dV_{eb}} \]

**References**

For more information on the IGBT model, refer to:


Battery Model

General form

.X awbflooded_cell PARAMS VOC=<value> AH=<value> SOC=<value>
.X awbflooded_cell PARAMS VOC=<value> AH=<value> SOC=<value>.....
.X awbvalve_regulated_cell PARAMS VOC=<value> AH=<value> SOC=<value>

Examples

.X awbflooded_cell PARAMS VOC=5 AH=15 SOC=0.8

Arguments and options
awbflooded_cell

PSpice model for modelling the flooded cell batteries. In the flooded cells batteries since the gases created during charging are vented to the atmosphere, distilled water must be added occasionally to bring the electrolyte back to its required level.

Example: 12-V automobile battery.

awbvalve_regulated_cell

PSpice model for modelling the valve regulated batteries.

VOC

Indicates the open circuit voltage. This is the voltage across the two terminals of the battery when the battery is not connected to a circuit.

AH

It is the ampere hour of the battery. This is the amount of time for which a battery operates without having to recharge it. For example, if a battery is marked 300Ah then it is assumed that the battery can supply 20A current for 15 hours or 10A current for 30 hrs.

**Note:** An ampere hour (Ah) indicates the amount of energy charge in a battery that will allow one ampere of current to flow for one hour.

SOC

Indicates the state of charge in a battery. For a completely charged battery, SOC is 100% and for a fully discharged battery, SOC is 0%.

**Capture Parts**
# Digital devices

<table>
<thead>
<tr>
<th>Behavioral primitives on page 397</th>
<th>Multi-bit A/D and D/A converter on page 393</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bidirectional transfer gates on page 362</td>
<td>Programmable logic array on page 379</td>
</tr>
<tr>
<td>Delay line on page 378</td>
<td>Pullup and pulldown on page 377</td>
</tr>
<tr>
<td>Digital input (N device) on page 434</td>
<td>Random access read-write memory on page 388</td>
</tr>
<tr>
<td>Digital output (O Device) on page 439</td>
<td>Read only memory on page 384</td>
</tr>
<tr>
<td>File stimulus on page 425</td>
<td>Standard gates on page 356</td>
</tr>
<tr>
<td>Flip-flops and latches on page 366</td>
<td>Stimulus generator on page 418</td>
</tr>
<tr>
<td>Input/output model on page 431</td>
<td>Tristate gates on page 360</td>
</tr>
</tbody>
</table>
Digital device summary

<table>
<thead>
<tr>
<th>Device class</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>primitives</td>
<td>U</td>
<td>low-level digital devices (e.g., gates and flip-flops)</td>
</tr>
<tr>
<td>stimuli</td>
<td>U</td>
<td>digital stimulus generators</td>
</tr>
<tr>
<td></td>
<td></td>
<td>file-based stimulus</td>
</tr>
<tr>
<td>interface</td>
<td>N</td>
<td>digital input device</td>
</tr>
<tr>
<td></td>
<td>O</td>
<td>digital output device</td>
</tr>
</tbody>
</table>

Primitives are primarily used in subcircuits to model complete devices.

Stimulus devices are used in the circuit to provide input for other digital devices during the simulation.

Interface devices are mainly used inside subcircuits that model analog/digital and digital/ analog interfaces.

Note: The digital devices are part of the digital simulation feature of PSpice A/D. For more information on digital simulation and creating models, refer to your PSpice User’s Guide.
Digital primitive summary

Digital primitives are low-level devices whose main use is modeling off-the-shelf parts, often in combination with each other.

Digital primitives should not be confused with the subcircuits in the libraries that use them. For instance, the 74LS00 subcircuit in 74ls.lib uses a NAND digital primitive to model the 74LS00 part, but it also includes timing and interface information that makes the model adapted for use in a circuit simulation. For more information, refer to your PSpice User’s Guide.

This section provides a reference for each of the digital primitives supported by the simulator, to help you create digital parts that are not in the model library.

<table>
<thead>
<tr>
<th>Primitive class</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard gates</td>
<td>BUF</td>
<td>buffer</td>
</tr>
<tr>
<td></td>
<td>INV</td>
<td>inverter</td>
</tr>
<tr>
<td></td>
<td>AND</td>
<td>AND gate</td>
</tr>
<tr>
<td></td>
<td>NAND</td>
<td>NAND gate</td>
</tr>
<tr>
<td></td>
<td>OR</td>
<td>OR gate</td>
</tr>
<tr>
<td></td>
<td>NOR</td>
<td>NOR gate</td>
</tr>
<tr>
<td></td>
<td>XOR</td>
<td>exclusive OR gate</td>
</tr>
<tr>
<td></td>
<td>NXOR</td>
<td>exclusive NOR gate</td>
</tr>
<tr>
<td></td>
<td>BUFA</td>
<td>buffer array</td>
</tr>
<tr>
<td></td>
<td>INVA</td>
<td>inverter array</td>
</tr>
<tr>
<td></td>
<td>ANDA</td>
<td>AND gate array</td>
</tr>
<tr>
<td></td>
<td>NANDA</td>
<td>NAND gate array</td>
</tr>
<tr>
<td></td>
<td>ORA</td>
<td>OR gate array</td>
</tr>
<tr>
<td></td>
<td>NORA</td>
<td>NOR gate array</td>
</tr>
<tr>
<td></td>
<td>XORA</td>
<td>exclusive OR gate</td>
</tr>
<tr>
<td></td>
<td>NXORA</td>
<td>exclusive NOR gate array</td>
</tr>
<tr>
<td></td>
<td>AO</td>
<td>AND-OR compound gate</td>
</tr>
<tr>
<td></td>
<td>OA</td>
<td>OR-AND compound gate</td>
</tr>
<tr>
<td></td>
<td>AOI</td>
<td>AND-NOR compound gate</td>
</tr>
<tr>
<td></td>
<td>OAI</td>
<td>OR-NAND compound gate</td>
</tr>
<tr>
<td>Primitive class</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-------</td>
<td>---------------------------</td>
</tr>
<tr>
<td><strong>Tristate gates</strong> on page 360</td>
<td>BUF3</td>
<td>buffer</td>
</tr>
<tr>
<td></td>
<td>INV3</td>
<td>inverter</td>
</tr>
<tr>
<td></td>
<td>AND3</td>
<td>AND gate</td>
</tr>
<tr>
<td></td>
<td>NAND3</td>
<td>NAND gate</td>
</tr>
<tr>
<td></td>
<td>OR3</td>
<td>OR gate</td>
</tr>
<tr>
<td></td>
<td>NOR3</td>
<td>NOR gate</td>
</tr>
<tr>
<td></td>
<td>XOR3</td>
<td>exclusive OR gate</td>
</tr>
<tr>
<td></td>
<td>NXOR3</td>
<td>exclusive NOR gate</td>
</tr>
<tr>
<td></td>
<td>BUF3A</td>
<td>buffer array</td>
</tr>
<tr>
<td></td>
<td>INV3A</td>
<td>inverter array</td>
</tr>
<tr>
<td></td>
<td>AND3A</td>
<td>AND gate array</td>
</tr>
<tr>
<td></td>
<td>NAND3A</td>
<td>NAND gate array</td>
</tr>
<tr>
<td></td>
<td>OR3A</td>
<td>OR gate array</td>
</tr>
<tr>
<td></td>
<td>NOR3A</td>
<td>NOR gate array</td>
</tr>
<tr>
<td></td>
<td>XOR3A</td>
<td>exclusive OR gate array</td>
</tr>
<tr>
<td></td>
<td>NXOR3A</td>
<td>exclusive NOR gate array</td>
</tr>
<tr>
<td><strong>Bidirectional transfer gates</strong></td>
<td>NBTG</td>
<td>N-channel transfer gate</td>
</tr>
<tr>
<td>on page 362</td>
<td>PBTG</td>
<td>P-channel transfer gate</td>
</tr>
<tr>
<td><strong>Flip-flops and latches</strong> on</td>
<td>JKFF</td>
<td>J-K, negative-edge triggered</td>
</tr>
<tr>
<td>page 366</td>
<td>DFF</td>
<td>D-type, positive-edge triggered</td>
</tr>
<tr>
<td></td>
<td>SRFF</td>
<td>S-R gated latch</td>
</tr>
<tr>
<td></td>
<td>DLTCH</td>
<td>D gated latch</td>
</tr>
<tr>
<td><strong>Pullup and pulldown</strong> on page</td>
<td>PULLUP</td>
<td>pullup resistor array</td>
</tr>
<tr>
<td>377</td>
<td>PULLDN</td>
<td>pulldown resistor array</td>
</tr>
<tr>
<td><strong>Delay line</strong> on page 378</td>
<td>DLYLINE</td>
<td>delay line</td>
</tr>
<tr>
<td>Primitive class</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------</td>
<td>------------------------------------</td>
</tr>
<tr>
<td>Programmable logic array on page 379</td>
<td>PLAND</td>
<td>AND array</td>
</tr>
<tr>
<td></td>
<td>PLOR</td>
<td>OR array</td>
</tr>
<tr>
<td></td>
<td>PLXOR</td>
<td>exclusive OR array</td>
</tr>
<tr>
<td></td>
<td>PLNAND</td>
<td>NAND array</td>
</tr>
<tr>
<td></td>
<td>PLNOR</td>
<td>NOR array</td>
</tr>
<tr>
<td></td>
<td>PLNXOR</td>
<td>exclusive NOR array</td>
</tr>
<tr>
<td></td>
<td>PLANDC</td>
<td>AND array, true and complement</td>
</tr>
<tr>
<td></td>
<td>PLORC</td>
<td>OR array, true and complement</td>
</tr>
<tr>
<td></td>
<td>PLXORC</td>
<td>exclusive OR array, true and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>complement</td>
</tr>
<tr>
<td></td>
<td>PLNANDC</td>
<td>NAND array, true and complement</td>
</tr>
<tr>
<td></td>
<td>PLNORC</td>
<td>NOR array, true and complement</td>
</tr>
<tr>
<td></td>
<td>PLNXORC</td>
<td>exclusive NOR array, true and</td>
</tr>
<tr>
<td></td>
<td></td>
<td>complement</td>
</tr>
</tbody>
</table>

| Read only memory on page 384 | ROM   | read-only memory                   |
| Random access read-write memory on page 388 | RAM   | random access read-write memory    |
| Multi-bit A/D and D/A converter on page 393 | ADC   | multi-bit A/D converter            |
|                                           | DAC   | multi-bit D/A converter            |
| Behavioral primitives on page 397        | LOGICEXP | logic expression                  |
|                                           | PINDLY | pin-to-pin delay                   |
|                                           | CONSTRAINT | constraint checking               |

The format for specifying a digital primitive follows the general format described in the next section. Primitive-specific formats are also described which includes parameters and nodes that are specific to the primitive type.

Also listed is the specific timing model format for each primitive, along with the appropriate timing model parameters.

For example, the 74393 part provided in the model library is defined as a subcircuit composed of U devices as shown below.
When adding digital parts to a part library, you can create corresponding digital device models by connecting U devices in a subcircuit definition similar to the one shown above. We recommend that these be saved in a custom model file. The model files can then be configured into the model library or specified for use in a given design.
General digital primitive format

The format of digital primitives is similar to that of analog devices. One difference is that most digital primitives use two models instead of one. One of the models is the timing model, which specifies propagation delays and timing constraints, such as setup and hold times. The other model is the I/O model, which specifies information specific to the device's input/output characteristics. The reason for having two models is that, while timing information is specific to a device, the input/output characteristics apply to a whole device family. Thus, many devices in the same family reference the same I/O model, but each device has its own timing model. If wanted, the timing models can be selected among primitives of the same class.

The general digital primitive format is shown below. Each statement can span one or more lines by using the \+(line continuation) on page 127 character in the first column position. Comments can be added to each line by using the ; (in-line comment) on page 126. For specific information on each primitive type, see the sections that follow.

**General form**

```plaintext
U<name> <primitive type> [(<parameter value>*)]
+ <digital power node> <digital ground node>
+ <node>*
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]
```

**Model form**

```
.MODEL <model name> UIO ( <model parameters>* )
```

See Input/output model on page 431 for a list of the UIO model parameters.

**Timing model format**

```
.MODEL <model name> <model type> ( <model parameters>* )
```

**Examples**

```
U1 NAND(2) $G_DPWR $G_DGND 1 2 10 D0_GATE IO_DFT
U2 JKFF(1) $G_DPWR $G_DGND 3 5 200 3 3 10 2 D_293ASTD IO_STD
U3 INV $G_DPWR $G_DGND IN OUT D_INV IO_INV MNTYMXDLY=3
   IO_LEVEL=2
```

**Arguments and options**
<primitive type> [([parameter value]*)]

The type of digital device, such as NAND, JKFF, or INV. It is followed by zero or more parameters specific to the primitive type, such as number of inputs. The number and meaning of the parameters depends on the primitive type. See the sections that follow for a complete description of each primitive type and its parameters.

<digital power node> <digital ground node>

These nodes are used by the interface subcircuits which connect analog nodes to digital nodes or vice versa. Refer to your PSpice User’s Guide for more information.

<node>*

One or more input and output nodes. The number of nodes depends on the primitive type and its parameters. Analog devices, digital devices, or both can be connected to a node. If a node has both analog and digital connections, then the simulator automatically inserts an interface subcircuit to translate between logic levels and voltages. Refer to your PSpice User’s Guide for more information.
<timing model name>

The name of a timing model that describes the device's timing characteristics, such as propagation delay and setup and hold times. Each timing parameter has a minimum, typical, or maximum value which can be selected using the optional MNTYMXDLY device parameter (described below) or the DIGMNTYM option (see .OPTIONS (analysis options) on page 71). The type of the timing model and its parameters are specific to each primitive type and are discussed in the following sections. (Note that the PULLUP, PULLDN, and PINDLY primitives do not have timing models.)

<I/O model name>

The name of an I/O model, which describes the device’s loading and driving characteristics. I/O models also contain the names of up to four DtoA and AtoD interface subcircuits, which are automatically called by the simulator to handle interface nodes. Refer to your PSpice User’s Guide for a more detailed description of I/O models.

<model type>

Is specific to the primitive type. See the specific primitive for the correct <model type> and associated <model parameters>. General timing model issues are discussed in the next section.
MNTYMXDLY

An optional device parameter that selects either the minimum, typical, or maximum delay values from the device’s timing model. A fourth option operates the primitive in Digital Worst-Case (min/max) mode. If not specified, MNTYMXDLY defaults to 0. Valid values are:

- 0 = Current value of .OPTIONS DIGMNTYMX (default=2)
- 1 = Minimum
- 2 = Typical
- 3 = Maximum
- 4 = Worst-case (min/max) timing

IO_LEVEL

An optional device parameter that selects one of the four AtoD or DtoA interface subcircuits from the device’s I/O model. The simulator calls the selected subcircuit automatically in the event a node connecting to the primitive also connects to an analog device. If not specified, IO_LEVEL defaults to 0. Valid values are:

- 0 = the current value of .OPTIONS DIGIOLVL (default=1)
- 1 = AtoD1/DtoA1
- 2 = AtoD2/DtoA2
- 3 = AtoD3/DtoA3
- 4 = AtoD4/DtoA4

Refer to your PSpice User’s Guide for more information.
Timing models

With the exception of the PULLUP, PULLDN, and PINDLY devices, all digital primitives have a timing model that provides timing parameters to the simulator. Within a timing model, there can be one or more types of parameters

- propagation delays (TP)
- setup times (TSU)
- hold times (TH)
- pulse widths (TW)
- switching times (TSW)

Each parameter is further divided into three values: minimum (MN), typical (TY), and maximum (MX). For example, the typical low-to-high propagation delay on a gate is specified as TPLHTY. The minimum data-to-clock setup time on a flip-flop is specified as TSUDCLKMNN.

One or more parameters can be missing from the timing model definition. Data books do not always provide all three (minimum, typical, and maximum) timing specifications. The way the simulator handles missing parameters depends on the type of parameter.

Treatment of unspecified propagation delays

**Note:** This discussion applies only to propagation delay parameters (TP). All other timing parameters, such as setup/hold times and pulse widths, are handled differently and are described in Treatment of unspecified timing constraints on page 354.

Often, only the typical and maximum delays are specified in data books. If, in this case, the simulator were to assume that the unspecified minimum delay just defaults to zero, the logic in certain circuits could break down.

For this reason, the simulator provides two configurable options, DIGMNTYSSCALE and DIGTYMXSSCALE (set using the .OPTIONS (analysis options) on page 71 command), which are used to extrapolate unspecified propagation delays in the timing models.

**DIGMNTYSSCALE**

This option computes the minimum delay when a typical delay is known, using the formula

$$ TPxxMN = \text{DIGMNTYSSCALE} \times TPxxTY $$
DIGMNTYSSCALE has a default value of 0.4, or 40% of the typical delay. Its value must be between 0.0 and 1.0.

DIGTYMXSCALE

This option computes the maximum delay from a typical delay, using the formula

\[ TPxxMX = DIGTYMXSCALE \cdot TPxxTY \]

DIGTYMXSCALE has a default value of 1.6. Its value must be greater than 1.0.

When a typical delay is unspecified, its value is derived from the minimum and/or maximum delays, in one of the following ways. If both the minimum and maximum delays are known, the typical delay is the average of these two values. If only the minimum delay is known, the typical delay is derived using the value of the DIGMNTYSSCALE option. Likewise, if only the maximum delay is specified, the typical delay is derived using DIGTYMXSCALE. Obviously, if no values are specified, all three delays have a default value of zero.

Treatment of unspecified timing constraints

The remaining timing constraint parameters are handled differently from the propagation delays. Often, data books state pulse widths, setup times, and hold times as a minimum value. These parameters do not lend themselves to the extrapolation method used for propagation delays.

Instead, when one or more timing constraints are omitted, the simulator uses the following steps to fill in the missing values:

- If the minimum value is omitted, the default value is zero.
- If the maximum value is omitted, it takes on the typical value if one was specified, otherwise it takes on the minimum value.
- If the typical value is omitted, it is computed as the average of the minimum and maximum values.
Gates

Logic gates come in two types: standard and tristate. Standard gates always have their outputs enabled, whereas tristate gates have an enable control. When the enable control is 0, the output's strength is Z and its level is X.

Logic gates also come in two forms: simple gates and gate arrays. Simple gates have one or more inputs and only one output. Gate arrays contain one or more simple gates in one component. Gate arrays allow one to work directly using parts that have several gates in one package.

The usual Boolean equations apply to these gates having the addition of the X level. The rule for X is: if an input is X, and if changing that input between one and zero would cause the output to change, then the output is also X. In other words, X is only propagated to the output when necessary. For example: 1 AND X = X; 0 AND X = 0; 0 OR X = X; 1 OR X = 1.
Standard gates

Device format

U<name> <gate type> (<parameter value>*)
  + <digital power node> <digital ground node>
  + <input node>* <output node>*
  + <timing model name> <I/O model name>
  + [MNTYMXDLY=<delay select value>]
  + [IO_LEVEL=<interface subckt select value>]

The standard gate types and their parameters are listed in Standard Gate Types on page 358.

Timing model format

Examples

U5 AND(2) $G_DPWR $G_DGND IN0 IN1 OUT two-input AND gate
  + T_AND2 IO_STD
U2 INV $G_DPWR $G_DGND 3 5; simple INVerter
  + T_INV IO_STD
U13 NANDA(2,4) $G_DPWR $G_DGND; four two-input NAND gates
  + INA0 INA1 INB0 INB1 INC0 INC1
  + IND0 IND1 OUTA OUTB OUTC OUTD
  + T_NANDA IO_STD
U9 AO(3,3) $G_DPWR $G_DGND; three-input AND-OR gate
  + INA0 INA1 INA2 INB0 INB1 INB2 INC0 INC1 INC2
  + OUT T_AO IO_STD
  + MNTYMXDLY=1 IO_LEVEL=1
  .MODEL T_AND2 UGATE ; AND2 Timing Model
  + TPLHMN=15ns TPLHTY=20ns TPLHMX=25ns
  + TPHLMN=10ns TPHLTY=15ns TPHLMX=20ns
  +)

Arguments and options

<no. of inputs><no. of gates>

The <no. of inputs> is the number of inputs per gate and <no. of gates> is the number of gates. in* and out* mean one or more nodes, whereas in and out refer to only one node.

In gate arrays the order of the nodes is: all inputs for the first gate, all inputs for the second gate, ..., output for the first gate, output for the second gate, ... In other words, all of the input nodes come first, then all of the output nodes. The total number of input nodes is <no. of inputs>*<no. of gates>; the number of output nodes is <no. of gates>.
A compound gate is a set of <no. of gates> first-level gates which each have <no. of inputs> inputs. Their outputs are connected to a single second-level gate. For example, the AO component has <no. of gates> AND gates whose outputs go into one OR gate. The OR gate’s output is the AO device’s output. The order of the nodes is: all inputs for the first, first-level gate; all inputs for the second, first-level gate; …; the output of the second-level gate. In other words, all of the input nodes followed by the one output node.
## Standard gates

![Standard gates diagram](image)

### Table 3-1 Standard Gate Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Parameters</th>
<th>Nodes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>(&lt;no. of inputs&gt;)</td>
<td>in*, out</td>
<td>AND gate</td>
</tr>
<tr>
<td>ANDA</td>
<td>(&lt;no. of inputs&gt;, &lt;no. of gates&gt;)</td>
<td>in*, out*</td>
<td>AND gate array</td>
</tr>
<tr>
<td>AO</td>
<td>(&lt;no. of inputs&gt;, &lt;no. of gates&gt;)</td>
<td>in*, out</td>
<td>AND-OR compound gate</td>
</tr>
<tr>
<td>AOI</td>
<td>(&lt;no. of inputs&gt;, &lt;no. of gates&gt;)</td>
<td>in*, out</td>
<td>AND-NOR compound gate</td>
</tr>
<tr>
<td>BUF</td>
<td>not applicable</td>
<td>in, out</td>
<td>buffer</td>
</tr>
<tr>
<td>BUFA</td>
<td>(&lt;no. of gates&gt;)</td>
<td>in*, out*</td>
<td>buffer array</td>
</tr>
<tr>
<td>INV</td>
<td>not applicable</td>
<td>in, out</td>
<td>inverter</td>
</tr>
<tr>
<td>INVA</td>
<td>(&lt;no. of gates&gt;)</td>
<td>in*, out*</td>
<td>inverter array</td>
</tr>
<tr>
<td>NAND</td>
<td>(&lt;no. of inputs&gt;)</td>
<td>in*, out</td>
<td>NAND gate</td>
</tr>
<tr>
<td>NANDA</td>
<td>(&lt;no. of inputs&gt;, &lt;no. of gates&gt;)</td>
<td>in*, out*</td>
<td>NAND gate array</td>
</tr>
<tr>
<td>NOR</td>
<td>(&lt;no. of inputs&gt;)</td>
<td>in*, out</td>
<td>NOR gate</td>
</tr>
</tbody>
</table>
**Table 3-1 Standard Gate Types**

<table>
<thead>
<tr>
<th>Type</th>
<th>Parameters</th>
<th>Nodes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORA</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
\(<no. of inputs>,<no. of gates>\)        | in*, out* | NOR gate array                   |
| NXOR   | not applicable                           | in1, in2, out | exclusive NOR gate              |
| NXORA  | 
\(<no. of gates>\)                       | in*, out* | exclusive NOR gate array         |
| OA     | 
\(<no. of inputs>,<no. of gates>\)        | in*, out | OR-AND compound gate             |
| OAI    | 
\(<no. of inputs>,<no. of gates>\)        | in*, out | OR-NAND compound gate            |
| OR     | 
\(<no. of inputs>\)                      | in*, out | OR gate                          |
| ORA    | 
\(<no. of inputs>,<no. of gates>\)        | in*, out* | OR gate array                    |
| XOR    | not applicable                           | in1, in2, out | exclusive OR gate              |
| XORA   | 
\(<no. of gates>\)                       | in*, out* | exclusive OR gate array          |

**Table 3-2 Standard gate timing model parameters**

<table>
<thead>
<tr>
<th>Model parameters(^1)</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPLHMN</td>
<td>delay: low to high, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHTY</td>
<td>delay: low to high, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHMX</td>
<td>delay: low to high, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLMN</td>
<td>delay: high to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLTY</td>
<td>delay: high to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLMX</td>
<td>delay: high to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58
Tristate gates

**Device format**

\[ \text{U<name> <tristate gate type> [( <parameter value>* )]} \]
+ <digital power node> <digital ground node>
+ <input node>* <enable node> <output node>*
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]

**Timing model format**

\[ \text{.MODEL <timing model name> UTGATE [model parameters]} \]

**Examples**

\[ \text{U5 AND3(2) $G_{DPWR}$ $G_{DGND}$ IN0 IN1 ENABLE OUT two-input AND} \]
+ T_TRIAND2 IO_STD
\[ \text{U2 INV3 $G_{DPWR}$ $G_{DGND}$ 3 100 5; INVerter} \]
+ T_TRIINV IO_STD
\[ \text{U13 NAND3A(2,4) $G_{DPWR}$ $G_{DGND}$; four two-input NAND} \]
+ INA0 INA1 INB0 INB1 INC0 INC1 IND0 IND1
+ ENABLE OUTA OUTB OUTC OUTD
+ T_TRINAND IO_STD

**Arguments and options**

- <no. of inputs>
The number of inputs per gate.
- <no. of gates>
The number of gates in model.

**Comments**

In gate arrays the order of the nodes is: all inputs for the first gate, all inputs for the second gate, ..., enable, output for the first gate, output for the second gate, ... In other words, all of the input nodes come first, then the enable, then all of the output nodes. The total number of input nodes is <no. of inputs>·<no. of gates>+1; the number of output nodes is <no. of gates>. If a tristate gate is connected to a net that has at least one device input using an INLD I/O model, or a device output using an OUTLD I/O model where both parameters are greater than zero, then that net is simulated as a charge storage net.

**Tristate gate types**

<table>
<thead>
<tr>
<th>Type</th>
<th>Parameters</th>
<th>Nodes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND3</td>
<td>(&lt;no. of inputs&gt;)</td>
<td>in*, en, out</td>
<td>AND gate</td>
</tr>
</tbody>
</table>
### Tristate gate timing model parameters

<table>
<thead>
<tr>
<th>Model parameters(^1)</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPLHMN</td>
<td>Delay: low to high, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHTY</td>
<td>Delay: low to high, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHMX</td>
<td>Delay: low to high, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLMN</td>
<td>Delay: high to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLTY</td>
<td>Delay: high to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLMX</td>
<td>Delay: high to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHZMN</td>
<td>Delay: high to Z, min</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. in* and out*—Mean one or more nodes present. in and out—Refer to only one node. en—Refers to the output enable node.
Bidirectional transfer gates

The bidirectional transfer gate is a passive device that connects or disconnects two nodes. Bidirectional transfer gates have no parameters.

The state of the gate input controls whether the gate connects the two digital nets. The device type NBTG connects the nodes if the gate is one, and disconnects the nodes if the gate is zero. Device type PBTG connects the nodes if the gate is zero and disconnects the nodes if the gate is one.

The I/O Model \texttt{DRVH} and \texttt{DRVRL} parameters are used as a ceiling on the strength of a one or zero, which is passed through a bidirectional transfer gate. If a bidirectional transfer gate is connected to a net which has at least one device input using an \texttt{INLD} I/O model parameter.

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPHZTY</td>
<td>Delay: high to Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHZMX</td>
<td>Delay: high to Z, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLZMN</td>
<td>Delay: low to Z, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLZTY</td>
<td>Delay: low to Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLZMX</td>
<td>Delay: low to Z, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPZLMN</td>
<td>Delay: Z to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPZLTY</td>
<td>Delay: Z to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPZLMX</td>
<td>Delay: Z to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPZHMN</td>
<td>Delay: Z to high, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPZHTY</td>
<td>Delay: Z to high, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPZHMX</td>
<td>Delay: Z to high, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See \texttt{.MODEL} statement.
greater than zero, or a device output using an OUTLD I/O model parameter greater than zero, then that net is simulated as a charge storage net.

**Device format**

U<name> NBTG
+ <digital power node> <digital ground node>
+ <gate node> <channel node 1> <channel node 2>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY = <delay select value>]
+ [IO_LEVEL = <interface subckt select value>]

U<name> PBTG
+ <digital power node> <digital ground node>
+ <gate node> <channel node 1> <channel node 2>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY = <delay select value>]
+ [IO_LEVEL = <interface subckt select value>]

**Examples**

U4 NBTG $G_DPWR $G_DGND GATE SD1 SD2
+ BTG1 IO_BTG
  .MODEL BTG1 UBTG

**Model form**

.MODEL <timing model name> UBTG

**Special behavior when the NBTG or PBTG is connected to an analog device**

If a channel node of one of these bidirectional transfer gates is connected to an analog device, then the bidirectional transfer gate is removed during simulation and is replaced with the digital-to-analog subcircuit specified by the bidirectional transfer gate’s I/O model. Because the bidirectional transfer gate is passive and bidirectional, this digital-to-analog subcircuit must model the behavior of the whole bidirectional transfer gate, not just convert its digital levels to analog signals. Use this format to define the digital-to-analog subcircuit:

```
.SUBCKT <DtoA subckt name> <gate node> <channel node 1> <channel node 2>
  + <digital power node> <digital ground node>
  + params: DRVL=0 DRVH=0 OutLD=0 InLD=0
```

The contents of the subcircuit must model the behavior of the transfer gate in the analog domain, at least for the channel. If the subcircuit’s gate node is connected to analog devices, then PSpice will simulate the gate node as an analog net. If this behavior is not desired (e.g., the gate will be connected to a clock signal, which will slow simulation if it is an analog signal), then the subcircuit should not have any analog devices connected to the gate node.

**Note:** The gate node has the same behavior if it is connected to an analog net as other digital device pins: the analog-to-digital subcircuit specified by the I/O model and IO_LEVEL is connected between the analog net and the gate pin of the device.
Examples

The first example is a subcircuit that models the switch with an analog gate connection. In some circuit topologies, this may cause large parts of a circuit to convert to analog if a single net is connected to an analog part. To avoid this, use the \_D version of the digital-to-analog converter by setting \texttt{IO\_LEVEL} to 3 or 4.

```
.model io\_nbtg uio (drvh=200 drvl=200 inld=10pf outld=15pf
 +      digpower="DIGIFPWR"\texttt{TstoreMN}=10us
 +      inR=10MEG\texttt{drvZ} =5MEG
 +      AtoD1="AtoD\_HC"AtoD2="AtoD\_HC"
 +      AtoD3="AtoD\_HC"AtoD4="AtoD\_HC"
 +      DtoA1="DtoA\_NBTG"DtoA2="DtoA\_NBTG"
 +      DtoA3="DtoA\_NBTG\_D"DtoA4="DtoA\_NBTG\_D"
.model io\_pbtg uio (drvh=200 drvl=200 inld=10pf outld=15pf
 +      digpower="DIGIFPWR"\texttt{TstoreMN}=10us
 +      inR=10MEG\texttt{drvZ} =5MEG
 +      AtoD1="AtoD\_HC"AtoD2="AtoD\_HC"
 +      AtoD3="AtoD\_HC"AtoD4="AtoD\_HC"
 +      DtoA1="DtoA\_PBTG"DtoA2="DtoA\_PBTG"
 +      DtoA3="DtoA\_PBTG\_D"DtoA4="DtoA\_PBTG\_D"
.model io\_nbtgs uio (drvh=200 drvl=200 inld=10pf outld=15pf
 +      digpower="DIGIFPWR"\texttt{TstoreMN}=10us
 +      inR=10MEG\texttt{drvZ} =5MEG
 +      AtoD1="AtoD\_HC"AtoD2="AtoD\_HC"
 +      AtoD3="AtoD\_HC"AtoD4="AtoD\_HC"
 +      DtoA1="DtoA\_NBTG"DtoA2="DtoA\_NBTG"
 +      DtoA3="DtoA\_NBTG\_D"DtoA4="DtoA\_NBTG\_D"
.model io\_pbtgs uio (drvh=200 drvl=200 inld=10pf outld=15pf
 +      digpower="DIGIFPWR"\texttt{TstoreMN}=10us
 +      inR=10MEG\texttt{drvZ} =5MEG
 +      AtoD1="AtoD\_HC"AtoD2="AtoD\_HC"
 +      AtoD3="AtoD\_HC"AtoD4="AtoD\_HC"
 +      DtoA1="DtoA\_PBTG"DtoA2="DtoA\_PBTG"
 +      DtoA3="DtoA\_PBTG\_D"DtoA4="DtoA\_PBTG\_D"
.model btg1 ubtg
```

The next two examples are switch models with digital gate inputs. The digital-to-analog conversion of the gate inputs uses an I/O model (\texttt{HC} in this example) that is defined here, not the I/O model of the device driving the gate.

Use these examples in cases where using an analog input would create too many analog switches. Do not use these when the gate is analog, since this would make an analog-to-digital-to-analog conversion, which may cause invalid simulation results. (This is because the analog gate is squared up before being converted to analog again and applied to the “gate” of the switch.)

```
.subckt DtoA\_NBTG gate sd1 sd2 pwr gnd
 +      \texttt{params}: DRVL=0 DRVH=0 INLD=0 OUTLD=0 VTH=.9 VSAT=1.2
 S1 sd1 sd2 gate gnd nbtg\_smod
 C1 sd1 gnd {.1pf+\texttt{outld}}
 C2 sd2 gnd {.1pf+\texttt{outld}}
 C3 gate gnd {.1pf+\texttt{inld}}
 .model nbtg\_smod vswitch
 + (\texttt{ron}=((\texttt{drvl}+\texttt{drv}h)/2) \texttt{rof}f=1meg \texttt{von}={(VSAT)} \texttt{voff}={(VTH)})
.ends
```

.subckt DtoA_PBTG gate sd1 sd2 pwr gnd
  + params: DRVL=0 DRVH=0 INLD=0 OUTLD=0 VTH=-0.9 VSAT=-1.2
  S1 sd1 sd2 gate pwr pbtg_smod
  C1 sd1 pwr {.1pf+outld}
  C2 sd2 pwr {.1pf+outld}
  C3 gate gnd {.1pf+inld}
  .model pbtg_smod vswitch
  + (ron={(drvl+drvh)/2} roff=1meg von={VSAT} voff={VTH})
.ends

.subckt DtoA_NBTG_D gate sd1 sd2 pwr gnd
  + params: DRVL=0 DRVH=0 INLD=0 OUTLD=0 VTH=.9 VSAT=1.2
  X1 gate gate_a pwr gnd DtoA_HC
  + params: DRVL={DRVL} DRVH={DRVH} CAPACITANCE={INLD}
  S1 sd1 sd2 gate_a gnd nbtg_smidx
  C1 sd1 gnd {.1pf+outld}
  C2 sd2 gnd {.1pf+outld}
  .model nbtg_smidx vswitch
  + (ron={drvl+drvh}/2) roff=1meg von={VSAT} voff={VTH})
.ends

.subckt DtoA_PBTG_D gate sd1 sd2 pwr gnd
  + params: DRVL=0 DRVH=0 INLD=0 OUTLD=0 VTH=-.9 VSAT=-1.2
  X1 gate gate_a pwr gnd DtoA_HC
  + params: DRVL={DRVL} DRVH={DRVH} CAPACITANCE={INLD}
  S1 sd1 sd2 gate_a pwr pbtg_smidx
  C1 sd1 gnd {.1pf+outld}
  C2 sd2 gnd {.1pf+outld}
  .model pbtg_smidx vswitch
  + (ron={drvl+drvh}/2) roff=1meg von={VSAT} voff={VTH})
.ends

October 2012 365 Product Version 16.6
Flip-flops and latches

The simulator supports both edge-triggered and gated flip-flops. Edge-triggered flip-flops change state when the clock changes: on the falling edge for JKFFs, on the rising edge for DFFs. Gated flip-flops are often referred to as latches. The state of gated flip-flops follows the input as long as the clock (gate) is high. The state is frozen when the clock (gate) falls. Multiple flip-flops can be specified in each device. This allows direct modeling of parts which contain more than one flip-flop in a package.

Initialization

By default, at the beginning of each simulation, all flip-flops and latches are initialized to the unknown state (that is, they output an X). Each device remains in the unknown state until explicitly set or cleared by an active-low pulse on either the preset or clear pins, or until a known state is clocked in.

You can override the X start-up state by setting .OPTIONS (analysis options) on page 71 DIGINITSTATE to either zero or one. If set to zero, all flip-flops and latches in the circuit are cleared. Likewise, if set to one, all such devices are preset. Any other values produce the default (X) start-up state. The DIGINITSTATE option is useful in situations where the initial state of the flip-flop is unimportant to the function of the circuit, such as a toggle flip-flop in a frequency divider.

It is important to note that if the initial state is set to zero or one, the device still outputs an X at the beginning of the simulation if the inputs would normally produce an X on the output. For example, if the initial state is set to one, but the clock is an X at time zero, Q and QBar both go to X when the simulation begins.

X-level handling

The truth-table for each type of flip-flop and latch is given in the sections that follow. However, how the flip-flops treat X levels on the inputs is not depicted in the truth tables because it can depend on the state of the device.

The rule is as follows: if an input is X, and if changing that input between one and zero would cause the output to change, then the output is set to X. In other words, X is only propagated to the output when necessary. For example: if Q = 0 and PresetBar = X, then Q → X; but if Q = 1 and PresetBar = X, then Q → 1.
Timing violations

The flip-flop and latch primitives have model parameters which specify timing constraints such as setup/hold times and minimum pulse-widths. If these model parameter values are greater than zero, the simulator compares measured times on the inputs against the specified value. See Standard gate timing model parameters on page 359 and Tristate gate timing model parameters on page 361.

The simulator reports flip-flop timing violations as digital simulation warning messages in the .out file. These messages can also be viewed using the Windows version of Probe.

Edge-triggered flip-flops

The simulator supports four types of edge-triggered flip-flops:

- D-type flip-flop (DFF), which is positive-edge triggered
- J-K flip-flop (JKFF), which is negative-edge triggered
- Dual-edge D flip-flop (DFFDE), which is selectively positive and/or negative edge triggered
Dual-edge J-K flip-flop (JKFFDE), which is selectively positive and/or negative edge triggered

**Device format**

```plaintext
U<name> DFF (<no. of flip-flops>)
+ <digital power node> <digital ground node>
+ <presetbar node> <clearbar node> <clock node>
+ <d node 1> ... <d node n>
+ <q output 1> ... <q output n>
+ <qbar output 1> ... <qbar output n>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]

U<name> JKFF (<no. of flip-flops>)
+ <digital power node> <digital ground node>
+ <presetbar node> <clearbar node> <clockbar node>
+ <j node 1> ... <j node n>
+ <k node 1> ... <k node n>
+ <q output 1> ... <q output n>
+ <qbar output 1> ... <qbar output n>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]

U<name> DFFDE(<no. of flip-flops>)
+ <digital power node> <digital ground node>
+ <presetbar node> <clrbar node> <clock node>
+ <positive-edge enable node> <negative-edge enable node>
+ <d node 1> ... <d node n>
+ <q output 1> ... <q output n>
+ <qbar output 1> ... <qbar output n>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY = <delay select value>]
+ [IO_LEVEL = <interface subckt select value>]

U<name> JKFFDE(<no. of flip-flops>)
+ <digital power node> <digital ground node>
+ <presetbar node> <clrbar node> <clock node>
+ <positive-edge enable node> <negative-edge enable node>
+ <j node 1> ... <j node n>
+ <k node 1> ... <k node n>
+ <q output 1> ... <q output n>
+ <qbar output 1> ... <qbar output n>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY = <delay select value>]
+ [IO_LEVEL = <interface subckt select value>]
```

**Timing model format**

```plaintext
.MODEL <timing model name> UEFF [model parameters]
```
Examples

U5 JKFF(1) $G_DPWR $G_DGND PREBAR CLRBAR CLKBAR
* one JK flip-flop
+ J K Q QBAR
+ T_JKFF IO_STD
U2 DFF(2) $G_DPWR $G_DGND PREBAR CLRBAR CLK
* two DFF flip-flops
+ D0 D1 Q0 Q1 QBAR0 QBAR1
+ T_DFF IO_STD

.MODEL T_JKFF UEFF(...) ; JK Timing Model

Comments

Use <no. of flip-flops> to specify the number of flip-flops
in the device. The three nodes, <presetbar node>,
<clearbar node>, and <clock(bar) node>, are common
to all flip-flops in the device.

The <positive-edge enable node> and <negative-
edge enable node> are common to all flip-flops in the dual-
edge flip-flops.
### Edge-triggered flip-flop timing model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>THDCLKMN</td>
<td>Hold: (j/k/d) after clk/clkb edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THDCLKTY</td>
<td>Hold: (j/k/d) after clk/clkb edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THDCLKMX</td>
<td>Hold: (j/k/d) after clk/clkb edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCLKQLHMN</td>
<td>Delay: clk/clkb edge to q/qb low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCLKQLHTY</td>
<td>Delay: clk/clkb edge to q/qb low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCLKQLHMX</td>
<td>Delay: clk/clkb edge to q/qb low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCLKQHLMN</td>
<td>Delay: clk/clkb edge to q/qb hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCLKQHLTY</td>
<td>Delay: clk/clkb edge to q/qb hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCLKQHLMX</td>
<td>Delay: clk/clkb edge to q/qb hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQLHMN</td>
<td>Delay: preb/clrb to q/qb low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQLHTY</td>
<td>Delay: preb/clrb to q/qb low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQLHMX</td>
<td>Delay: preb/clrb to q/qb low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQHLMN</td>
<td>Delay: preb/clrb to q/qb hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQHLTY</td>
<td>Delay: preb/clrb to q/qb hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQHLMX</td>
<td>Delay: preb/clrb to q/qb hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDCLKMN</td>
<td>Setup: (j/k/d) to clk/clkb edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDCLKTY</td>
<td>Setup: (j/k/d) to clk/clkb edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDCLKMX</td>
<td>Setup: (j/k/d) to clk/clkb edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUPCCLKHNM</td>
<td>Setup: preb/clrb hi to clk/clkb edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUPCCLKHTY</td>
<td>Setup: preb/clrb hi to clk/clkb edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUPCCLKHMX</td>
<td>Setup: preb/clrb hi to clk/clkb edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>
### Model parameters

<table>
<thead>
<tr>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min preb/clrb width low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min preb/clrb width low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min preb/clrb width low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min clk/clkb width low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min clk/clkb width low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min clk/clkb width low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min clk/clkb width hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min clk/clkb width hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Min clk/clkb width hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Setup: clock enable to clk edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Setup: clock enable to clk edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Setup: clock enable to clk edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Hold: clock enable after clk edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Hold: clock enable after clk edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Hold: clock enable after clk edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58.

### Table 3-3 Edge-triggered flip-flop truth tables DFF

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>D</strong></td>
<td><strong>CLK</strong></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>↑</td>
</tr>
</tbody>
</table>
Table 3-3  Edge-triggered flip-flop truth tables DFF

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>CLK</td>
</tr>
<tr>
<td>1</td>
<td>↑</td>
</tr>
</tbody>
</table>

1. Shows an unstable condition.

Table 3-4  Edge-triggered flip-flop truth tables JKFF

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>J</td>
<td>K</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

1. Shows an unstable condition.


Edge-triggered flip-flop truth tables DFFDE and JKFFDE

### Table 3-5 Dual-edge D flip-flop (DFFDE) truth table

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>CLK</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0</td>
<td>↑</td>
</tr>
<tr>
<td>1</td>
<td>↑</td>
</tr>
<tr>
<td>0</td>
<td>↓</td>
</tr>
<tr>
<td>1</td>
<td>↓</td>
</tr>
</tbody>
</table>

1. Shows an unstable condition.

### Table 3-6 Dual-edge J-K flip-flop (JKFFDE) truth table

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>J</td>
<td>K</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

1. Shows an unstable condition.
Gated latch

The simulator supports two types of gated latches: the S-R flip-flop (SRFF) and the D-type latch (DLTCH).

Device format

```
U<name> SRFF (<no. of flip-flops>)
+ <digital power node> <digital ground node>
+ <presetbar node> <clearbar node> <gate node>
+ <s node 1> ... <s node n>
+ <r node 1> ... <r node n>
+ <q output 1> ... <q output n>
+ <qbar output 1> ... <qbar output n>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]

U<name> DLTCH (<no. of latches>)
+ <digital power node> <digital ground node>
+ <presetbar node> <clearbar node> <gate node>
+ <d node 1> ... <d node n>
+ <q output 1> ... <q output n>
+ <qbar output 1> ... <qbar output n>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]
```

Model form

```
.MODEL <timing model name> UGFF [model parameters]
```

Examples

```
U5 SRFF(4) $G_DPWR $G_DGND PRESET CLEAR GATE
* four S-R latches
+ S0 S1 S2 S3 R0 R1 R2 R3
+ Q0 Q1 Q2 Q3 QB0 QB1 QB2 QB3
+ T_SRFF IO_STD

U2 DLTCH(8) $G_DPWR $G_DGND PRESET CLEAR GATE
* eight D latches
+ D0 D1 D2 D3 D4 D5 D6 D7
+ Q0 Q1 Q2 Q3 Q4 Q5 Q6 Q7
+ QB0 QB1 QB2 QB3 QB4 QB5 QB6 QB7
+ T_DLTCH IO_STD

.MODEL T_SRFF UGFF(...) ; SRFF Timing Model
```

Comments

Use `<no. of flip-flops>` to specify the number of flip-flops in the device. The three nodes, `<presetbar node>`, `<clearbar node>`, and `<gate node>`, are common to all of the flip-flops in the device.

Gated latch timing model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>THDGMN</td>
<td>Hold: s/r/d after gate edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THDGTY</td>
<td>Hold: s/r/d after gate edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Model parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default</td>
</tr>
<tr>
<td>------------------</td>
<td>----------------------------------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>THDGMX</td>
<td>Hold: s/r/d after gate edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDQLHMN</td>
<td>Delay: s/r/d to q/qb low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDQLHTY</td>
<td>Delay: s/r/d to q/qb low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDQHLMX</td>
<td>Delay: s/r/d to q/qb low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDQHLMN</td>
<td>Delay: s/r/d to q/qb hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDQHHTY</td>
<td>Delay: s/r/d to q/qb hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDQHLMX</td>
<td>Delay: s/r/d to q/qb hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPGQLHMN</td>
<td>Delay: gate to q/qb low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPGQLHTY</td>
<td>Delay: gate to q/qb low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPGQLHMX</td>
<td>Delay: gate to q/qb low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPGQHLMN</td>
<td>Delay: gate to q/qb hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPGQHHTY</td>
<td>Delay: gate to q/qb hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPGQHLMX</td>
<td>Delay: gate to q/qb hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQLHMN</td>
<td>Delay: preb/clrb to q/qb low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQLHTY</td>
<td>Delay: preb/clrb to q/qb low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQLHMX</td>
<td>Delay: preb/clrb to q/qb low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQHLMN</td>
<td>Delay: preb/clrb to q/qb hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQHHTY</td>
<td>Delay: preb/clrb to q/qb hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPPCQHLMX</td>
<td>Delay: preb/clrb to q/qb hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDGMN</td>
<td>Setup: s/r/d to gate edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDGTY</td>
<td>Setup: s/r/d to gate edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDGMX</td>
<td>Setup: s/r/d to gate edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUPCGHMN</td>
<td>Setup: preb/clrb hi to gate edge, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUPCGHTY</td>
<td>Setup: preb/clrb hi to gate edge, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUPCGHMX</td>
<td>Setup: preb/clrb hi to gate edge, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWPCLMN</td>
<td>Min preb/clrb width low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWPCLTY</td>
<td>Min preb/clrb width low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>
Gated latch truth tables

The function tables for the SRFF and DLTCH primitives are given below.

### D-type latch (DLTCH) truth table

<table>
<thead>
<tr>
<th>D</th>
<th>GATE</th>
<th>PRE</th>
<th>CLR</th>
<th>Q</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1*</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Q'</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>Q'</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1*</td>
</tr>
</tbody>
</table>

1. Shows an unstable condition.
Pullup and pulldown

The PULLUP and PULLDN primitives function as digital pullup/pulldown resistors. They have no inputs (other than the digital power and ground nodes). Their output is a one level (pullup) or a zero level (pulldown), having a strength determined by the I/O model.

**Device format**

```
U<name> <resistor type> (<number of resistors>)
+ <digital power node> <digital ground node>
+ <output node>*
+ <I/O model name>
+ [IO_LEVEL=<interface subckt select value>]
```

**Examples**

```
U5 PULLUP(4) $G_DPWR $G_DGND ; four pullup resistors
+ BUS0 BUS1 BUS2 BUS3  R1K
U2 PULLDN(1) $G_DPWR $G_DGND ; one pulldown resistor
+ 15 R500
```

**Arguments and options**

- `<resistor type>`
  - One of the following:
    - **PULLUP** pullup resistor array
    - **PULLDN** pulldown resistor array

- `<number of resistors>`
  - Specifies the number of resistors in the array.

**Comments**

Notice that PULLUP and PULLDN do not have Timing Models, just I/O models.
Delay line

The output of a delay line follows the input after the delay specified in the Timing Model. Any width pulse can propagate through a delay line. This behavior is different from gates, which don’t propagate a pulse when its width is less than the propagation delay.

The delay line device has no parameters, and only one input and one output node.

**Device format**

```
U<name> DLYLINE
  + <digital power node> <digital ground node>
  + <input node> <output node>
  + <timing model name> <I/O model name>
  + [MNTYMXDLY=<delay select value>]
  + [IO_LEVEL=<interface subckt select value>]
```

**Examples**

```
U5 DLYLINE $G_DPWR $G_DGND IN OUT; delay line
  + DLY20NS IO_STD
  .MODEL DLY20NS UDLY(; delay line Timing Model
  + DLYMN=20ns DLYTY=20ns DLYMX=20ns
  + )
```

**Timing model format**

```
.MODEL <timing model name> UDLY [model parameters]
```

---

### Delay line timing model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLYMN</td>
<td>Delay: min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>DLYTY</td>
<td>Delay: typical</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>DLYMX</td>
<td>Delay: max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58.
Programmable logic array

The programmable logic array is made up of a variable number of inputs, which form columns, and a variable number of outputs, which form rows. Each output (row) is driven by one logic gate. The “program” for the device determines which of the inputs (columns) are connected to each gate. All of the gates in the array are the same type (e.g., AND, OR, NAND, and NOR). Commercially available ICs (PALs, GALs, PEELs, and such) can have buffers, registers, more than one array of gates, and so on, all on the same part. These would normally be combined in a library subcircuit to make the part easier to use.

There are two ways to provide the program data for Programmable Logic Arrays. The normal way is to give the name of a JEDEC format file which contains the program data. This file would normally be produced by a PLD design package, or by using MicroSim PLSyn, which translates logic design information into a program for a specific programmable logic part. The other way to program the logic array is by including the program data, in order, on the device line (using the DATA=... construct).

If one of the PAL or GAL devices are being used in the model library, you will not need to use the Programmable Logic Array primitive directly, nor any of the model information below, since the library contains all of the appropriate modeling information. Using a PLD from the library is just like using any other logic device from the library, except that the simulator needs to know the name of the JEDEC file which contains the program for that part. A TEXT parameter name JEDEC_FILE is used to specify the file name, as shown in the following example:
This example creates a 14H4 PAL which is programmed by the JEDEC file `myprog.jed`.

**Device format**

```
U<name> <pld type> (<no. of inputs>, <no. of outputs>)
+ <digital power node> <digital ground node>
+ <input_node>* <output_node>*
+ <timing model name> <I/O model name>
+ [FILE=<(file name) text value>]
+ [DATA=<radix flag>$<program data>$]
+ [MNTYMXDL=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]
```

**Timing model format**

```
.MODEL <timing model name> UPLD [model parameters]
```

**Examples**

```
UDECODE PLANDC(3, 8) ; 3 inputs, 8 outputs
+ $G_DPWR $G_DGND ; digital power supply and ground
+ IN1 IN2 IN3 ; the inputs
+ OUT0 OUT1 OUT2 OUT3 OUT4 OUT5 OUT6 OUT7 ; the outputs
+ PLD_MDL ; the timing model name
+ IO_STD ; the I/O model name
+ DATA=B$ ; the programming data
* IN1 IN2 IN3
* TF TF TF
+ 01 01 01 ; OUT0
+ 01 01 10 ; OUT1
+ 01 10 01 ; OUT2
+ 01 10 10 ; OUT3
+ 10 01 01 ; OUT4
+ 10 01 10 ; OUT5
+ 10 10 01 ; OUT6
+ 10 10 10 $ ; OUT7

.MODEL PLD_MDL UPLD(...) ; PLD timing model definition
```

**Arguments and options**

```
<pld type>
```

One of the following:

<table>
<thead>
<tr>
<th>PLD type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLAND</td>
<td>AND array</td>
</tr>
<tr>
<td>PLANDC</td>
<td>AND array using true and complement columns for each input</td>
</tr>
<tr>
<td>PLNAND</td>
<td>NAND array</td>
</tr>
</tbody>
</table>
### PLD type | Description
--- | ---
PLNANDC | NAND array using true and complement columns for each input
PLNOR | NOR array
PLNORC | NOR array using true and complement columns for each input
PLNXOR | Exclusive NOR array
PLNXORC | Exclusive NOR array using true and complement columns for each input
PLOR | OR array
PLORC | OR array using true and complement columns for each input
PLXOR | Exclusive OR array
PLXORC | Exclusive OR array using true and complement columns for each input
<file name text value>

The name of a JEDEC format file which specifies the programming data for the array. The file name can be specified as a text constant (enclosed in double quotes ""), or as a text expression (enclosed in vertical bars "|"). If a FILE is specified, any programming data specified by a DATA section is ignored. The mapping of addresses in the JEDEC file to locations in the array is controlled by model parameters specified in the timing model.

<radix flag>

One of the following:

B - binary data follows

O - octal data follows (most significant bit has the lowest address)

X - hexadecimal data follows (most significant bit has lowest address)

<program data>

A string of data values used to program the logic array. The values start at address zero, which programs the array for the connection of the first input pin to the gate which drives the first output. A 0 (zero) specifies that the input is not connected to the gate, and a 1 specifies that the input is connected to the gate. (Initially, all inputs are not connected to any gates.) The next value programs the array for the connection of the complement of the first input to the gate which drives the first output (if this is a programmable gate having true and complement inputs) or, the second input connection to the gate which drives the first output. Each additional 1 or 0 programs the connection of the next input or its complement to the gate which drives the first output, until the connection of all inputs (and their complements) to that gate have been programmed. Data values after that, program the connection of inputs to the gate driving the second output, and so on.

The data values must be enclosed in dollar signs ($), but can be separated by spaces or continuation lines.
The example defines a 3-to-8 line decoder. The inputs are IN1 (MSB), IN2, and IN3 (LSB). If the inputs are all low, OUT0 is true. If IN1 and IN2 are low and IN3 is high, then OUT1 is true, and so on. The programming data has been typed in as an array, so that it is easier to read. The comments above the columns identify the true and false (complement) inputs, and the comments at the end of the line identify the output pin which is controlled by that gate. (Note, the simulator does not process any of these comments—they just help make the programming data readable.)

### Programmable logic array timing model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMPOFFSET</td>
<td>JEDEC file mapping: address of complement of first input and first gate program</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>INScale</td>
<td>JEDEC file mapping: amount the JEDEC file address changes for each new input pin</td>
<td>std</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true/cmp</td>
<td>2</td>
</tr>
<tr>
<td>OFFSET</td>
<td>JEDEC file mapping: address of first input and first gate program</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>OUTSCALE</td>
<td>JEDEC file mapping: amount the JEDEC file address changes for each new output pin (gate)</td>
<td>std</td>
<td>&lt;no. of inputs&gt;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>true/cmp</td>
<td>2*&lt;no. of inputs&gt;</td>
</tr>
<tr>
<td>TPHLMN</td>
<td>delay: in to out, hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLTY</td>
<td>delay: in to out, hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPHLMX</td>
<td>delay: in to out, hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHMN</td>
<td>delay: in to out, low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHTY</td>
<td>delay: in to out, low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPLHMX</td>
<td>delay: in to out, low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See [MODEL (model definition)] on page 58.
Read only memory

There are two ways to provide the program data for ROMs. The normal way is to provide the name of an Intel Hex Format file. This file is read before the simulation starts, and the ROM is programmed to contain the data in the file. The other way to program the ROM is to include the program data on the device line (with the DATA=... construct).

The example below defines a 4-bit by 4-bit to 8-bit multiplier ROM.

Device format

U<name> ROM( <no. of address pins>, <no. of output pins> )
+ <digital power node> <digital ground node>
+ <enable_node> <address node msb> ... <address node lsb>
+ <output node msb> ... <output node lsb>
+ <timing model name> <I/O model name>
+ [FILE=<file name text value>]
+ [DATA=<radix flag>$<program data>$]
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]

Timing model format

.MODEL <timing model name> UROM (<model parameters>*)
Examples

```
UMULTIPLY ROM(8, 8) ; 8 address bits, 8 outputs
+ $G_DPWR $G_DGND; digital power supply and ground
+ ENABLE ; enable node
+ AIN3 AIN2 AIN1 AIN0 ; the first 4 bits of address
+ BIN3 BIN2 BIN1 BIN0 ; the second 4 bits of address
+ OUT7 OUT6 OUT5 OUT4 OUT3 OUT2 OUT1 OUT0 ; the outputs
+ ROM_MDL ; the Timing Model name
+ IO_STD ; the I/O MODEL name
+ DATA=X$ ; the programming data
* B input value:

*  0  1  2  3  4  5  6  7  8  9ABCDEF
+ 00 00 00 00 00 00 00 00 00 00000000
  00 00 00 ; A = 0
+ 00 01 02 03 04 05 06 07 08 090A0B0C
  0D 0E 0F ; A = 1
+ 00 02 04 06 08 0A 0C 0E 10 12141618
  1A 1C 1E ; A = 2
+ 00 03 06 09 0C 0F 12 15 18 1B1E2124
  27 2A 2D ; A = 3
+ 00 04 08 0C 10 14 18 1C 20 24282C30
  34 38 3C ; A = 4
+ 00 05 0A 0F 14 19 1E 23 28 2D32373C
  41 46 4B ; A = 5
+ 00 06 0C 12 18 1E 24 2A 30 36C4248
  4E 5A 5E ; A = 6
+ 00 07 0E 15 1C 23 2A 31 38 3F4654D5
  5B 62 69 ; A = 7
+ 00 08 10 18 20 28 30 38 40 48505860
  68 70 78 ; A = 8
+ 00 09 12 1B 24 2D 36 3F 48 51A636C
  75 7E 87 ; A = 9
+ 00 0A 14 1E 28 32 3C 46 50 5A646E78
  82 8C 96 ; A = A
+ 00 0B 16 21 2C 37 42 4D 58 636E7984
  8F 9A A5 ; A = B
+ 00 0C 18 24 30 3C 48 54 60 6C788490
  9C A8 B4 ; A = C
+ 00 0D 1A 27 34 41 4E 5B 68 75828F9C
  A9 B6 C3 ; A = D
+ 00 0E 1C 2A 38 46 54 62 70 7E8C9AA8
  B6 C4 D2 ; A = E
+ 00 0F 1E 2D 3C 4B 5A 69 78 8796A5B4
  C3 D1 E1$ ; A = F
```

```
/MITMPOLY ROM_MDL UROM(...); ROM Timing Model definition
```

Arguments and options

<file name text value>

The name of an Intel Hex format file which specifies the programming data for the ROM. The file name can be specified as a text constant (enclosed in double quotes “ ”), or as a text expression (enclosed in vertical bars “|”). If a FILE is specified, any programming data specified by a DATA section is ignored.

<radix flag>

One of the following:

B - binary data follows
O - octal data follows (most significant bit has lowest address)
X - hexadecimal data follows (most significant bit has lowest address)

<program data>

The program data is a string of data values used to program the ROM. The values start at address zero, first output bit. The next bit specifies the next output bit, and so on until all of the output bits for that address have been specified. Then the output values for the next address are given, and so on.

The data values must be enclosed in dollar signs ($ $), but can be separated by spaces or continuation lines.

Read only memory timing model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPADHMN</td>
<td>delay: address to data, low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADHTY</td>
<td>delay: address to data, low to hi-Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADHMX</td>
<td>delay: address to data, low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADLMN</td>
<td>delay: address to data, hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADLTY</td>
<td>delay: address to data, hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADLMX</td>
<td>delay: address to data, hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>
### PSpice Reference Guide
Digital devices

<table>
<thead>
<tr>
<th>Model parameters(^1)</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPEDHMHN</td>
<td>delay: enable to data, hi-Z to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDHTY</td>
<td>delay: enable to data, hi-Z to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDHMX</td>
<td>delay: enable to data, hi-Z to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDLMN</td>
<td>Delay: enable to data, hi-Z to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDLTY</td>
<td>delay: enable to data, hi-Z to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDLMX</td>
<td>delay: enable to data, hi-Z to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDHZMN</td>
<td>delay: enable to data, hi to hi-Z, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDHZTY</td>
<td>delay: enable to data, hi to hi-Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDHZMX</td>
<td>delay: enable to data, hi to hi-Z, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDLZMN</td>
<td>delay: enable to data, low to hi-Z, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDLZTY</td>
<td>delay: enable to data, low to hi-Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPEDLZMX</td>
<td>delay: enable to data, low to hi-Z, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See `.MODEL (model definition)` on page 58.
Random access read-write memory

The RAM is normally initialized using unknown data at all addresses. There are two ways to provide other initialization data for RAMs. The normal way is to give the name of an Intel Hex Format file. This file is read before the simulation starts, and the RAM is initialized to match the data in the file. The other way to initialize the RAM is to include the initialization data on the device line (using the DATA=... construct).

**Device format**

U<name> RAM(<no. of address bits>, <no. of output bits>)
+ <digital power node> <digital ground node>
+ <read enable node> <write enable node>
+ <address msb node>...<address lsb node>
+ <write-data msb node>...<write-data lsb node>
+ <read-data msb node>...<read-data lsb node>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]
+ [FILE=<file name text value>]
+ [DATA=<radix flag>$<initialization data>$]

**Timing model format**

.MODEL <timing model name> URAM (<model parameters>*)

**Arguments and options**
<file name text value>

The name of an Intel Hex format file which specifies the initialization data for the RAM. The file name can be specified as a text constant (enclosed in double quotes ""), or as a text expression (enclosed in vertical bars | |). If a FILE is specified, any initialization data specified by a DATA section is ignored.

<radix flag>

One of the following:

B - binary data follows

O - octal data follows (most significant bit has lowest address)

X - hexadecimal data follows (most significant bit has lowest address)

<initialization data>

A string of data values used to initialize the RAM. The values start at address zero, first output bit. The next bit specifies the next output bit, and so on until all of the output bits for that address have been specified. Then the output values for the next address are given, and so on.

The data values must be enclosed in dollar signs ($ $), but can be separated by spaces or continuation lines.

The initialization of a RAM using the DATA=... construct is the same as the programming of a ROM. See Read only memory on page 384 on the ROM primitive for an example.
Comments

The RAM has separate read and write sections, using separate data and enable pins, and shared address pins. To write to the RAM, the address and write data signals must be stable for the appropriate setup times, then write enable is raised. It must stay high for at least the minimum time, then fall. Address and data must remain stable while write enable is high, and for the hold time after it falls. Write enable must remain low for at least the minimum time before changing.

To read from the RAM, raise read enable, and the outputs change from Z (high impedance) to the appropriate value after a delay. The address can change while read enable is high, and if it does, the new data is available at the outputs after the delay.

Nothing prevents both the read and write enable from being true at the same time, although most real devices would not allow this. The new value from the write is sent to the read data outputs on the falling edge of write enable.

Random access memory timing model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPADHMN</td>
<td>delay: address to read data, low to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADHTY</td>
<td>delay: address to read data, low to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADHMX</td>
<td>delay: address to read data, low to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADLMN</td>
<td>delay: address to read data, hi to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADLTY</td>
<td>delay: address to read data, hi to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPADLMX</td>
<td>delay: address to read data, hi to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDHMN</td>
<td>delay: read enable to read data, hi-Z to hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDHTY</td>
<td>delay: read enable to read data, hi-Z to hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDHMX</td>
<td>delay: read enable to read data, hi-Z to hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDLMN</td>
<td>delay: read enable to read data, hi-Z to low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDLTY</td>
<td>delay: read enable to read data, hi-Z to low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDLMX</td>
<td>delay: read enable to read data, hi-Z to low, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>Model parameters</td>
<td>Description</td>
<td>Units</td>
<td>Default</td>
</tr>
<tr>
<td>------------------</td>
<td>-------------</td>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>TPERDHZMN</td>
<td>delay: read enable to read data, hi to hi-Z, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDHZTY</td>
<td>delay: read enable to read data, hi to hi-Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDHZMX</td>
<td>delay: read enable to read data, hi to hi-Z, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THAEWTY</td>
<td>min hold time: write enable fall to address change, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THAEWMX</td>
<td>min hold time: write enable fall to address change, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THDEWMN</td>
<td>min hold time: write enable fall to data change, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THDEWTY</td>
<td>min hold time: write enable fall to data change, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THDEWMX</td>
<td>min hold time: write enable fall to data change, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>THAEWMN</td>
<td>min hold time: write enable fall to address change, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDLZMN</td>
<td>delay: read enable to read data, low to hi-Z, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDLZTY</td>
<td>delay: read enable to read data, low to hi-Z, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPERDLMX</td>
<td>delay: read enable to read data, low to hi-Z, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDEWMN</td>
<td>min setup time: data to write enable rise, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDEWTY</td>
<td>min setup time: data to write enable rise, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUDEWMX</td>
<td>min setup time: data to write enable rise, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUAEWMN</td>
<td>min setup time: address to write enable rise, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUAEWTY</td>
<td>min setup time: address to write enable rise, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSUAEWMX</td>
<td>min setup time: address to write enable rise, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWEWHMN</td>
<td>min width: enable write hi, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWEWHTY</td>
<td>min width: enable write hi, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWEWHMX</td>
<td>min width: enable write hi, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWEWLMN</td>
<td>min width: enable write low, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TWEWLTY</td>
<td>min width: enable write low, typ</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>
### Model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TWEWLMX</td>
<td>min width: enable write low, max</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

1. See [MODEL (model definition)] on page 58.
Multi-bit A/D and D/A converter

The simulator provides two primitives to model analog-to-digital converters and digital-to-analog converters: the ADC and the DAC. These two primitives simplify the modeling of these complex mixed-signal devices.

Multi-bit analog-to-digital converter

**Device format**

U<name> ADC(<number of bits>)
+ <digital power node> <digital ground node>
+ <in node> <ref node> <gnd node> <convert node>
+ <status node> <over-range node>
+ <output msb node> ... <output lsb node>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]

**Timing model format**

.Model <timing model name> UADC [model parameters]

**Examples**

U5 ADC(4) $G_DPWR $G_DGND ; 4-bit ADC
+ Sig Ref 0 Conv Stat OvrRng Out3 Out2 Out1 Out0
+ ADCModel IO_STD

.Model ADCModel UADC(...) ; Timing Model

---

**Multi-bit A/D converter timing model parameters**

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPCSMN</td>
<td>propagation delay: rising edge of convert to rising edge of status, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCSTY</td>
<td>propagation delay: rising edge of convert to rising edge of status, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPCSMX</td>
<td>propagation delay: rising edge of convert to rising edge of status, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDSMN</td>
<td>propagation delay: data valid to falling edge of status, min</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDSTY</td>
<td>propagation delay: data valid to falling edge of status, typ</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPDSMX</td>
<td>propagation delay: data valid to falling edge of status, max</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TPSDMN</td>
<td>propagation delay: rising edge of status to data valid, min</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>
ADC primitive device timing

DATA refers to both the data and over-range signals. The Convert pulse can be any width, including zero. If the propagation delay between the rising edge of the Convert signal and the Status signal (tpsd) is zero, the data and over-range do not go to unknown but directly to the new value. There is a resistive load from <ref node> to <gnd node>, and from <in node> to <gnd node>, of 1/GMIN.

The voltage at <in node> and <ref node> with respect to <gnd node> is sampled starting at the rising edge of the Convert signal, and ending when the Status signal becomes high. This gives a sample aperture time of tpcs plus any rising time for Convert. If, during the sample aperture, the output calculated having the minimum <ref node> voltage and maximum <in node> voltage is different from the output calculated having the maximum <ref node> voltage and minimum <in node> voltage, the appropriate output bits are set to the unknown state and a warning message is printed in the output file.

The output is the binary value of the nearest integer to

\[
\frac{V(in, gnd)}{V(ref, gnd)} \cdot 2^{nbits}
\]
If this value is greater than $2^{n\text{bits}} - 1$, then all data bits are 1, and over-range is 1. If this value is less than zero, then all data bits are zero, and over-range is 1.

**Multi-bit digital-to-analog converter**

**Device format**

```plaintext
U<name> DAC(<number of bits>)
+ <digital power node> <digital ground node>
+ <out node> <ref node> <gnd node>
+ <input msb node> ... <input lsb node>
+ <timing model name> <I/O model name>
+ [MNTYMXDLY=<delay select value>]
+ [IO_LEVEL=<interface subckt select value>]
```

**Timing model format**

```plaintext
.MODEL <timing model name> UDAC [model parameters]
```

**Examples**

```plaintext
U7 DAC(4) $G_DPWR $G_DGND ; 4-bit DAC
+ Sig Ref 0 In3 In2 In1 In0
+ DACModel IO_STD

.MODEL DACModel UDAC(...) ; Timing model
```

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSWMN</td>
<td>Switching time: change in data to analog out stable, min</td>
<td>sec</td>
<td>10ns</td>
</tr>
<tr>
<td>TSWTY</td>
<td>Switching time: change in data to analog out stable, typ</td>
<td>sec</td>
<td>10ns</td>
</tr>
<tr>
<td>TSWMX</td>
<td>Switching time: change in data to analog out stable, max</td>
<td>sec</td>
<td>10ns</td>
</tr>
</tbody>
</table>

1. See .MODEL (model definition) on page 58.

**DAC primitive device timing**

The DAC is a zero impedance voltage source from `<out node>` to `<gnd node>`. The voltage is

$$V(\text{ref, gnd}) \cdot \frac{(\text{binary value of inputs})}{2^{n\text{bits}}}$$
There is a resistance of $1/GMIN$ between $<ref \ node>$ and $<gnd \ node>$. If any inputs are unknown ($X$), the output voltage is halfway between the output voltage if all the $X$ bits were 1 and the output voltage if all the $X$ bits were 0. When an input bit changes, the output voltage changes linearly to the new value during the switching time.

![Diagram showing voltage and data transition](image-url)
Behavioral primitives

The simulator offers three primitives to aid in the modeling of complex digital devices: the Logic Expression, Pin-to-Pin Delay, and Constraint Checker primitives. These devices are distinct from other primitives in that they allow data-sheet descriptions to be specified more directly, allowing a one-to-one correspondence using the function diagrams and timing specifications.

The Logic Expression primitive, LOGICEXP, uses free-format logic expressions to describe the functional behavior device.

The Pin-To-Pin Delay primitive, PINDELAY, describes propagation delays using sets of rules based on the activity on the device inputs.

The Constraint Checker primitive, CONSTRAINT allows a listing of timing rules such as setup/hold times, and minimum pulse widths. When a violation occurs, the simulator issues a message indicating the time of the violation and its cause.

Logic expression

The LOGICEXP primitive allows combinational logic to be expressed in an equation-like style, using standard logic operators, node names, and temporary variables.

Device format

U<name> LOGICEXP ( <no. of inputs>, <no. of outputs> )
+ <digital power node> <digital ground node>
+ <input node 1> ... <input node n>
+ <output node 1> ... <output node n>
+ <timing model name>
+ <I/O model name>
+ [IO_LEVEL = <value>]
+ [MNTYMXDLY = <value>]
+ LOGIC:
+ [logic assignment]*

Timing model format

.MODEL <timing model name> UGATE [model parameters]

Arguments and options

LOGIC:

Marks the beginning of a sequence of one or more <logic assignments>. A <logic assignment> can have one of the two following forms:

<output node> = { <logic expression> }
<temporary value> = { <logic expression> }
<output node>

One of the output node names as it appears in the interface list. Assignments to an <output node> causes the result of the <logic expression> to be scheduled on that output pin. Each <output node> must have exactly one assignment.

<temporary value>

Any target of an assignment which is not specified as one of the nodes attached to the device defines a temporary variable. Once assigned, <temporary values> can be used inside subsequent <logic expressions>. They are provided to reduce the complexity and improve the readability of the model. The rules for node names apply to <temporary value> names.

<logic expression>

A C-like, infix-notation expression that returns one of the five digital logic levels. Like all other expressions, <logic expressions> must be surrounded by curly braces { }. They can span one or more lines using the + continuation character in the first column position.

The logic operators are listed below from highest-to-lowest precedence.

Logic Expression Operators

~ unary not
& and
^ exclusive or
| or

The allowed operands are:

- <input nodes>
- Previously assigned <temporary values>
- Previously assigned <output nodes>
- <logic constants>: 0, 1, X, R, F
As in other expressions, parentheses ( ) can be used to group subexpressions. Note that these logic operators can also be used in Probe trace definitions.

**Comments**

The LOGICEXP primitive uses the same timing model as the standard gate primitives, UGATE.

See “Standard gate timing model parameters” on page 359 for the list of UGATE model parameters.

**Simulation behavior**

When a LOGICEXP primitive is evaluated during a transient analysis, the assignment statements using in it are evaluated in the order they were specified in the netlist. The logic expressions are evaluated using no delay. When the result is assigned to an output node, it is scheduled on that output pin using the appropriate delay specified in the timing model.

Internal feedback loops are not allowed in expressions. That is, an expression cannot reference a value which has yet to be defined. However, external feedback is allowed if the output node also appears on the list of input nodes.

This example models the functionality of the 74181 Arithmetic/Logic Unit. The logic for the entire part is contained in just one primitive. Timing would be handled by the PINDLY and CONSTRAINT primitives. Refer to any major device manufacturer’s data book for a detailed description of the operation of the 74181.

```
U74181 LOGICEXP( 14, 8 ) DPWR DGND
+ A0BAR A1BAR A2BAR A3BAR B0BAR B1BAR B2BAR B3BAR S0 S1 S2 S3 M CN
+ LF0BAR LF1BAR LF2BAR LF3BAR LAEQUALB LPBAR LGBAR LCN+4
+ D0_GATE IO_STD
+
+ LOGIC:
+ *
+ Intermediate terms:
+ *
+ I31 = { ~((B3BAR & S3 & A3BAR) | (A3BAR & S2 & ~B3BAR)) }
+ I32 = { ~((~B3BAR & S1) | (S0 & B3BAR) | A3BAR ) }
+ I21 = { ~((B2BAR & S3 & A2BAR) | (A2BAR & S2 & ~B2BAR)) }
+ I22 = { ~((~B2BAR & S1) | (S0 & B2BAR) | A2BAR ) }
+ I11 = { ~((B1BAR & S3 & A1BAR) | (A1BAR & S2 & ~B1BAR)) }
+ I12 = { ~((~B1BAR & S1) | (S0 & B1BAR) | A1BAR ) }
+ I01 = { ~((B0BAR & S3 & A0BAR) | (A0BAR & S2 & ~B0BAR)) }
+ I02 = { ~((~B0BAR & S1) | (S0 & B0BAR) | A0BAR ) }
+ MBAR = { ~M }
+ P = { I31 & I21 & I11 & I01 }
+ *
```
* Output Assignments
*
+ LF3BAR = {(I31 & ~I32) ^
+ ~( (I21 & I11 & I01 & Cn & MBAR) | (I21 & I11 & I02 & MBAR) |
+ (I21 & I12 & MBAR) | (I22 & MBAR) )}
+
+ LF2BAR = {(I21 & ~I22) ^
+ ~( (I11 & I01 & Cn & MBAR) | (I11 & I02 & MBAR) |
+ (I12 & MBAR) )}
+
+ LF1BAR = {(I11 & ~I12) ^ ~( (Cn & I01 & MBAR) |
+ (I02 & MBAR) )}
+
+ LF0BAR = { (I01 & ~I02) ^ ~(MBAR & Cn) }
+
+ LGBAR = { ~( I32 | (I31 & I22) | (I31 & I21 & I12) |
+ (I31 & I22 & I11 & I02) ) }
+
+ LCN+4 = { ~LGBAR | (P & Cn) }
+
+ LPBAR = { ~P }
+
+ LAEQUALB = { LF3BAR & LF2BAR & LF1BAR & LF0BAR }

**Pin-to-pin delay**

The pin-to-pin (PINDLY) primitive is a general mechanism that allows the modeling of complex device timing. It can be thought of as a set of delay-lines (paths) and rules describing how to associate specific amounts of delay using each path.

A PINDLY primitive is used in the output path of a device model, typically at the output pins of a subcircuit definition. A single PINDLY primitive can model the timing and output characteristics of an entire part, including tristate behavior.

PINDLY primitives are expressed and evaluated in a manner similar to the LOGICEXP primitive, except in this case a delay expression is assigned to each output. Whenever an output path undergoes a transition, its delay expression is evaluated to determine the propagation delay which is to be applied to that change.

A delay expression can contain one or more rules that determine which activity on the part's inputs is responsible for the output change, for example, “is the output changing because the clock changed or the data changed?” This allows device models to be derived directly from data sheets, which typically specify propagation delays based on which input is changing. The PINDLY primitive uses its reference inputs to determine the logic state and recent transitions on nodes which are not in the output path.

Pin-to-pin delay modeling is much simpler compared to earlier methods, in which input-to-output delays had to be distributed among the low-level primitives used to model the device. The latter method can require a great deal of trial and error because manufacturer’s data sheets do not provide a one-to-one association between the logic diagram and the timing specifications.
PINDLY primitives can also contain constraints such as setup/hold, width, and frequency specifications, like those supported by the CONSTRAINT primitive. When used in the PINDLY primitive, these constraints allow the simulator to propagate hazard conditions and report violations in subsequent logic.

**Device format**

\[ U<\text{name}> \text{ PINDLY} \{ \text{<no. of paths>, <no. of enables>, <no. of refs>} \} \]
+ \text{<digital-power-node>} \text{<digital-ground-node>}
+ \text{<input node 1> ... <input node n>}
+ [\text{<enable node 1> ... <enable node n>}] 
+ [\text{<reference node 1> ... <reference node n>}] 
+ \text{<output node 1> ... <output node n>}
+ \text{<I/O model name>}
+ \text{<MNTYMXDL} = \text{<delay select value>}
+ \text{<IO\_LEVEL} = \text{<interface subckt select value>}
+ \text{<BOOLEAN>}
+ \text{<boolean assignment>*}
+ \text{PINDLY:}
+ \text{<delay assignment>*}
+ \text{<TRISTATE>}
+ \text{<ENABLE LO | HI <enable node>}
+ \text{<delay assignment>*}
+ \text{<SETUP\_HOLD: <setup\_hold\_specification>}}
+ \text{<WIDTH: <width\_specification>}}
+ \text{<FREQ: <frequency\_specification>}}
+ \text{<GENERAL: <general\_specification>}}

**Examples**

\[ U1 \text{ PINDLY}(4,0,3) \text{ $G\_DPWR} \text{ $G\_DGND} \]
+ \text{IN1 IN2 IN3 IN4}
+ \text{REF1 REF2 REF3}
+ \text{OUT1 OUT2 OUT3 OUT4}
+ \text{IO\_MODEL D0\_GATE}
+ \text{PINDLY:}
+ \text{...}
Arguments and options

<no. of paths>

Specifies the number of input-to-output paths represented by the device; the number of inputs must be equal to the number of outputs. A path is defined as an input-to-output association, having the appropriate delay rules started according to the described conditions.

<no. of enables>

Specifies the number of tristate enable nodes used by the primitive. Enable nodes are used in TRISTATE sections. <no. of enables> can be zero.

<no. of refs>

Specifies the number of reference nodes used by the primitive. Reference nodes are used within delay expressions to get state information about signals which are not in the input-to-output paths. <no. of refs> can be zero.

Comments

The example depicts the relationship and purpose of the different pins on the PINDLY primitive.

The PINDLY primitive can be viewed as four buffers, IN1 to OUT1 through IN4 to OUT4, and three reference nodes which are used by the output delay rules. The figure shows how the reference nodes can be used in one or more set of delay rules. In this case, REF1 and REF2 are used by the delay rules for OUT2, and REF3 is used by the delay rules for OUT1 and OUT4. The figure also shows that OUT2 and OUT3 can share the same delay rules. The remainder of the format description describes how to create delay rules.

BOOLEAN:

Marks the beginning of a section of one or more <boolean assignments>, which define temporary variables that can be used in subsequent <delay expressions>. BOOLEAN sections can appear in any order within the PINDLY primitive. A <boolean assignment> has the following form:

<boolean variable> = { <boolean-expression> }

<boolean variable> can be any name which follows the node name rules.
<boolean expression> is a C-like, infix-notation expression which returns the boolean value TRUE or FALSE. Like all other expressions, <boolean expressions> must be surrounded by curly braces {...}. They can span one or more lines by using the + continuation character in the first column position. The boolean operators are listed below from highest-to-lowest precedence:

- ~ unary not
- == equality
- != inequality
- & and
- ^ exclusive or
- | or

All boolean operators take the following boolean values as operands:
- Previously assigned <boolean variables>
- Reference functions (defined below)
- Transition functions (defined below)
- <boolean constants>: TRUE, FALSE

In addition, the == and != operators take logic values, such as <input nodes> and <logic constants>. This allows for a check of the values on nodes; for example, CLEAR == 1 returns TRUE if the current level on the node CLEAR is a logic one and FALSE otherwise.

**Reference functions**

Reference functions are used to detect changes (transitions) on <reference nodes> or <input nodes>. All reference functions return boolean values, and therefore can be used within any <boolean expression>. Following is the list of available reference functions and their arguments:

```plaintext
CHANGED <node>, <delta time> )
CHANGED_LH <node>, <delta time> )
CHANGED_HL <node>, <delta time> )
```

The CHANGED function returns TRUE if the specified <node> has undergone any state transition within the past <delta time>, prior to the current simulation time; otherwise it returns FALSE.

Similarly, CHANGED_LH returns TRUE if <node> has specifically undergone a low-to-high transition within the past <delta time>; FALSE otherwise. Note that CHANGED_LH only looks at the most recent (or current) transition. It cannot, for example, determine if 0→1 occurred two transitions ago.

Finally, CHANGED_HL is similar to CHANGED_LH, but checks for high-to-low transitions.
If a `<delta time>` is specified zero, the reference functions return TRUE if the node has changed at the current simulation time. This allows all of the functionality of a device to be modeled in zero delay so that the total delay through the device can be described using the delay expressions.

**Transition functions**

Transition functions are used to determine the state change occurring on the changing output, that is, the `<output node>` for which the `<delay expression>` is being evaluated. Like reference functions, transition functions return boolean values. However, they differ from reference functions in that transition functions take no arguments, since they implicitly refer to the changing output at the current time. The transition functions are of the general form:

```
TRN_pn
```

where p is the previous state value and n is the new state value. State values are taken from the set `{ L H Z $ }`. Where appropriate, the $ can be used to signify don’t care, e.g., a TRN_H$ matches a transition from H to ANY state. Rising states automatically map to High, and Falling states automatically map to Low.

As a term in any boolean expression, for example, TRN_LH takes on a TRUE value if the changing output is propagating a change from zero to one.

Following is the complete set of transition functions.

```
TRN_LH TRN_LZ TRN_L$ TRN_HL TRN_HZ TRN_H$ TRN_ZL TRN_ZH TRN_Z$ TRN_$L TRN _$H TRN_$Z
```

**Note:** The TRN_pZ and TRN_Zn functions return true only if it is used within a TRISTATE section, described below. Although open-collector outputs also transition to a high-impedance Z (instead of H), most data books describe propagation times on open-collector outputs as TPLH or TPHL. Therefore, open-collector output devices should use TRN_LH and TRN_HL, and tristate output devices should use TRN_LZ, TRN_HZ, TRN_ZL, and TRN_ZH.

PINDL Y: marks the beginning of a section of one or more `<delay assignments>`, which are used to associate propagation delays using the PINDLY primitive’s outputs. `<delay assignments>` are of the form:

```
<output node>* = { <delay expression> }
```

`<output node>` is one of the output node names as it appears in the interface list. Each `<output node>` must have exactly one assignment. Several outputs can share the same delay rules by listing them (separated by spaces or commas) on the left-hand side of the `<delay expression>`.

 `<delay expression>` is an expression which, when evaluated, returns a triplet (min, typ, max) of delay values. Like all other expressions, `<delay expressions>` must be
surrounded by curly braces {...}. They can span one or more lines by using the +222222222222 continuation character in the first column position.

The simplest <delay expression> is a single <delay value>, defined as:

```
DELAY(<min>, <typ>, <max>)
```

where <min>, <typ>, and <max> are floating point constants or expressions (involving parameters), expressed in seconds. To specify unknown values, use -1. For example, DELAY(20ns,-1,35ns) specifies a minimum time of 20ns, a default (program-computed) value for typical, and a maximum of 35ns. See Treatment of unspecified propagation delays on page 353 for more information on default delays.

The delay assignment below specifies the propagation delays through output Y to be:

\[
\begin{align*}
\text{min} &= 2\text{ns} \\
\text{typ} &= 5\text{ns} \\
\text{max} &= 9\text{ns}
\end{align*}
\]

\[+ \quad \text{PINDLY:} \quad Y = \{ \text{DELAY(2ns, 5ns, 9ns)} \}\]

To define more complex, rule-based <delay expressions>, use the CASE function, which has the form:

```
CASE(
  <boolean expression>, <delay expression>, ; Rule 1
  <boolean expression>, <delay expression>, ; Rule 2
  ... ; ...
  <delay expression> ; Default delay
)
```

The arguments to the CASE function are pairs of <boolean expressions> and <delay expressions>, followed by a final default <delay expression>. <boolean expressions> (described above) can contain <boolean values>, reference functions, and transition functions.

When the CASE function is evaluated, each <boolean expression> is evaluated in order of appearance until one produces a TRUE result. When this occurs, the <delay expression> is paired with the result of the CASE function, and the evaluation of the CASE is ended. If none of the <boolean expressions> return a TRUE result, the value of the final <delay expression> is used. Because it is possible for all <boolean expressions> to evaluate FALSE, the default delay value must be supplied. Note that each argument to the CASE function must be separated by commas.

```
... + BOOLEAN:
+  CLOCK = \{ \text{CHANGED_LH( CLK, 0 )} \}
+  PINDLY:
+    QA QB QC QD = {
+      CASE ( 
+        CLOCK & TRN_LH, \text{DELAY(-1,13ns,24ns)},
+        CLOCK & TRN_HL, \text{DELAY(-1,18ns,27ns)},
+      
```
This example describes the delays through a four-bit counter. It shows how rules can be defined to precisely isolate the cause of the output change. In this example, the boolean variable CLOCK is being defined. It is TRUE whenever the reference input CLK changes from low-to-high at the current simulation time. This is only true if the device functionality is modeled in zero delay.

The four outputs QA through QD all share the same delay expression. The CASE is used to specify different delays when the device is counting or clearing. The first two rules define delays when the device is counting (CLK changing low-to-high); the first when the output (QA through QD) is going from low-to-high, the second from high-to-low.

The third rule simply uses the CHANGED_HL function directly to determine whether CLRBAR is changing, and in this case the specification applies to any change (low-to-high or high-to-low) on the output. The default delay applies to all other output transitions which are not covered by the first three rules.

**TRISTATE:** marks the beginning of a sequence of one or more <delay assignments>. The TRISTATE section differs from the PINDLY section in that the outputs are controlled by the specified enable node.

Immediately following the TRISTATE keyword, an enable node must be specified using its polarity and the ENABLE keyword:

```
ENABLE HI <enable node>; Specifies active HI enable
ENABLE LO <enable node>; Specifies active LO enable
```

The specified <enable node> applies to all <output node> assignments in the current section.

**Note:** Note that <delay expressions> within a TRISTATE section can contain the transition functions pertaining to the Z state, for example TRN_ZL and TRN_HZ.

The following example demonstrates how an enable node can be used to control more than one output. It also shows that some device outputs can use the standard output (PINDLY) while others use the tristate output. (Delay values have been omitted.)
1. Each CONSTRAINT clause operates independently of all others within a device.
2. By default, for violations involving <input node>, the message tag propagates to the <output node> having positional correspondence.

3. By default, for violations involving <reference node>, the message tag propagates to ALL <output node>s.

4. The default behavior can be overridden by use of one of the following statements, which can appear anywhere within any constraint clause proper:

   AFFECTS (#OUTPUTS) = <output node> ( ... )
   AFFECTS_ALL

5. AFFECTS_NONE is always the default for the GENERAL constraint.

   SETUP-HOLD:    Marks the beginning of a constraint specification.
   WIDTH:        These constructs have the same syntax as those used in the CONSTRAINT primitive (see General notes on page 415).
   FREQ:
   GENERAL:

       When a PINDLY primitive is used, the constraint specifications allow the simulator to report timing violations and track the effects of the violations in downstream logic. This allows persistent hazards to be reported. This differs from the CONSTRAINT primitive, which only reports timing violations.

**PINDLY primitive simulation behavior**

A PINDLY primitive is evaluated whenever any of its <input nodes> or <enable nodes> change. The <input node> is positionally associated using its corresponding <output node>. The BOOLEAN statements up to the output assignment are evaluated first, then the appropriate PINDLY or TRISTATE <delay expression> which has been assigned to the changing <output node> is evaluated. The changing input’s state is then applied to the output, using its delay value.

The following PINDLY primitive models the timing behavior of a 74LS160A counter. This example is derived directly from the device model in the model library.

```
ULS160ADLY PINDLY(5,0,4) DPWR DGND
+ RCO QA QB QC QD ; Inputs
+ CLK LOADBAR ENT CLRBAR ; Reference nodes
+ RCO_0 QA_0 QB_0 QC_0 QD_0; Outputs
+ IO_LS MNTYMXDLY={MNTYMXDLY} IO_LEVEL={IO_LEVEL}
+
+ BOOLEAN:
+   CLOCK = { CHANGED_LH(CLK,0) }
+   CNTENT = { CHANGED(ENT,0) }
```
Constraint checker

The CONSTRAINT primitive provides a general constraint checking mechanism to the digital device modeler. It performs setup and hold time checks, pulse width checks, frequency checks, and includes a general mechanism to allow user-defined conditions to be reported.

The CONSTRAINT primitive only reports timing violations. It does not affect propagated or stored logic state or propagation delays.

Timing specifications are usually given at the device (i.e., package pin) level. Thus, the inputs to the constraint description typically are those of the subcircuit description of the device, after any necessary buffering. CONSTRAINT devices can be used in conjunction with any combination of digital primitives, including gates, logic expressions, and pin-to-pin delay primitives.

Device format

```
U<name> CONSTRAINT { <no. of inputs> } 
+ <digital power node> <digital ground node> 
+ <input node 1> ... <input node n> 
+ <I/O model name> 
+ [ IO_LEVEL = <interface subckt select value> ] 
+ [ BOOLEAN: <boolean assignment>* ] ... 
+ [ SETUP_HOLD: <setup_hold specification> ] ... 
+ [ WIDTH: <width specification> ] ... 
+ [ FREQ: <frequency specification> ] ... 
+ [ GENERAL: <general specification> ] ...
```
Arguments and Options

BOOLEAN

marks the beginning of a section containing one or more
<boolean assignments>, of the form:

<boolean variable> = { <boolean expression> }

BOOLEAN sections can appear in any order within the
CONSTRAINT primitive.

The syntax of the <boolean expression> is the same
as that defined in the PINDLY primitive reference, having the
exception that transition functions have no meaning within
the CONSTRAINT primitive.

SETUP_HOLD:

Marks the beginning of a setup/hold constraint specification,
which has the following format:

+ SETUP_HOLD:
+   CLOCK <assertion edge> = <input node>
+   DATA ( <no. of data inputs> ) = <input node j> ...
+   <input node k>
+   [ SETUPTIME = <time value> ]
+   [ HOLDTIME = <time value> ]
+   [ RELEASETIME = <time value> ]
+   [ WHEN {<boolean expression>} ]
+   [ MESSAGE = "<additional message text>" ]
+   [ ERRORLIMIT = <value> ]
+   [ AFFECTS_ALL | AFFECTS_NONE |
+     AFFECTS (#OUTPUTS) = <output-node-list> ]

Note: One or more sections can be specified in any order. Note
that AFFECTS clauses are only allowed in PINDLY primitives.

CLOCK defines the node to be used as the reference for setup/
hold/release specification. <assertion edge> is one of LH
or HL, and specifies which edge of the CLOCK node the setup/
hold time is measured against. The CLOCK node must be
specified.

DATA defines one or more nodes to be the nodes whose setup/
hold time is being measured. At least one DATA node must be
specified.
SETUPTIME defines the minimum time that all DATA nodes must be stable prior to the <assertion edge> of the clock. The <time value> must be a nonnegative constant or expression, expressed in seconds. Some devices have different setup time requirements which depend on whether the data is a low or a high at the time of the clock change. In this case, one or both of the following can be used:

\[
\text{SETUPTIME}_\text{LO} = \text{<time value>}
\]
\[
\text{SETUPTIME}_\text{HI} = \text{<time value>}
\]

instead of SETUPTIME, which defines both low- and high-level specifications. If one or both SETUPTIME_xx specifications is zero, the simulator does not perform a setup check for that data level.

HOLDTIME defines the minimum time that all DATA nodes must be stable after the <assertion edge> of the clock. The <time value> must be a nonnegative constant or expression, expressed in seconds. Some devices have different hold time requirements which depend on whether the data is a low or a high at the time of the clock change. In this case, one or both of the following can be used:

\[
\text{HOLDTIME}_\text{LO} = \text{<time value>}
\]
\[
\text{HOLDTIME}_\text{HI} = \text{<time value>}
\]

instead of HOLDTIME, which defines both low- and high-level specifications. If one or both HOLDTIME_xx specifications is zero, the simulator does not perform a hold check for that data level.

RELEASETIME specifications cause the simulator to perform a special-purpose setup check. In a data sheet, release time (also called recovery time) specifications refer to the minimum time a signal (such as CLEAR) can go inactive before the active CLOCK edge. In other words, release times refer to the position of a specific data edge in relation to the clock edge. For this reason, one or both of the following can be used:

\[
\text{RELEASETIME}_\text{LH} = \text{<time value>}
\]
\[
\text{RELEASETIME}_\text{HL} = \text{<time value>}
\]

instead of RELEASETIME, which defines both LH- and HL-edge specifications. The <time value> must be a nonnegative constant or expression, expressed in seconds.
The difference between the release-time checker and the setup-time checker is that simultaneous CLOCK/DATA changes are never allowed in the release-time check. That is, a nonzero hold time is assumed, even though the HOLDTIME is not specified. This feature allows the data sheet values to be specified for release-times directly in a model. For this reason, release times are usually given alone, and not in conjunction with SETUPTIME or HOLDTIME specifications.

**Simulation behavior: CLOCK**

The sequence of setup/hold/release checks begins when the CLOCK node undergoes the specified LH or HL transition. At that time, the WHEN expression is evaluated. If the result is TRUE, all checks using nonzero specifications are performed for during this clock cycle. If the result is FALSE, then no setup, hold, or release checks are performed. The WHEN expression is used in device models to block the reporting of violations when the device is not listening to the DATA inputs, such as during a clearing function.

The simulator performs setup-time checks when the CLOCK node undergoes an `<assertion edge>`. If the HOLDTIME specification is zero, simultaneous CLOCK/DATA transitions are allowed, however the previous value of DATA is still checked for setup-time. If the HOLDTIME is not zero, simultaneous CLOCK/DATA transitions are reported as a HOLDTIME violation.

The simulator performs hold-time checks on any DATA node that changes after the `<assertion edge>` on the CLOCK node. If the SETUPTIME is zero, simultaneous CLOCK/DATA changes are allowed, and the next transition on DATA which occurs before the non-asserting clock edge is checked for a hold-time violation.

The simulator performs release-time checks when the CLOCK node undergoes an `<assertion edge>`. Simultaneous CLOCK/DATA transitions are not allowed, and is flagged as a violation.

If either the CLOCK or DATA node is unknown (X) at the time of a check, no violation is reported for that node. This reduces the number of unnecessary warning messages: an X being clocked into a device is usually a symptom of another problem which has already been reported.

The sequence ends when the CLOCK node undergoes the other (non-asserting) edge. At this time, any violations which occurred during that clock cycle are reported. (This makes it possible for violations to appear out of time-order in the .out file.)

WIDTH: Marks the beginning of a minimum pulse-width constraint specification, which has the following format:
+ WIDTH:
+  NODE = <input node>
+  [ MIN_HI = <time value> ]
+  [ MIN_LO = <time value> ]
+  [ WHEN {<boolean expression>} ]
+  [ MESSAGE = "<additional message text>" ]
+  [ ERRORLIMIT = <value> ]
+  [ AFFECTS_ALL | AFFECTS_NONE |
+    AFFECTS (#OUTPUTS) = <output-node-list> ]

**Note:** One or more sections can be specified in any order. Note that AFFECTS clauses are only allowed in the PINDLY primitive.

**NODE** defines the input node whose pulse width is to be checked.

**MIN_HI** specifies the minimum time that the <input node> can remain at a high (1) logic level. The <time value> must be a nonnegative constant or expression, expressed in seconds. If not specified, MIN_HI defaults to 0, meaning that any width HI pulse is allowed.

**MIN_LO** likewise specifies the minimum time that the <input node> can remain at a low (0) logic level. The <time value> must be a nonnegative constant or expression, expressed in seconds. If not specified, MIN_LO defaults to 0, meaning that any width LO pulse is allowed.

At least one instance of MIN_HI or MIN_LO must appear within a WIDTH specification.

**FREQ:** marks the beginning of a frequency constraint specification, which has the following format:

+ FREQ:
+  NODE = <input node>
+  [ MINFREQ = <frequency value> ]
+  [ MAXFREQ = <frequency value>]
+  [ WHEN { <boolean expression> }]
+  [ MESSAGE "<additional message text>" ]
+  [ ERRORLIMIT = <value> ]
+  [ AFFECTS_ALL | AFFECTS_NONE |
+    AFFECTS (#OUTPUTS) = <output-node-list> ]

**Note:** One or more sections can be specified in any order. Note that AFFECTS clauses are only allowed in the PINDLY primitive.

**NODE** defines the input node whose frequency is to be checked.

**MINFREQ** specifies the minimum frequency allowed on <input node>. The <frequency value> must be a nonnegative floating point constant or expression, expressed in hertz.

**MAXFREQ** specifies the maximum frequency allowed on <input node>. The <frequency value> must be a nonnegative floating point constant or expression, expressed in hertz.
At least one of MINFREQ or MAXFREQ must be specified within a FREQ specification.

**Simulation Behavior: FREQ**

When performing a MINFREQ check, the simulator reports a violation whenever the duration of a period on the <input node> is greater than $1/<frequency value>$. Likewise, when performing a MAXFREQ check, it reports a violation whenever any period is less than $1/<frequency value>$. To avoid large numbers of violations, the simulator does not report subsequent violations until after a valid cycle occurs.

Note that the use of maximum FREQ specifications provides a slightly different functionality from that achieved by use of minimum pulse-width checks: in the FREQ specification case, the duty-cycle characteristic of the signal is not measured or constrained in any way, whereas the pulse-width check effectively defines the allowable duty-cycle.

Some clocked state-storage device specifications include information about maximum clock frequency, but omit duty-cycle information.

**GENERAL:** Marks the beginning of a general condition test. GENERAL constraints have the following form:

```
+ GENERAL:
+   WHEN { <boolean expression> }  
+   MESSAGE = "<message text>"  
+   [ ERRORLIMIT = <value> ]  
+   [ AFFECTS_ALL | AFFECTS_NONE |  
+     AFFECTS (#OUTPUTS) = <output-node-list> ]
```

**Note:** One or more sections can be specified in any order. Note that AFFECTS clauses are only allowed in the PINDLY primitive. The default for the GENERAL constraint is AFFECTS_NONE.

WHEN is used to define a boolean expression, which can describe arbitrary signal relationships that represent the error or condition of interest.

MESSAGE defines the message to be reported by the simulation whenever the WHEN expression evaluates TRUE. The <message text> must be a text constant (enclosed by double quotes “ ”) or a text expression.

**Note:** The <boolean expression> is evaluated whenever the CONSTRAINT primitive is evaluated, that is, whenever any of its inputs undergo a transition. If the result is TRUE, the simulator produces a header containing the time of the occurrence, followed by the <message text>.
General notes

Any or all of the constraint specifications (SETUP_HOLD, WIDTH, FREQ, GENERAL) can appear, in any order, within a CONSTRAINT primitive. Further, more than one constraints of the same type can appear (such as two WIDTH specifications). Each of the constraint specifications is evaluated whenever any inputs to the CONSTRAINT primitive instance change.

All constraint specifications can optionally include a WHEN statement, which is interpreted as “only perform the check when result of <boolean expression> == TRUE.” The WHEN statement is required in the GENERAL constraint.

Each constraint type (SETUP_HOLD, WIDTH, FREQ, and GENERAL) has an associated built-in message. In addition, each instance can include a MESSAGE specification, which takes a text constant (enclosed in double quotes “” ) or text expression. The <additional message text> is appended to the end of the internally-generated, type-specific message which is output whenever a violation occurs. The MESSAGE clause is required for the GENERAL constraint device.

All of the constraint specifications can accept an optional ERRORLIMIT specification. The <value> must be a nonnegative constant or expression. The default <value> is obtained from the value of the DIGERRDEFAULT (set using the .OPTIONS command), which defaults to 20. A value of zero is interpreted as infinity, i.e., no limit. When more than <value> violations of the associated constraint have occurred, no further message output is generated for that constraint checker; other checkers within the CONSTRAINT primitive that have not exceeded their own ERRORLIMITs continue to operate.

During simulation, if the total number of digital violations reported exceeds the value given by DIGERRLIMIT (set using the .OPTIONS (analysis options) on page 71 command), then the simulation is halted. DIGERRLIMIT defaults to infinity.

This CONSTRAINT primitive example below was derived from the 74LS160A device in the model library. It demonstrates how all of the timing checks can be performed by a single primitive.

```plaintext
ULS160ACON CONSTRAINT(10) DPWR DGND
+ CLK ENP ENT CLRBAR LOADBAR A B C D EN
+ IO_LS
+ FREQ:
+ NODE = CLK
+ MAXFREQ = 25MEG
+ WIDTH:
+ NODE = CLRBAR
+ MIN_LO = 20NS
+ SETUP_HOLD:
```
+ DATA(1) = LOADBAR
+ CLOCK LH = CLK
+ SETUPTIME = 20NS
+ HOLDTIME = 3NS
+ WHEN = { CLRBAR! = '0' }
+ SETUP_HOLD:
+   DATA(2) = ENP ENT
+   CLOCK LH = CLK
+   SETUPTIME = 20NS
+   HOLDTIME = 3NS
+   WHEN = { CLRBAR! = '0' & (LOADBAR! = '0 ^ CHANGED(LOADBAR, 0))
+     & CHANGED(EN, 20NS) }
+ SETUP_HOLD:
+   DATA(4) = A B C D
+   CLOCK LH = CLK
+   SETUPTIME = 20NS
+   HOLDTIME = 3NS
+   WHEN = { CLRBAR! = '0' & (LOADBAR! = '1 ^ CHANGED(LOADBAR, 0)) }


Stimulus devices

Stimulus devices apply digital waveforms to a node. Their purpose is to provide the input to a digital circuit or a digital portion of a mixed circuit. They play the same role in the digital simulator that the independent voltage and current sources (V and I devices) do in the analog simulator.

There are two types of stimulus devices: the stimulus generator (STIM), which uses a simple command to generate a wide variety of waveforms; and the file stimulus (FSTIM), which obtains the waveforms from an external file.

Unlike digital primitives, stimulus devices do not have a Timing Model. This is similar to the analog V and I devices: the timing characteristics are described by the device itself, not in a separate model.
Stimulus generator

Device format

U<name> STIM(<width>, <format array>)
+ <digital power node> <digital ground node>
+ <node>*
+ <I/O model name>
+ [STIMULUS=<stimulus name>]
+ [IO_LEVEL=<interface subckt select value>]
+ [TIMESTEP=<stepsize>]
+ <command>*

Arguments and options

<width>
Specifies the number of signals (nodes) output by the stimulus generator.

<format array>
Specifies the format of <value>s used in defining the stimulus. <format array> is a sequence of digits which specifies the number of signals (nodes) that the corresponding digit in a <value> represents. Each digit of <value> is assumed to be in base 2<sup>m</sup> where <sup>m</sup> is the corresponding digit in <format array>. Each <value> must have the same number of digits as <format array>. The sum of the digits in <format array> must be <width>, and each digit must be either a 1, 3, or 4 (that is, binary, octal, or hexadecimal).

<digital power node> <digital ground node>
These nodes are used by the interface devices which connect analog nodes to digital nodes or vice versa. Refer to your PSpice User’s Guide for more information.

<node>*
One or more node names which are output by the stimulus generator. The number of nodes specified must be the same as <width>.

<I/O model name>
The name of an I/O model, which describes the driving characteristics of the stimulus generator. I/O models also contain the names of up to fourDtoA interface subcircuits, which are automatically called by the simulator to handle interface nodes. In most cases, the I/O model named IO_STM can be used from the “dig_io.lib” library file. Refer to your PSpice User’s Guide for a more detailed description of I/O models.
STIMULUS
An optional parameter for referencing a stimulus definition.

IO_LEVEL
An optional device parameter which selects one of the four DtoA interface subcircuits from the I/O model. The simulator calls the selected subcircuit automatically in the event a <node> connects to an analog device. If not specified, IO_LEVEL defaults to 0. Valid values are:

0 = the current value of .OPTIONS DIGIOLVL (default=1)
1 = DtoA1
2 = DtoA2
3 = DtoA3
4 = DtoA4

Refer to your PSpice User’s Guide for more information.

TIMESTEP
Number of seconds per clock cycle, or step. Transition times that are specified in clock cycles (using the C suffix) are multiplied by this amount to determine the actual time of the transition. (See <time> below.) If TIMESTEP is not specified, the default is zero seconds. TIMESTEP has no effect on <time> values which are specified in seconds (using the S suffix).

<command>*
A description of the stimuli to be generated, using one or more of the following.

<time> <value>
LABEL=<label name>
<time> GOTO <label name> <n> TIMES
<time> GOTO <label name> UNTIL GT <value>
<time> GOTO <label name> UNTIL GE <value>
<time> GOTO <label name> UNTIL LT <value>
<time> GOTO <label name> UNTIL LE <value>
<time> INCR BY <value>
<time> DECR BY <value>
REPEAT FOREVER
REPEAT <n> TIMES
ENDREPEAT
FILE=<file name>

<time>
Specifies the time for the new <value>, GOTO, or INCR/DECR command to occur.
Time units

Time values can be stated in seconds or in clock cycles (see TIMESTEP above). To specify a time value in clock cycles, use the C suffix. Otherwise, the units default to seconds.

Absolute/relative times

Times can be absolute, such as 45ns or 10c, or relative to the previous time. To specify a relative time, prefix the time using a “+” such as +5ns or +2c.

<value> is the value for each node ( 0, 1, R, F, X, or Z ). <value> is interpreted using the <format array>.

<label name> is the name used in GOTO statements. GOTO <label name> jumps to the next non-label statement after the <LABEL = <label name>> statement.

<n> is the number of times to repeat a GOTO loop. Use a -1 to specify forever.

Keep the following in mind when using the stimulus command:

Transitions using absolute times within a GOTO loop are converted to relative times based on the time of the previous command and the current step size.

■ GOTO <label name> must specify a label that has been defined in a previous LABEL=<label name> statement.

■ Times must be in strictly ascending order, except that the transition after a GOTO can be at the same time as the GOTO.

A simpler syntax for constructing counted loops in digital stimulus is to use the REPEAT/ENDREPEAT construct. Specify the count value, for example:

```
REPEAT 3 TIMES
  + 5ns 0
  + 5ns 1
ENDREPEAT
```

For an infinite loop, use REPEAT FOREVER (equivalent to REPEAT -1 TIMES). All times within REPEAT loops are interpreted as relative to the start of the loop.

Transition (i.e., time-value pairs) information can be placed in a FILE and accessed one or more times from the STIM device by using the FILE= statement. The syntax for the file contents is identical to what can appear directly in the body of the STIM device <command> section.
Stimulus generator examples

One

The first example creates a simple reset signal, which could be used to set or clear a flip-flop at the beginning of a simulation. The node, named Reset, is set to a level zero at time zero nanoseconds, and to a Z (high impedance) at 20 ns.

```
URESet STIM(1,1) $G_DPWR $G_DGND
  + Reset
  + IO_STM
  + 0s 0
  + 20ns Z
```

This is useful when the Reset node is being driven by another device which does not reset the flip-flop at time zero. By using the library I/O model named IO_STM, the stimulus generator drives with a high strength, and thus overpowers the other output. By outputting a Z for the duration of the simulation, the stimulus generator cannot affect the node.

Two

The second example is a simple example of a clock stimulus which pulses every 5 nanoseconds. It has one output node, OUT1, and the format is represented in binary notation. This example specifies the time as relative to the previous step. IO_STM is an I/O model for stimulus devices and is available in the dig_io.lib library file which comes with the digital simulation feature.

```
UXEx2 STIM( 1, 1 ) $G_DPWR $G_DGND Out1 IO_STM
  + 0s 0; At time=0 initialize Out1; to zero.
  + REPEAT FOREVER; repeats loop indefinitely
    + +5ns 1 ;5ns later Out1 is set to 1
    + +5ns 0 ;5ns later Out1 is set to 0
  + ENDREPEAT
```

Three

The third example illustrates the use of the timestep; a cycle is equal to one nanosecond:

```
UEx3 STIM( 2, 11 ) $G_DPWR $G_DGND 1 2
+ IO_STM TIMESTEP=1ns
+ 0c 00 ; At time=0ns, both nodes are set to 0.
+ REPEAT 4 TIMES ; What's in the loop is repeated ; 4 times
+ +1c 01 ; 1ns later node 1 is set to 0
  ; and node 2 is set to 1.
+ +2c 11 ; 2ns later both nodes set to 1.
+ ENDREPEAT
```

Four

The fourth example has four output nodes. The values of the nodes at each transition are in hexadecimal notation. This is because the <format array> is set to 4, meaning <value> is one digit representing the value of four nodes. Both the absolute and relative timing methods are used, but, at the start of execution, the simulation converts all absolute values to relative values based on the time of the command and the current step size. The timestep is equal to one nanosecond, setting the cycle to one nanosecond:

```
UEx4 STIM( 4, 4 ) $G_DPWR $G_DGND IN1 IN2 IN3 IN4
+ IO_STM TIMESTEP=1ns
+ 0s 0 ; At time=0 seconds, all nodes are set to 0.
+ LABEL=STARTLOOP
+ 10C 1 ; At time=10NS, IN1, IN2, & IN3 are set to 0 and IN4
  ; is set to 1.
+ +5NS 0 ; 5NS later, all nodes are set to 0.
+ 20NS A ; At time=20NS, nodes IN1 & IN3 are set to 1 and
  ; nodes IN2 &
  ; IN4 are to 0.
+ +5NS 0 ; 5NS later, all nodes are set to 0.
+ 30C GOTO STARTLOOP 1 TIMES ; At time=30NS, execute the
  ; first statement of the loop without
  ; a further delay. "1 TIMES" causes the logic to loop
  ; 1 time, actually executing the loop twice.
+ +10C 1 ; After the logic falls through the loop
  ; the second
  ; time and then waiting 10 additional cycles
  ; (or 10 nanoseconds),
  ; IN1, IN2, & IN3 are set to 0 and IN4 is set to 1.
```

Example four produces the following transitions. Note how all of the time values are calculated relative to the previous step:
Five

The fifth example illustrates the use of the INCR BY command used to increment the value of the 16 bit bus:

```
UEx5 STIM ( 16, 4444 ) $G_DPWR $G_DGND
+ 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
+ IO_STM TIMESTEP = 10ns
+ 0s 0000 ; At time=0 seconds, all nodes are set to 0.
+ LABEL=STARTLOOP
+ 10c INCR BY 0001 ; At 100ns, increment bus by 1.
+ 20c GOTO STARTLOOP UNTIL GE 000A ; If the bus value
+ ;is less
+ ;than 10, branch back to STARTLOOP and
+ ;execute the line following the
+ ;label without a further delay.
```
Six

The sixth example has seven output nodes: 1, 2, 3, 4, 5, 6, and 7. The <format array> specifies the notation (1=binary, 3=octal, or 4=hex) used to define the output of those seven nodes. The first two output signals are defined in binary, the next four are in hexadecimal, and the last one is in binary.

In this example, at time equal to one nanosecond, the value of 0070 creates the bit pattern 0001110 on the output nodes. The first two zeros correspond to outputs one and two, the 0111 (7 in hex) corresponds to output signals 3 through 6, and the last zero is the value of output signal 7.

```
UEX6 STIM( 7, 1141 ) $G_DPWR $G_DGND 1 2 3 4 5 6 7 IO_STM
+ 0ns 0000 ; At time=0ns, all nodes are set to 0.
+ REPEAT 4 TIMES ; Repeats what’s in loop 4 times.
+ +1ns 0070 ; At time=1ns, nodes 1, 2, & 3 are set to 0,
; nodes 4, 5, & 6 are set to 1,
; and node 7 is set to 0.
+ +2ns 11F1 ; At time=2ns, all nodes are set to 1.
+ ENDREPEAT
```

```
+ 0ns 0000 ; At time=0ns, all nodes are set to 0.
+ REPEAT 4 TIMES ; Repeats what’s in loop 4 times.
+ +1ns 0070 ; At time=1ns, nodes 1, 2, & 3 are set to 0,
; nodes 4, 5, & 6 are set to 1,
; and node 7 is set to 0.
+ +2ns 11F1 ; At time=2ns, all nodes are set to 1.
+ ENDREPEAT
```
File stimulus

The file stimulus device, FSTIM, allows the digital stimuli to be obtained from a file. This is often useful if the number of stimuli is very large, or if the inputs to one simulation come from the output of another simulation (or even from another simulator). To make the discussion of the FSTIM device more meaningful, the stimulus file format is discussed first.

Stimulus file format

The stimulus file has a simple format which allows outputs from other simulators, or the simulation output file, to be used with little modification. The file consists of two sections: the header, which contains a list of signal names, and the transitions, which is one or more lines containing the transition time and columns of values. The header and transitions must be separated by at least one blank line. Below is a simple example of the stimulus file format.

```
* Header, containing signal names (standard comments are allowed)
Clock, Reset, In1, In2; four signal names

* Beginning of the transitions - note the blank line
0    0000 ; values are in binary
10ns 1100
20ns 0101
30ns 1110
40ns 0111
```

**Header format**

```
[TIMESCALE=<value>]
<signame 1>...<signame n>...
OCT(<signame bit 3> ... <signame lsb>) ...
HEX(<signame bit 4> ... <signame lsb>) ...
```

The header consists of the list of signal names and an optional TIMESCALE value. The signal names can be separated by commas, spaces, or tabs. The list can span several lines, but must not include the + continuation character. The signal names listed correspond to the columns of values in the order that they are listed. Up to 255 signals can be listed in the header, however a maximum of 300 characters are allowed per line.

The OCT and HEX radix functions allows three or four signals to be grouped, respectively, into a single octal or hexadecimal digit in the columns of values. Note that exactly three signals must be included inside the parentheses in the OCT function, and that exactly four signals must be included in the HEX function. Signal names listed without the radix functions default to binary values.

The following example shows the use of the HEX radix function.

```
Clock Reset In1 In2
HEX(Addr7 Addr6 Addr5 Addr4) HEX(Addr3 Addr2 Addr1 Addr0)
```
In this example, there are four binary signals, followed by two occurrences of the HEX radix function, followed by a single binary signal. In the list of transitions following the header, there are seven values which correspond, in order, to the list of signals.

The optional TIMESCALE assignment is used to scale the time values in the transitions. The TIMESCALE assignment must be on a separate line. If unspecified, TIMESCALE defaults to 1.0. See $\text{time}$ below for more information on the use of TIMESCALE.

**Transition format**

$\text{time}$ $\text{value}$*

Following the first blank line after the header, the simulator looks for one or more lines containing transitions. Transitions consist of a time value, followed by one or more values corresponding to the signal names in the header. The $\text{time}$ and list of $\text{values}$ must be separated by at least one space or tab.

$\text{time}$

Transition times are always stated in seconds. Times can be absolute, such as 45ns, 1.2e-8, or 10; or relative to the previous time. To specify relative time, prefix the time using a +, such as +5ns or +1e-9.

Time values are always scaled by the value of TIMESCALE. This is useful if the time values in the file are expressed as whole numbers, but the actual units are, for example, 10ns. An example showing the use of TIMESCALE is given below.

$\text{value}$*

Each value corresponds to a single binary signal (the default) or the entire group of signals inside the OCT or HEX radix functions. The number of values listed must equal the total number of binary signals and radix functions which are specified in the header. Valid
<values> are:

<table>
<thead>
<tr>
<th></th>
<th>Binary</th>
<th>OCT</th>
<th>HEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic/Numeric</td>
<td>0,1</td>
<td>0-7</td>
<td>0-F</td>
</tr>
<tr>
<td>Unknown</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Hi-impedance</td>
<td>Z</td>
<td>Z</td>
<td>Z</td>
</tr>
<tr>
<td>Rising</td>
<td>R</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Falling</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>

When the <value> in a HEX or OCT column is a number, the simulator converts the number to binary and assigns the appropriate logic value of each bit (either zero or one) to the signals inside the radix function. The bits are assigned msb to lsb. When the <value> is X, Z, R, or F, all signals in the radix function take on that value. Note that there can be no falling value in a HEX column because F is used as a numeric value.

The following example shows the use of TIMESCALE and relative <time> values.

```
TIMESCALE=10ns  ; must appear on separate line
Clock, Reset, In1, In2
HEX(Addr7 Addr6 Addr5 Addr4) HEX(Addr3 Addr2 Addr1 Addr0)
ReadWrite
0 0000 00 0
1 110R 4E 0 ; transition occurs at 10ns
2 0101 4E 1
1 3 1111 4E 1 ; transition occurs at 50ns
7 011F C3 0 ; transition occurs at 70ns
8 11X0 C3 1
```

File stimulus device

The file stimulus device, FSTIM, is used to access one or more signals inside a stimulus file. More than one FSTIM device can access the same file. An FSTIM device can even refer to the same signal as another FSTIM device. Any number of stimulus files can be used during a simulation.

```
Device format
U<name> FSTIM(<# outputs>)
+ <digital power node> <digital ground node>
+ <node>*
+ <I/O model name>
+ FILE=<stimulus file name>
+ [IO_LEVEL=<interface subckt select value>]
+ [SIGNAMES=<stimulus file signal name>*]
```
Examples

U1 FSTIM(1) $G_DPWR $G_DGND
+ IN1 IO_STM FILE=DIG1.STM
U2 FSTIM(4) $G_DPWR $G_DGND
+ ADDR3 ADDR2 ADDR1 ADDR0
  + IO_STM
  + FILE = DIG_2.STM
  + SIGNAMES = AD3 AD2 AD1 AD0
U3 FSTIM(4) $G_DPWR $G_DGND
+ CLK PRE J K
  + IO_STM
  + FILE = FLIPFLOP.STM
  + SIGNAMES = CLOCK PRESET

Arguments and options

<# outputs>

Specifies the number of nodes driven by this device.

<digital power node> <digital ground node>

These nodes are used by the interface devices which connect analog nodes to digital nodes or vice versa. Refer to your PSpice User’s Guide for more information.

<node>*

One or more node names which are output by the file stimulus. The number of nodes specified must be the same as <# outputs>.

<I/O model name>

The name of an I/O model, which describes the driving characteristics of the stimulus device. I/O models also contain the names of up to four DtoA interface subcircuits, which are automatically called by the simulator to handle interface nodes. In most cases, the I/O model named IO_STM can be used from the library dig_io.lib. Refer to your PSpice User’s Guide for a more detailed description of I/O models.

FILE

The name of the stimulus file to be accessed by this device. The <stimulus file name> can be specified as a quoted string or as a text expression; see .TEXT (text parameter) on page 110. Note that the FILE device parameter is not optional.
**IO_LEVEL**

An optional device parameter which selects one of the four AtoD or DtoA interface subcircuits from the device’s I/O model. The simulator calls the selected subcircuit automatically in the event a node connecting to the primitive also connects to an analog device. If not specified, IO_LEVEL defaults to 0. Valid values are:

- 0 = the current value of .OPTIONS DIGIOLVL (default=1)
- 1 = AtoD1/DtoA1
- 2 = AtoD2/DtoA2
- 3 = AtoD3/DtoA3
- 4 = AtoD4/DtoA4

Refer to your *PSpice User’s Guide* for more information.

**SIGNAMES**

Used to specify the names of the signals inside the stimulus file which are to be referenced by the FSTIM device. The signal names correspond, in order, to the `<nodes>` connected to the device. If any or all SIGNAMES are unspecified, the simulator looks in the stimulus file for the names of the `<nodes>` given. Because the number of signal names can vary, the SIGNAMES parameter must be specified last. SIGNAMES can be a list of names or text expressions (see .TEXT), or a mixture of the two.
The first example references a file named `dig1.stm`. This file must have a signal named IN1.

The second example references `dig2.stm`. This file would have to have signals named AD3 through AD0. These are mapped, in order, to the nodes ADDR3 through ADDR0, which are driven by this device.

In the third example, the FSTIM device references the file `flipflop.stm`.

The contents of `flipflop.stm` are shown below:

```
J K PRESET CLEAR CLOCK
0 0 0 0 10
10ns 0 0 111
```

In this example, the first two nodes, CLK and PRE, reference the signals named CLOCK and PRESET in the stimulus file. The last two nodes, J and K, directly reference the signals named J and K in the file, and therefore do not need to be listed in SIGNAMES. Note that the order of the SIGNAMES on the FSTIM device does not need to match the order of the names listed in the header of the stimulus file. It is not required that every signal in the file be referenced by an FSTIM device. In the example above, the signal named CLEAR is not referenced. One, several, or all signals in a stimulus file can be referenced by one or more FSTIM devices.
Input/output model

Each digital device in the circuit must reference an I/O model. The I/O model describes the device’s loading and driving characteristics. It also contains the names of up to four AtoD and DtoA subcircuits that the simulator calls to handle interface nodes.

I/O models are common to device families. For example, of the digital devices in the model library, there are only four I/O Models for the entire 74LS family: IO_LS, for standard inputs and outputs; IO_LS_OC, for standard inputs and open-collector outputs; IO_LS_ST, for schmitt trigger inputs and standard outputs; and IO_LS_OC_ST, for schmitt trigger inputs and open-collector outputs. This is in contrast to timing models, which are unique to each device in the library.

Model form

```
.MODEL <I/O model name> UIO [model parameters]
```

<table>
<thead>
<tr>
<th>Model Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>AtoD1</td>
<td>Name of level 1 AtoD interface subcircuit</td>
<td></td>
<td>AtoDDefault</td>
</tr>
<tr>
<td>AtoD2</td>
<td>Name of level 2 AtoD interface subcircuit</td>
<td></td>
<td>AtoDDefault</td>
</tr>
<tr>
<td>AtoD3</td>
<td>Name of level 3 AtoD interface subcircuit</td>
<td></td>
<td>AtoDDefault</td>
</tr>
<tr>
<td>AtoD4</td>
<td>Name of level 4 AtoD interface subcircuit</td>
<td></td>
<td>AtoDDefault</td>
</tr>
<tr>
<td>DIGPOWER</td>
<td>Name of power supply subcircuit</td>
<td></td>
<td>DIGIFPWR</td>
</tr>
<tr>
<td>DRVH</td>
<td>Output high level resistance</td>
<td>ohm</td>
<td>50</td>
</tr>
<tr>
<td>DRVL</td>
<td>Output low level resistance</td>
<td>ohm</td>
<td>50</td>
</tr>
<tr>
<td>DRVZ</td>
<td>Output Z-state leakage resistance</td>
<td>ohm</td>
<td>250 Kohm</td>
</tr>
<tr>
<td>DtoA1</td>
<td>Name of level 1 DtoA interface subcircuit</td>
<td></td>
<td>DtoADefault</td>
</tr>
<tr>
<td>DtoA2</td>
<td>Name of level 2 DtoA interface subcircuit</td>
<td></td>
<td>DtoADefault</td>
</tr>
<tr>
<td>DtoA3</td>
<td>Name of level 3 DtoA interface subcircuit</td>
<td></td>
<td>DtoADefault</td>
</tr>
<tr>
<td>DtoA4</td>
<td>Name of level 4 DtoA interface subcircuit</td>
<td></td>
<td>DtoADefault</td>
</tr>
<tr>
<td>INLD</td>
<td>Input load capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>INR</td>
<td>Input leakage resistance</td>
<td>ohm</td>
<td>30 Kohm</td>
</tr>
<tr>
<td>OUTLD</td>
<td>Output load capacitance</td>
<td>farad</td>
<td>0</td>
</tr>
</tbody>
</table>
INLD and OUTLD are used in the calculation of loading capacitance, which factors into the propagation delay. Refer to your PSpice User’s Guide for more information.

DRVH and DRVL are used to determine the strength of the output. Refer to your PSpice User’s Guide for more information.

DRVZ, INR, and TSTOREMN are used to determine which nets should be simulated as charge storage nets.

AtoD1 through AtoD4 and DtoA1 through DtoA4 are used to hold the names of interface subcircuits. Note that INLD and AtoD1 through AtoD4 do not apply to stimulus generators because they have no input nodes. Refer to your PSpice User’s Guide for more information.

The switching times (TSWLHn and TSWHLn) are subtracted from a device’s propagation delay on the outputs which connect to interface nodes. This compensates for the time it takes the DtoA device to change its output voltage from its current level to that of the switching threshold. By subtracting the switching time from the propagation delay, the analog signal reaches the switching threshold at the correct time (that is, at the exact time of the digital transition). The values for these model parameters should be obtained by measuring the time it takes the analog output of the DtoA (using a nominal analog load attached) to change to the switching threshold after its digital input changes. If the switching time is larger than the

<table>
<thead>
<tr>
<th>Model Parameter</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPWRT</td>
<td>Pulse width rejection threshold</td>
<td>sec</td>
<td>same as propagation delay</td>
</tr>
<tr>
<td>TSTOREMN</td>
<td>Minimum storage time for net to be simulated as a charge</td>
<td>sec</td>
<td>1.0 msec</td>
</tr>
<tr>
<td>TSWHL1</td>
<td>Switching time high to low for DtoA1</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWHL2</td>
<td>Switching time high to low for DtoA2</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWHL3</td>
<td>Switching time high to low for DtoA3</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWHL4</td>
<td>Switching time high to low for DtoA4</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWLH1</td>
<td>Switching time low to high for DtoA1</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWLH2</td>
<td>Switching time low to high for DtoA2</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWLH3</td>
<td>Switching time low to high for DtoA3</td>
<td>sec</td>
<td>0</td>
</tr>
<tr>
<td>TSWLH4</td>
<td>Switching time low to high for DtoA4</td>
<td>sec</td>
<td>0</td>
</tr>
</tbody>
</table>

INLD and OUTLD are used in the calculation of loading capacitance, which factors into the propagation delay. Refer to your PSpice User’s Guide for more information.
propagation delay for an output, no warning is issued, and a delay of zero is used. Note that
the switching time parameters are not used when the output drives a digital node.

DIGPOWER specifies the name of the power supply subcircuit the simulator calls for when an
AtoD or DtoA interface is created. The default value is DIGIFPWR, which is the power supply
subcircuit used by the TTL and CMOS device libraries.

For more information on how to change the default power supplies, refer to your *PSpice
User’s Guide*. 
Digital/analog interface devices

The simulator provides two devices for converting digital logic levels to analog voltages or vice versa. These devices are at the heart of the interface subcircuits found in dig_io.lib. These devices also provide the Digital Files interface for interfacing using external logic simulators.

Digital input (N device)

The digital input device is used to translate logic levels (typically 1s, 0s, Xs, Zs, Rs, and Fs) into representative voltage levels using series resistances. These voltages and resistances model the output stage of a logic device (like a 74LS04) and hence form a digital input to the analog circuit. The logic level information can come from two places: the digital simulator or a file. (The file can be created by hand, or can be an output file from an external logic simulator.)

The general form for a digital input device, and some of the model parameters, are different for devices driven from a file and for those driven by the digital simulation feature. The digital simulation inserts digital input devices automatically when a digital device’s output is connected to an analog component. The automatic insertion of digital input devices is discussed in your PSpice User’s Guide. Examples of the devices that are inserted can be found in the dig_io.lib library file.
General form

**for digital simulation**

\[ N<\text{name}> \ <\text{interface node}> <\text{low level node}> <\text{high level node}> \\
+ <\text{model name}> \\
+ \text{DGTLNET} = <\text{digital net name}> \\
+ <\text{digital I/O model name}> \\
+ [\text{IS} = \text{initial state}] \\

**for digital files**

\[ N<\text{name}> \ <\text{interface node}> <\text{low level node}> <\text{high level node}> \\
+ <\text{model name}> \\
+ [\text{SIGNAME} = <\text{digital signal name}>] \\
+ [\text{IS} = \text{initial state}] \\

**Examples**

\[ N1 \ \text{ANALOG} \ \text{DIGITAL\_GND} \ \text{DIGITAL\_PWR} \ \text{DIN74} \\
+ \text{DGTLNET=}\text{DIGITAL\_NODE} \ \text{IO\_STD} \\
N\text{RESET} \ 7 \ 15 \ 16 \ \text{FROM\_TTL} \\
N12 \ 18 \ 0 \ 100 \ \text{FROM\_CMOS} \ \text{SIGNAME=}VCO\_GATE \ \text{IS}=0 \]

**Model form**

\[ .\text{MODEL} <\text{model name}> \ \text{DINPUT} [\text{model parameters}] \]

---

**Table 3-7 Digital input model parameters**

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHI</td>
<td>capacitance to high level node</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>CLO</td>
<td>capacitance to low level node</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>FILE</td>
<td>digital input file name (digital files only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORMAT</td>
<td>digital input file format (digital files only)</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>S0NAME</td>
<td>state 0 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S0TSW</td>
<td>state 0 switching time</td>
<td>sec</td>
<td></td>
</tr>
<tr>
<td>S0RLO</td>
<td>state 0 resistance to low level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>S0RHI</td>
<td>state 0 resistance to high level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>S1NAME</td>
<td>state 1 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1TSW</td>
<td>state 1 switching time</td>
<td>sec</td>
<td></td>
</tr>
<tr>
<td>S1RLO</td>
<td>state 1 resistance to low level node</td>
<td>ohm</td>
<td></td>
</tr>
</tbody>
</table>
Table 3-7 Digital input model parameters

<table>
<thead>
<tr>
<th>Model parameters¹</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1RHI</td>
<td>state 1 resistance to high level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>S2NAME</td>
<td>state 2 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S2TSW</td>
<td>state 2 switching time</td>
<td>sec</td>
<td></td>
</tr>
<tr>
<td>S2RLO</td>
<td>state 2 resistance to low level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>S2RHI</td>
<td>state 2 resistance to high level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>S19NAME</td>
<td>state 19 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S19TSW</td>
<td>state 19 switching time</td>
<td>sec</td>
<td></td>
</tr>
<tr>
<td>S19RLO</td>
<td>state 19 resistance to low level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>S19RHI</td>
<td>state 19 resistance to high level node</td>
<td>ohm</td>
<td></td>
</tr>
<tr>
<td>TIMESTEP</td>
<td>digital input file step-size (digital files only)</td>
<td>sec</td>
<td>1E-91</td>
</tr>
</tbody>
</table>

¹ See .MODEL (model definition) on page 58.

Note: For more information on using the digital input device to simulate mixed analog/digital systems refer to your PSpice User’s Guide.

As shown below, the digital input device is modeled as a time varying resistor from <low level node> to <interface node>, and another time varying resistor from <high level node> to <interface node>. Each of these resistors has an optional fixed value capacitor in parallel: CLO and CHI. When the state of the digital signal changes, the values of the resistors change (exponentially) from their present values to the values specified for the new state over the switching time specified by the new state. Normally the low and high level nodes would be attached to voltage sources which would correspond to...
the highest and lowest logic levels. (Using two resistors and two voltage levels, any voltage between the two levels can be created at any impedance.

For a digital simulation driven digital input, the parameters

\[ \text{DGTLNET} = \text{<digital net name>} \ \text{<digital I/O model name>} \]

must be specified. Refer to your *PSpice User’s Guide* for more information on digital I/O models. The digital net must not be connected to any analog devices, otherwise the automatic analog/digital interface process disconnects the digital input device from the digital net.

Digital simulation can send states named 0, 1, X, R, F, and Z to a digital input device. The simulation stops if the digital simulation sends a state which is not modeled (does not have SnNAME, SnTSW, SnRLO, and SnRHI specified) to a digital input device.

The initial state of a digital simulation driven digital input is controlled by the bias point solution of the analog/digital system. It is sometimes necessary to override this solution (for example, an oscillator which contains both analog and digital parts). The optional parameter

\[ \text{IS} = \text{<initial state name>} \]

can be used to do this. The digital input remains in the initial state until the digital simulation value changes from its TIME=0 value.

The model parameters FILE, FORMAT, and TIMESTEP are not used by digital simulation driven digital input devices, and only the FILE parameter is used for VIEWsim A/D driven digital inputs. For file driven digital inputs the FILE parameter defines the name of the file to be read, and the FORMAT parameter defines the format of the data in that file. The TIMESTEP parameter defines the conversion between the digital simulation’s integer timing tick numbers and the simulation’s floating-point time values:

\[ \text{tick number} \cdot \text{TIMESTEP} = \text{seconds} \]

**Note:** Tick number must be an integer.
For a file driven or VIEWsim A/D driven digital input, the DGTLNET parameter must not be specified, but the optional parameter

   SIGNAME = <digital signal name>

is used to specify the name of the digital signal in the file (or the digital net name in VIEWsim A/D). If no SIGNAME is given, then the portion of the device name after the leading N identifies the name of the digital signal.

The parameter

   IS=<initial state name>

can be used as described above to override the initial (TIME=0) values from the file.

The file name DGTPSPC is used with VIEWsim A/D to tell the simulator to get digital state values from the VIEWsim A/D interface, rather than a file.

Any number of digital input models can be specified, and both file driven and digital simulation driven digital inputs can be used in the same circuit. Different digital input models can reference the same file, or different files. If the models reference the same file, the file must be specified in the same way, or unpredictable results occur. For example, if the default drive is C:, then one model should not have FILE=C:TEST.DAT if another has FILE=TEST.DAT.

For diagnostic purposes, the state of the digital input can be viewed in Probe by specifying B(Nxxx). The value of B(Nxxx) is 0.0 if the current state is S0NAME, 1.0 if the current state is S1NAME, and so on through 19.0. B(Nxxx) cannot be specified on a .PRINT, .PLOT, or .PROBE line. (For digital simulation, the digital window of Probe provides a better way to look at the state of the digital net connected to the digital input.)
Digital output (O Device)

The digital output device is used to translate analog voltages into digital logic levels (typically 1, 0, X, R, or F). The conversion of a voltage into a logic level, models the input stage of a logic device (like a 74LS04) and hence forms a digital output from the analog circuit. The logic level information can go to two places: the digital simulation, or a file. (The file can simply be inspected manually, or can be used as a stimulus file for an external logic simulator.)

General form for digital simulation

\[ O<\text{name}> \ <\text{interface node}> \ <\text{reference node}> \ <\text{model name}> + \ \text{DGTLNET} = \ <\text{digital net name}> \ <\text{digital I/O model name}> \]

for digital files

\[ O<\text{name}> \ <\text{interface node}> \ <\text{reference node}> \ <\text{model name}> + \ [\text{SIGNAME} = \ <\text{digital signal name}>] \]

Model form

\[ .\text{MODEL} \ <\text{model name}> \ \text{DOUTPUT} \ [\text{model parameters}] \]

Examples

\[
\begin{align*}
O12 \ \text{ANALOG\_NODE} & \ \text{DIGITAL\_GND} \ \text{DO74} \ \text{DGTLNET=}\ \text{DIGITAL\_NODE} \\
& \ \text{IO\_STD} \\
& \ \text{OVCO} \ 17 \ 0 \ \text{TO\_TTL} \\
& \ \text{O5} \ 22 \ 100 \ \text{TO\_CMOS} \ \text{SIGNAME=}\ \text{VCO\_OUT}
\end{align*}
\]

Table 3-8 Digital output model parameters

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHGONLY</td>
<td>0: write each timestep, 1: write upon change</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>CLOAD</td>
<td>output capacitor</td>
<td>farad</td>
<td>0</td>
</tr>
<tr>
<td>FILE</td>
<td>digital input file name (digital files only)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FORMAT</td>
<td>digital input file format (digital files only)</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>RLOAD</td>
<td>output resistor</td>
<td>ohm</td>
<td>1/GMIN</td>
</tr>
<tr>
<td>S0NAME</td>
<td>state 0 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S0VLO</td>
<td>state 0 low level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>S0VHI</td>
<td>state 0 high level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>S1NAME</td>
<td>state 1 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S1VLO</td>
<td>state 1 low level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>S1VHI</td>
<td>state 1 high level voltage</td>
<td>volt</td>
<td></td>
</tr>
</tbody>
</table>
The general form for a digital output device, and some of the model parameters, are different for devices that drive a file (or VIEWsim A/D) and those that drive the digital simulation feature. The digital simulation inserts digital output devices automatically when a digital device’s input is connected to an analog component. The automatic insertion of digital output devices is discussed in your *PSpice User’s Guide*, and examples of the devices which are inserted can be found in the *dig_io.lib* library file. 

**Note:** For more information on using the digital output device to simulate mixed analog/digital systems, refer to your *PSpice User’s Guide*.

As shown in Figure 3-1 on page 441, the digital output device is modeled as a resistor and capacitor, of the values specified in the model statement, connected between <interface node> and <reference node>. At times which are integer multiples of TIMESTEP, the state of the device node is determined and written to the specified file.

### Table 3-8 Digital output model parameters, *continued*

<table>
<thead>
<tr>
<th>Model parameters</th>
<th>Description</th>
<th>Units</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2NAME</td>
<td>state 2 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S2VLO</td>
<td>state 2 low level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>S2VHI</td>
<td>state 2 high level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>S19NAME</td>
<td>state 19 character abbreviation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S19VLO</td>
<td>state 19 low level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>S19VHI</td>
<td>state 19 high level voltage</td>
<td>volt</td>
<td></td>
</tr>
<tr>
<td>SXNAME</td>
<td>state applied when the interface node voltage falls outside all ranges</td>
<td></td>
<td>“?”</td>
</tr>
<tr>
<td>TIMESTEP</td>
<td>digital input file step-size</td>
<td>sec</td>
<td>1E-9</td>
</tr>
<tr>
<td>TIMESCALE</td>
<td>scale factor for timestep (digital files only)</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>

1. See [MODEL (model definition)](page 58) on page 58.
Figure 3-1 Digital output model

The process of converting the input node voltage to a logic state begins by first obtaining the difference in voltage between the <interface node> and the <reference node>. The DOUTPUT model defines a voltage range, form SxVLO to SxVHI, for each state. If the input voltage is within the range defined for the current state, no state change occurs. Otherwise, the simulator searches forward through the model, starting at the current state, to find the next state whose voltage range contains the input voltage. This state then becomes the new state. When the end of the list (S19) is reached, the simulator wraps around to S0 and continues.

If the entire model has been searched and no valid voltage range has been found, the simulator generates a simulation warning message. Further if the O device is interfacing at the digital simulator, and the SXNAME parameter has not been specified in the model, the simulator uses the state whose voltage range is closed to the input voltage. Otherwise it uses SXNAME as the new state.

This circular state searching mechanism allows hysteresis to be modeled directly. The following model statement models the input thresholds of a 7400 series TTL Schmitt-trigger input. Notice that the 0.8 volt overlap between the 0 state voltage range and the 1 state voltage range.

```
.model D074_STd output (  
+  s0name="0"  s0vlo=1.5  s0vhi=1.7  
+  s1name="1"  s1vlo=0.9  s1vhi=7.0  
+  )
```

Starting from the 0 state, a positive-going voltage must cross 1.7 volts to get out of the 0 state's voltage range. The next state which contains that voltage is 1. Once there, a negative-going voltage must go below 0.9 volts to leave the 1 state's range. Since no further states are defined, the simulator wraps around back to state 0, which contains the new voltage.

For a digital output driving digital simulation, the parameters

```
DGTLNET = <digital net name> <digital I/O model name>
```
must be specified. Refer to your *PSpice User’s Guide* for more information on digital I/O models. The digital net must not be connected to any analog devices, otherwise the automatic analog/digital interface process disconnects the digital output device from the analog net.

For interfacing using digital simulation, the state names must be 0, 1, X, R, F, or Z (Z is usually not used however, since high impedance is not a voltage level). Other state names cause the simulator to stop if they occur; this includes the state ? that occurs if the voltage is outside all the ranges specified.

The model parameters TIMESCALE, FILE, CHGONLY, and FORMAT are not used for digital outputs which drive digital simulation, but the TIMESTEP is used. The TIMESTEP value controls how accurately the analog simulator tries to determine the exact time at which the node voltage crosses a threshold.

To be sure that the transition time is accurately determined, the analog simulator has to evaluate the analog circuit at intervals no larger than TIMESTEP when a transition is about to occur. The default value for TIMESTEP is 1ns, or 1/DIGFREQ (a .OPTIONS (analysis options) on page 71 option) if it is larger. In many circuits, this is a much greater timing resolution than is required, and some analog simulation time can be saved by increasing the TIMESTEP value.

For digital outputs which write files, or drive VIEWsim A/D, the parameter  

```
SIGNAME = <digital signal name>
```

can be used to specify the name written to the file of the digital signal (or for VIEWsim A/D, the name of the VIEWsim net). If SIGNAME is not specified, then the portion of the device name after the leading O identifies the name of the digital signal.

For digital outputs which write files, the FILE parameter defines the name of the file to be written, and the FORMAT parameter defines the format of the data written to that file.

The file name `PSPCDGTL` is used with VIEWsim A/D to tell the simulator to send the digital state values to the VIEWsim A/D interface, rather than a file. For VIEWsim A/D, the parameters FORMAT and CHGONLY are ignored.

The state of each device is written to the output file at times which are integer multiples of TIMESTEP. The time that is written is the integer:

```
time = TIMESCALE·TIME/TIMESTEP
```

TIMESCALE defaults to 1, but if digital simulation is using a very small timestep compared to the analog simulation timestep, it can speed up the simulation to increase the value of both TIMESTEP and TIMESCALE. This is because the simulator must take timesteps no greater than the digital TIMESTEP size when a digital output is about to change, in order to accurately determine the exact time that the state changes. The value of TIMESTEP should therefore
be the time resolution required at the analog-digital interface. The value of TIMESCALE is then used to adjust the output time to be in the same units as digital simulation uses.

For example, if a digital simulation using a timestep of 100 ps is being run, but the circuit has a clock rate of 1μs, setting TIMESTEP to 0.1μs should provide enough resolution. Setting TIMESCALE to 1000 scales the output time to be in 100 ps units.

If CHGONLY = 1, only those timesteps in which a digital output state changes are written to the file.

Any number of digital output models can be specified, and both file writing and digital simulation driving digital outputs can be used in the same circuit. Different digital output models can reference the same file, or different files. If the models reference the same file, the file must be specified in the same way, or unpredictable results occur. For example, if the default drive is C:, then one model should not have FILE=C:TEST.DAT if another has FILE=TEST.DAT.

For diagnostic purposes, the state of the digital output can be viewed in Probe by specifying B(Oxxx). The value of B(Oxxx) is 0.0 if the current state is S0NAME, 1.0 if the current state is S1NAME, and so on through 19.0. B(Oxxx) cannot be specified on a .PRINT, .PLOT, or .PROBE line. (For digital simulation, the digital window of Probe provides a better way to look at the state of the digital net connected to the digital output.)
## Digital model libraries

<table>
<thead>
<tr>
<th>File</th>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>7400.LIB</td>
<td>7400-series TTL</td>
</tr>
<tr>
<td>74AC.LIB</td>
<td>Advanced CMOS</td>
</tr>
<tr>
<td>74ACT.LIB</td>
<td>TTL-compatible, Advanced CMOS</td>
</tr>
<tr>
<td>74AS.LIB</td>
<td>Advanced Schottky TTL</td>
</tr>
<tr>
<td>74F.LIB</td>
<td>FAST</td>
</tr>
<tr>
<td>74H.LIB</td>
<td>High-Speed TTL</td>
</tr>
<tr>
<td>74HCT.LIB</td>
<td>TTL-compatible, High-Speed CMOS</td>
</tr>
<tr>
<td>74HC.LIB</td>
<td>High-Speed CMOS</td>
</tr>
<tr>
<td>74L.LIB</td>
<td>Low-Power TTL</td>
</tr>
<tr>
<td>74LS.LIB</td>
<td>Low-Power Schottky TTL</td>
</tr>
<tr>
<td>74S.LIB</td>
<td>Schottky TTL</td>
</tr>
<tr>
<td>CD4000.LIB</td>
<td>CD4000 devices</td>
</tr>
<tr>
<td>DIG_ECL.LIB</td>
<td>10 K and 100K ECL devices</td>
</tr>
<tr>
<td>DIG_GAL.LIB</td>
<td>GAL devices</td>
</tr>
<tr>
<td>DIG_IO.LIB</td>
<td>I/O models, AtoD andDtoA interface subcircuits, digital power supply subcircuits</td>
</tr>
<tr>
<td>DIG_MISC.LIB</td>
<td>pull-up/down resistors, delay line</td>
</tr>
<tr>
<td>DIG_PAL.LIB</td>
<td>PAL devices</td>
</tr>
<tr>
<td>DIG_PRIM.LIB</td>
<td>Digital primitives</td>
</tr>
<tr>
<td>NOM.LIB</td>
<td>master library: which references NOM_DIG.LIB,¹ which references each of the above libraries.</td>
</tr>
</tbody>
</table>

¹. Depending upon the platform being worked on, NOM.LIB references the appropriate list of libraries. For “digital only” platforms, NOM.LIB references NOM_DIG.LIB.
7400-series TTL and CMOS library files

The PSpice Library List provides a list of all parts contained in the PSpice libraries. In the Digital section, each part is sorted by Device Type, and includes the part name and the library where it is located. In the Index by Library Name section, each library is listed along with the the parts it contains. This information is needed if a netlist is created manually. Netlists normally are generated automatically by the schematic capture package.

4000-series CMOS library

The PSpice Library List provides a list of all parts contained in the PSpice libraries. In the Digital section, each part is sorted by Device Type, and includes the part name and the library where it is located. In the Index by Library Name section, each library is listed along with the the parts it contains. This information is needed if a netlist is created manually. Netlists normally are generated automatically by the schematic capture package.

If power supply nodes on CD4000 devices are not specified in the circuit, they can use the default power supply nodes $G_CD4000_VDD and $G_CD4000_VSS, which default to 5 volts. A new power supply can be created, and new power supply nodes can be specified to the devices in the circuit. Refer to your PSpice User’s Guide for more information on specifying your own power supplies. Output drives and input thresholds are correctly modeled for power supplies between 3 and 18 volts. Currently, propagation delays do not vary using supply voltages. For correct propagation delays at supply voltages other than 5 volts, the timing models in cd4000.lib have to be modified.
Programmable array logic devices

Using a PLD from the library is just like using any other logic device from the library, except that the simulator has to be told the name of the JEDEC file which contains the program for the part. A TEXT parameter name JEDEC_FILE is used to specify the file name, as shown in the following example:

```
X1 IN1 IN2 IN3 IN4 IN5 IN6 IN7 IN8 IN9 IN10 IN11 IN12
+ IN13 IN14
+ OUT1 OUT2 OUT3 OUT4
+ PAL14H4
+ TEXT: JEDEC_FILE = "myprog.jed"
```

This example creates a 14H4 PAL which is programmed by the JEDEC file `myprog.jed`. 
Behavioral Simulation Functions

General functions
ZERO() on page 448
CEIL(arg) on page 448
INTQ(arg) on page 449
ONE(expression) on page 448
FLOOR(arg) on page 449

Simulation variables
DELTAD() on page 450
STATE() on page 451

Simulation functions
BREAK() on page 452

System Variables
System Variables on page 453
ZERO()

Purpose: Evaluates the function, and returns the value 0.

General form: ZERO(expression)

Arguments: expression
This can be any arithmetic expression. The expression is evaluated, its value ignored and the output is 0.

Examples: E1 1 0 value = {ZERO(sin(PI/2))}

Comments: In the example above the voltage V(1) will always be 0.

ONE(expression)

Purpose: Evaluates the function, and returns the value 1.

General form: ONE(expression)

Arguments: expression
This can be any arithmetic expression. The expression is evaluated, its value ignored and the output is 1.

Examples: EE15 12 0 VALUE={ if ( V(14)>0.7, ONE(PI), ZERO(PI)) }

Comments: In the example above, if V(14) is greater than 7, the V(12) is 1 volts, else it is 0 volts.

CEIL(arg)

Purpose: Returns an integer value equal to or greater than the value of the argument passed to the CEIL function.

General form: CEIL(arg)

Arguments: arg
The argument passed to the CEIL function, can be a numeric value or an expression that evaluates to a numeric value.
FLOOR(arg)

Purpose
Returns an integer value.

General form
\[ \text{FLOOR}(\text{arg}) \]

Arguments
\[ \text{arg} \]

The argument passed to the FLOOR function, can be a numeric value or an expression that evaluates to a numeric value.

Examples
\[ \text{E2 2 0 value = \{FLOOR(PI)\}} \]
\[ \text{G3 3 0 value = \{FLOOR(5)\}} \]
\[ \text{FLOOR(PI/2) = 1} \]

Comments
If \( \text{arg} \) is an integer, the return value is equal to the argument value. If \( \text{arg} \) is a non-integer value, the return value is the nearest integer smaller than the argument value.

In the first example, \( V(2) \) will be 3. Similarly, in the second example the value of the charge source will be set to 5.

INTQ(arg)

Purpose
Returns 1 if the argument is an integer, else returns 0

General form
\[ \text{INTQ}(\text{arg}) \]

Arguments
\[ \text{arg} \]

The argument passed to integer function, \( \text{INTQ} \), can be a numeric value or an expression that evaluates to a numeric value.

Examples
\[ \text{E1 1 0 value = \{CEIL(5)\}} \]
\[ \text{E1 1 0 value = \{CEIL(5.4)\}} \]

Comments
If \( \text{arg} \) is an integer, the return value is equal to the argument value. If \( \text{arg} \) is a non-integer value, the return value is the nearest integer greater than the argument value.

In the first example, the value of the charge source will be 4. Similarly, the value of \( V(1) \) in the second and third examples would be 5 and 6, respectively.
DELTA()

Purpose
Returns the value of last three time steps taken by the simulator

General form
DELTA(n)

Arguments
n
Valid values are 0, 1, 2, and 3.

The argument n can also be an expression that evaluates to an integer value. If the evaluated value is greater than three, an error is generated.

Examples
Exy 11 0 VALUE= {delta(1)}

Comments
If the value of n is 0, the result is same using the simulation variable Delta.
At time t=n,
\[ \text{delta}(1) = t_n - t_{n-1} \]

TIME()

Purpose
Returns the time point value for last three simulation time-points.

General form
time(n)

where \( n = 0, 1, 2, 3 \)

Arguments
n
Valid values are 0, 1, 2, and 3.

The argument n can also be an expression that evaluates to an integer value. If the evaluated value is greater than three, an error is generated.
**STATE()**

**Purpose**
Returns the history of the behavioral source up to last three states.

**General form**
STATE(n,source)

**Arguments**
- **n**
  - Valid values are 0, 1, 2, and 3.
  - The argument n can also be an expression that evaluates to an integer value. If the evaluated value is greater than three, an error is generated. And if the state were not mentioned, then the current state would be assumed.

**<source>**
- A valid voltage, current, flux, or charge source

**Examples**
- Esin 5 0 value={sin(1e4*time)}
- Estate1 6 0 value={state(1,V(5))}
- Estate2 7 0 value={state(2,V(5))}
- Estate3 8 0 value={state(3,V(5))}

**Examples**
- Exy 2 0 VALUE= {time(0)}
- Exy 11 0 VALUE= {2*time(1)}

**Comments**
For \( n = 0 \), the function returns the current simulation time-point. This is same value that you get value by adding the simulation variable DELTA to the last simulation time-point.

If the value of \( n \) is 0, the result is same using the simulation variable Time.

In the first example, voltage V(2) will be same as the current time. The value of V(2) at different time is shown below.

\[
\begin{align*}
  t = 0, & \quad V(2) = 0 \\
  t = 1, & \quad V(2) = 1 \\
  t = 0.73, & \quad V(2) = 0.73
\end{align*}
\]
**BREAK()**

**Purpose**
Schedules a break point at the specified time.

**General form**

\[ \text{BREAK}(\text{time}) \]

**Example**

\[ \text{Exy 12 0 \ V= \{BREAK(TIME+1E-5)\}} \]

**Comments**
The scheduling would be deferred until the correct solution at current time point.

Use BREAK() to ensure that simulator calculates values at a specific time point, defined by the value of the argument passed to the BREAK().

In the example statement, simulator will calculate the output V at time \( t \), where \( t = \text{current\_time} + 10 \ \text{millisecond} \).
System Variables

Table 4-1 on page 453, lists the system variables supported by the PSpice engine. You can use these variables in expressions to be evaluated using PSpice engine. These variables cannot be used in the trace expressions in the Probe window.

**Table 4-1 System variables**

<table>
<thead>
<tr>
<th>This variable...</th>
<th>Evaluates to this...</th>
</tr>
</thead>
<tbody>
<tr>
<td>TEMP</td>
<td>Temperature values resulting from a temperature, parametric temperature, or DC temperature sweep analysis. The default temperature, TNOM, is set in the Options tab of the Simulation Settings dialog box. TNOM defaults to 27°C.</td>
</tr>
<tr>
<td><strong>Note:</strong> TEMP can only be used in expressions pertaining to analog behavioral modeling and the propagation delay of digital models.</td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> If a passive or semiconductor device has an independent temperature assignment, then TEMP does not represent that device’s temperature.</td>
<td></td>
</tr>
<tr>
<td>To find out more about customizing temperatures for passive or semiconductor devices, <code>.MODEL (model definition)</code> on page 58.</td>
<td></td>
</tr>
<tr>
<td>TIME</td>
<td>Time values resulting from a transient analysis. If no transient analysis is run, this variable is undefined.</td>
</tr>
<tr>
<td><strong>Note:</strong> TIME can only be used in analog behavioral modeling expressions.</td>
<td></td>
</tr>
<tr>
<td>RELTOL</td>
<td>Relative tolerance of Voltage and current</td>
</tr>
<tr>
<td>ABSTOL</td>
<td>Current tolerance Describes the best accuracy of currents in a simulation run. The value of this variable is specified in the Options tab of the Simulation Settings dialog box.</td>
</tr>
<tr>
<td>VNTOL</td>
<td>Voltage tolerance Describes the best accuracy of voltages in a simulation run. The value of this variable is specified in the Options tab of the Simulation Settings dialog box.</td>
</tr>
</tbody>
</table>
### Table 4-1 System variables, *continued*

<table>
<thead>
<tr>
<th>This variable...</th>
<th>Evaluates to this...</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHGTOL</td>
<td>Charge tolerance</td>
</tr>
<tr>
<td></td>
<td>Describes the best accuracy of charges. The value of this variable is specified in the <em>Options</em> tab of the <em>Simulation Settings</em> dialog box.</td>
</tr>
<tr>
<td>GMIN</td>
<td>Indicates the minimum conductance used for any branch. The value of this variable is specified in the <em>Options</em> tab of the <em>Simulation Settings</em> dialog box.</td>
</tr>
</tbody>
</table>

⚠️ **Important**

System variables covered in this section are reserved keywords for PSpice. You can use these variables in your circuits file but cannot redefine them as user-defined variables. For example, you cannot redefine these parameters in `.PARAM` or `.PARAMS` statements. An exception to this are the keywords GMIN, PI, and TEMP. These three variables can be declared within a `.SUBCKT` statement, to have a user-defined value. Redefining GMIN, PI, and TEMP outside a `.SUBCKT` statement is not supported.
Behavioral Simulation Models

This chapter provides an overview of the behavioral models available with PSpice A/D. Using the behavioral models covered in this chapter, you can easily model the behavior of a system on your schematic. Besides providing you with an easy-to-use graphical interface, some of the models, such as DC motors, and tachometers, can be used to simulate systems which have no implementation in electrical circuitry. The models covered in this chapter are shipped with function.lib (Function library on page 460) and spice_elem.lib (Spice_elem library on page 498).

Table 5-1 on page 455 lists the application-specific categories and the library elements that fall in that category. For example, behavioral models for electromechanical parts, such as tachometer and DC motor, are listed under the mechanical elements category. Similarly, models such as ABS, SUM, and INTEGRATOR, are listed under arithmetic functions.

Table 5-2 on page 460 and Table 5-9 on page 498 provide the alphabetical listing of the elements in the function.lib and spice_elem.lib, respectively.

### Table 5-1 Model categories

<table>
<thead>
<tr>
<th>Category...</th>
<th>Library element...</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laplace Domain Sources</td>
<td><img src="image" alt="BEHAV_FREQ" /></td>
<td>These are also referred to as H(s) sources. You can use these behavioral sources to define voltage and current as an expression of s. The output is in the frequency domain. You can use these sources in either the time or frequency domain.</td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="VOLTAGE_FREQ" /></td>
<td></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="CURRENT_FREQ" /></td>
<td></td>
</tr>
<tr>
<td>Arbitrary Voltage Sources</td>
<td><img src="image" alt="BEHAV_GEN" /></td>
<td>These are also known as F(x) sources.</td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="VOLTAGE_GEN" /></td>
<td></td>
</tr>
<tr>
<td></td>
<td><img src="image" alt="CURRENT_GEN" /></td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-1 Model categories, continued

<table>
<thead>
<tr>
<th>Category...</th>
<th>Library element...</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mechanical Elements</td>
<td>▪ DC Motor</td>
<td>The behavioral models provided for the mechanical elements use a mechanical-to-electrical analogy in which current represents torque and voltage represents angular velocity. They are calibrated so that one volt corresponds to one radian per second of shaft velocity, and a current of one Ampere is equal to one Newton-meter of torque.</td>
</tr>
<tr>
<td></td>
<td>▪ Tachometer</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Gearbox</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Flywheel</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Viscous Friction</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Coil Spring</td>
<td></td>
</tr>
<tr>
<td>Laplace Domain Functions</td>
<td>▪ GAIN2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Real Pole (REALPOLE2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Real Zero (REALZERO2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Complex Pole, Frequency and Damping (COMPLEX_FZ)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Complex Pole, Real and Imaginary (COMPLEX_RI)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ First Order Transfer Function (FY1)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Second Order Transfer Function (FY2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Third Order Transfer Function (FY3)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ Fourth Order Transfer Function (FY4)</td>
<td></td>
</tr>
<tr>
<td>Analog Switches</td>
<td>▪ ASW</td>
<td></td>
</tr>
<tr>
<td></td>
<td>▪ ASW1</td>
<td></td>
</tr>
</tbody>
</table>
### Table 5-1  Model categories, *continued*

<table>
<thead>
<tr>
<th>Category...</th>
<th>Library element...</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Functions</td>
<td>■ DELAY</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ DELAY1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ ONE_SHOT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ VCO</td>
<td></td>
</tr>
<tr>
<td>Arithmetic Functions</td>
<td>■ ABS</td>
<td>The arithmetic function blocks let you perform basic mathematical operations, such as addition and multiplication, on signals.</td>
</tr>
<tr>
<td></td>
<td>■ SUM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ DIFFERENCE</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Multiply(MULTIPLIER2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Divide (DIVIDER2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Square Root(SQRT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Exponential (EXP)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Natural Log (LNX)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Maximum value (MAX)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ Minimum Value (MIN)</td>
<td></td>
</tr>
<tr>
<td>Counter Functions</td>
<td>■ DIV_BY_2</td>
<td>These are digital trigger-edge counters, that are used as frequency dividers.</td>
</tr>
<tr>
<td></td>
<td>■ DIV_BY_3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>■ DIV_BY_4</td>
<td>Each counter has two inputs: a clock input labeled with a &gt; symbol, and a reset input labeled R.</td>
</tr>
<tr>
<td></td>
<td>■ DIV_BY_5</td>
<td></td>
</tr>
</tbody>
</table>
# Table 5-1 Model categories, continued

<table>
<thead>
<tr>
<th>Category...</th>
<th>Library element...</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time Domain Functions</td>
<td>TIME</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Differentiator (DXDT)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Integrator (INTEGRATOR2)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Current Limiter (ILIM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Voltage Limiter (VLIM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Slew Rate Limiter (SLEW LIMIT)</td>
<td></td>
</tr>
<tr>
<td>Switch Models</td>
<td>CC_SWITCH</td>
<td>Switch models allow you to model switches in PSpice.</td>
</tr>
<tr>
<td></td>
<td>VV_SWITCH</td>
<td>The two types of switches that are supported are current controlled switches (CC_SWITCH) and voltage controlled switches (VC_SWITCH).</td>
</tr>
<tr>
<td>Dependent sources</td>
<td>Current Controlled Current Source (CCCS)</td>
<td>The sources covered in this section can be used to model constant, linear, or nonlinear dependent current or voltage sources by expressing the current or voltage as a polynomial function of the current. This function is expressed in terms of the current through the controlling source or the voltage difference between the controlling nodes.</td>
</tr>
<tr>
<td></td>
<td>Current Controlled Voltage Source (CCVS)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Voltage Controlled Current Source (VCCS)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Voltage Controlled Voltage Source (VCVS)</td>
<td></td>
</tr>
</tbody>
</table>
Table 5-1  Model categories, *continued*

<table>
<thead>
<tr>
<th>Category...</th>
<th>Library element...</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controlled sources</td>
<td>□ Single Controller</td>
<td>These are controlled voltage and current sources, where the output is governed by the controlling current or voltage source, currentsense and voltagesense, respectively. Depending on the number of controllers influencing the output, PSpice support two types of controlled sources:</td>
</tr>
<tr>
<td></td>
<td>-- CCS10</td>
<td>□ Double Controller</td>
</tr>
<tr>
<td></td>
<td>-- CVS10</td>
<td>□ Single Controller</td>
</tr>
<tr>
<td></td>
<td></td>
<td>These are the dependent source where output current or voltage is controlled by only one controller. <strong>CVS10</strong> is a controlled voltage source and <strong>CCS10</strong> is a controlled current source.</td>
</tr>
<tr>
<td></td>
<td>-- CCS23</td>
<td>□ Double Controller</td>
</tr>
<tr>
<td></td>
<td>-- CVS23</td>
<td>These are the dependent source, where the output is controlled by more than one controller. <strong>CVS23</strong> is a controlled voltage source and <strong>CCS23</strong> is a controlled current source.</td>
</tr>
</tbody>
</table>
Function library

Table 5-2  Elements in the Function library

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS</td>
<td>Returns absolute value of input</td>
<td>Calculates absolute value of the argument x, which can either be a number or an expression.</td>
</tr>
<tr>
<td>ASW</td>
<td>Is an analog switch</td>
<td>See ASW on page 467.</td>
</tr>
<tr>
<td>ASW1</td>
<td>Analog Switch</td>
<td>See ASW1 on page 468 and ASW on page 467.</td>
</tr>
<tr>
<td>BEHAV_FREQ</td>
<td>Frequency Domain Behavioral Voltage Source</td>
<td>See BEHAV_FREQ on page 469.</td>
</tr>
<tr>
<td>BEHAV_GEN</td>
<td>Arbitrary Behavioral Voltage Sources</td>
<td>See BEHAV_GEN on page 470.</td>
</tr>
<tr>
<td>CHARGE_GEN</td>
<td>Charge generator</td>
<td>See Charge source on page 178</td>
</tr>
<tr>
<td>COILSPRING</td>
<td>coil spring</td>
<td>See Coil Sprung on page 471.</td>
</tr>
<tr>
<td>COMPLEX_FZ</td>
<td>Complex Pole, frequency and damping</td>
<td>See COMPLEX_FZ on page 477.</td>
</tr>
<tr>
<td>COMPLEX_RI</td>
<td>Complex Pole, Real and Imaginary</td>
<td>See COMPLEX_RI on page 480.</td>
</tr>
<tr>
<td>CURRENT_FREQ</td>
<td>frequency-defined current source for describing continuous systems</td>
<td>The output current is specified by the IOUT property. For information on other properties, see BEHAV_FREQ on page 469.</td>
</tr>
</tbody>
</table>
### Table 5-2 Elements in the Function library, *continued*

<table>
<thead>
<tr>
<th>Element..</th>
<th>Purpose..</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURRENT_GEN</td>
<td>Arbitrary Current Sources</td>
<td>The output is an arbitrary current source specified by the $I_{OUT}$ property. The value of $I_{OUT}$ can be set to any valid expression. For information on other properties, see BEHAV_GEN on page 470.</td>
</tr>
<tr>
<td>DCMOTOR</td>
<td>DC Motor</td>
<td>For detailed information on DC motors, see DC Motor on page 473.</td>
</tr>
<tr>
<td>DELAY</td>
<td>Delay buffer</td>
<td>The value of the DELAY property specifies the delay time. For example, if you set the value of DELAY property to 1m, a delay of 1 millisecond will be introduced in the output. Note: In circuits with digital feedback, it is recommended that some delay parameter must be included at some point in the signal path, by using DELAY or DELAY1. This is to avoid erroneous simulation results, caused due to zero propagation delay in functions without any delay parameters.</td>
</tr>
<tr>
<td>DELAY1</td>
<td>Delay buffer with inverted output</td>
<td>The value of the DELAY property specifies the delay time. For example, if you the input waveform is a sine wave, and the value of the DELAY property is set to 1m, the output waveform will be an inverted sine wave (180 degree phase shift) with a delay of 1 millisecond.</td>
</tr>
<tr>
<td>DIFFERENCE2</td>
<td>two input difference</td>
<td>Output is the difference between two inputs.</td>
</tr>
</tbody>
</table>
### Table 5-2 Elements in the Function library, continued

<table>
<thead>
<tr>
<th>Element..</th>
<th>Purpose..</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIV_BY_2</td>
<td>Frequency divider</td>
<td>Divides the frequency by 2. For example, if the frequency of the input clock signal is 60 Hz the frequency of the output signal would be 30 Hz.</td>
</tr>
<tr>
<td>DIV_BY_3</td>
<td>Frequency divider</td>
<td>Divides the frequency by 3. For example, if the frequency of the input clock signal is 60 Hz the frequency of the output signal would be 20 Hz.</td>
</tr>
<tr>
<td>DIV_BY_4</td>
<td>Frequency divider</td>
<td>Divides the frequency by 4. For example, if the frequency of the input clock signal is 60 Hz the frequency of the output signal would be 15 Hz.</td>
</tr>
<tr>
<td>DIV_BY_5</td>
<td>Frequency divider</td>
<td>Divides the frequency by 5. For example, if the frequency of the input clock signal is 60 Hz the frequency of the output signal would be 12 Hz.</td>
</tr>
</tbody>
</table>
Table 5-2 Elements in the Function library, continued

<table>
<thead>
<tr>
<th>Element..</th>
<th>Purpose..</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>DIVIDER2</td>
<td>Divider Block</td>
<td>Use this function to perform the mathematical function divide on the two inputs. You are not required to pass any parameters to this component and the output is ( \text{Input1} \div \text{Input2} ).</td>
</tr>
<tr>
<td>DXDT</td>
<td>Differentiator</td>
<td>See DIFFERENTIATOR on page 492.</td>
</tr>
<tr>
<td>EXP</td>
<td>Exponential Block</td>
<td>The output is calculated using the mathematical expression ( e^x ), where ( x ) is the input.</td>
</tr>
<tr>
<td>FLUX_GEN</td>
<td>Flux generator</td>
<td>See Flux source on page 176.</td>
</tr>
<tr>
<td>FLYWHEEL</td>
<td>Flywheel</td>
<td>See Flywheel on page 482.</td>
</tr>
<tr>
<td>FY1</td>
<td>First order transfer function</td>
<td>See Transfer Functions on page 493</td>
</tr>
<tr>
<td>FY2</td>
<td>Second order transfer function</td>
<td>See Transfer Functions on page 493</td>
</tr>
<tr>
<td>FY3</td>
<td>Third order transfer function</td>
<td>See Transfer Functions on page 493</td>
</tr>
<tr>
<td>FY4</td>
<td>Fourth order transfer function</td>
<td>See Transfer Functions on page 493</td>
</tr>
<tr>
<td>GAIN2</td>
<td>Defines gain</td>
<td>This is a laplace function. If the value of the GAIN property attached to the component is set to 0, an error message pops up indicating that it is a zero value component.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Similarly, if the value of GAIN property is not defined, then also PSpice throws a error message.</td>
</tr>
<tr>
<td>Element</td>
<td>Purpose</td>
<td>Comments</td>
</tr>
<tr>
<td>----------</td>
<td>--------------------------------------</td>
<td>--------------------------------------------------------------------------</td>
</tr>
<tr>
<td>GEARBOX</td>
<td>Is used as a torque converter</td>
<td>See Gearbox on page 483.</td>
</tr>
<tr>
<td>ILIM</td>
<td>Current limiter</td>
<td>See ILIM on page 489</td>
</tr>
<tr>
<td>IN</td>
<td>Input Comparator</td>
<td>See IN on page 484.</td>
</tr>
<tr>
<td>INTEGRATOR2</td>
<td>Integrator</td>
<td>See INTEGRATOR on page 491.</td>
</tr>
<tr>
<td>LNX</td>
<td>Natural log</td>
<td>Returns the logarithmic value of the input. Output is equal to $\ln(x)$, where $x$ is the input.</td>
</tr>
<tr>
<td>MAX</td>
<td>Maximum value block</td>
<td>Use this when you want the greater of the two input voltages as output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To get correct results, the difference between two input voltages should be greater than the value of the HYSTERESIS property. By default, the value of switch over hysteresis is one millivolt. For critical applications, you can reduce this value, but a smaller hysteresis might cause convergence problems in the time domain.</td>
</tr>
<tr>
<td>MIN</td>
<td>Minimum value block</td>
<td>Use this when you want the lesser of the two input voltages as output.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>As in the case of MAX, the difference between two input voltages should be greater than the value of the HYSTERESIS property. By default, the value of switch over hysteresis is one millivolt. For critical applications, you can reduce this value, but a smaller hysteresis might cause convergence problems in the time domain.</td>
</tr>
</tbody>
</table>
### Table 5-2 Elements in the Function library, continued

<table>
<thead>
<tr>
<th>Element..</th>
<th>Purpose..</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>MULTIPLIER2</td>
<td>Multiplier block</td>
<td>Performs the mathematical task of multiplying the two inputs, and returns the product as the output.</td>
</tr>
<tr>
<td>ONE_SHOT</td>
<td>Monostable multivibrator</td>
<td>See ONE_SHOT on page 486.</td>
</tr>
<tr>
<td>OUT</td>
<td>Output buffer</td>
<td>See OUT on page 485.</td>
</tr>
<tr>
<td>OUT1</td>
<td>Output buffer with inverted output</td>
<td>OUT1 is similar to OUT, except that it performs a logic inversion.</td>
</tr>
<tr>
<td>REALPOLE2</td>
<td>Models a single pole on the real axis</td>
<td>See REALPOLE2 on page 495.</td>
</tr>
<tr>
<td>REALZERO2</td>
<td>Models a single zero on the real axis</td>
<td>See REALZERO2 on page 496.</td>
</tr>
<tr>
<td>SLEW_LIMIT</td>
<td>Slew Rate Limiter</td>
<td>See SLEW_LIMIT on page 497.</td>
</tr>
<tr>
<td>SQRT</td>
<td>Returns square root of the input</td>
<td>Performs mathematical operation $\sqrt{x}$, where $x$ is the input. Negative values of input are not supported and an error message will be generated if $x&lt;0$.</td>
</tr>
<tr>
<td>SUM2</td>
<td>Two Input Summer</td>
<td>Adds two inputs values and returns the sum</td>
</tr>
<tr>
<td>TACHO</td>
<td>tachometer</td>
<td>For details, see Tachometer on page 476.</td>
</tr>
<tr>
<td>TIME</td>
<td>time function</td>
<td>The output of the Time function is equal to $t$ times the input. For the Time function to work properly, simulation time must be less than 1000 seconds.</td>
</tr>
<tr>
<td>Element..</td>
<td>Purpose..</td>
<td>Comments</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>VCO</td>
<td>Macro Logic Voltage Controlled Oscillator</td>
<td>See VCO on page 487.</td>
</tr>
<tr>
<td>VISCOSITY</td>
<td>Use this component to simulate the viscous friction of a fluid.</td>
<td>The VISC property specifies the viscosity, which is the measure of the resistance of a fluid to deformation. It is measured in Newton-meter-seconds.</td>
</tr>
<tr>
<td>VLIM</td>
<td>Voltage limiter</td>
<td>See VLIM on page 490</td>
</tr>
<tr>
<td>VOLTAGE_FREQ</td>
<td>Frequency Domain Voltage Source</td>
<td>It is a two-pin voltage source that defines the voltage in terms of an expression and a transfer function. For information on the component properties, see BEHAV_FREQ on page 469.</td>
</tr>
<tr>
<td>VOLTAGE_GEN</td>
<td>Arbitrary Voltage Sources</td>
<td>It is a two-pin arbitrary voltage source. For information on the component properties, see BEHAV_GEN on page 470.</td>
</tr>
</tbody>
</table>
**Important**

The symbol ( ) denotes analog functionality. Terminals labeled with these symbols are the only terminals that can be connected to external circuitry.

**Table 5-3 Analog Switch Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
<th>Measured in</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROFF</td>
<td>Off resistance</td>
<td>ohms (Ω)</td>
</tr>
<tr>
<td></td>
<td>Specifies the value of the resistor when the digital input is low.</td>
<td>Default value is 1e12 ohms.</td>
</tr>
<tr>
<td>RON</td>
<td>On resistance</td>
<td>ohms (Ω)</td>
</tr>
<tr>
<td></td>
<td>Defines the value of the resistor when the digital input is high</td>
<td>Default value is set to 1 ohms.</td>
</tr>
</tbody>
</table>

**Comment**

If you use the ASW part to simulate the analog switch, the switch will be ON if the input at pin B is set to 1.

The resistance of the switch, when ON is determined by the value of RON property. Similarly, the value of the ROFF property determines the resistance of the switch in the OFF state.
ASW1

**Purpose**  
Analog switch

**Comment**  
If you use the ASW1 part to simulate the analog switch, the switch will be ON if the digital input at pin B is set to 0.

The resistance of the switch, when ON is determined by the value of RON property. Similarly, the value of the ROFF property determines the resistance of the switch in the OFF state.

For more information, see ASW on page 467.
**BEHAV_FREQ**

Conceptually, this device can be represented in two parts, a general source followed by a transfer function. The transfer function is defined in frequency domain terms.

$$H(s) = \exp(As + B) \times \frac{NUM}{DEN}$$

### Table 5-4 Behavioral Source Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOUT</td>
<td>Expression</td>
</tr>
<tr>
<td>EXP</td>
<td>Delay and decay of the transfer function, (\exp(As+B)).</td>
</tr>
<tr>
<td>NUM</td>
<td>Numerator of the transfer function as a polynomial of (s).</td>
</tr>
<tr>
<td>DEN</td>
<td>Denominator of the transfer function as a polynomial of (s).</td>
</tr>
</tbody>
</table>

You can use these sources both in time and frequency domain.

**Example**

A transfer function with a gain of two and a single pole at 1 KHz can be implemented with the following expressions:

- \(VOUT = 2 \times v(in)\)
- \(EXP = UNDEF \) (optional)
- \(NUM = 2\pi \times 1e3\)
- \(DEN = s + 2\pi \times 1e3\)
BEHAV_GEN

Description: Frequency domain arbitrary behavioral voltage source

Comment: This is a single pin arbitrary voltage source, where the output voltage is defined as an expression.

Table 5-5 Arbitrary Voltage Source Properties

<table>
<thead>
<tr>
<th>Property..</th>
<th>Meaning..</th>
</tr>
</thead>
<tbody>
<tr>
<td>VOUT</td>
<td>Specifies the output voltage in form of a valid expression. The expression can have arithmetic operators, variables, and nested functions.</td>
</tr>
<tr>
<td>ERROR_COND</td>
<td>Specifies a condition, which when TRUE will stop the simulation process.</td>
</tr>
<tr>
<td>ERROR_MESG</td>
<td>Specifies the message statement displayed to a user when ERROR_COND is met. This message will be displayed only if an error condition has been specified.</td>
</tr>
<tr>
<td>WARN_COND</td>
<td>Specifies a condition, which when TRUE will throw the warning message.</td>
</tr>
<tr>
<td>WARN_MESG</td>
<td>Specifies the message statement displayed to a user in case of warning. The message statement will be displayed only if a warning condition has been specified.</td>
</tr>
</tbody>
</table>

**Note:** The error and warning properties are optional.
Coil Spring

A coil wound in a spiral shape that reacts against twisting motion.

**General form**

```
X AWBCOILSPRING PARAMS: IC=<value> COILVAL=<value>
```

The symbol and properties for a coil spring are listed below.

**Figure 5-1  Coil Spring**

![Coil Spring Diagram]

Value = 1G

positive torque flow

**Table 5-6  Coil Spring Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
<th>Measured in</th>
</tr>
</thead>
<tbody>
<tr>
<td>IC</td>
<td>Current through the coil</td>
<td>Ampere (A)</td>
</tr>
<tr>
<td>COILVALUE</td>
<td>Spring constant</td>
<td>Newton-meters</td>
</tr>
<tr>
<td></td>
<td>The default value is 1M</td>
<td></td>
</tr>
</tbody>
</table>

The IC property represents the initial current through the inductor during the bias point calculation. A positive initial torque can be measured as a current flowing from left to right.
CURRENT_FREQ

Purpose: Frequency-defined behavioral current source

Comment: This function defines the output current as a function of input voltage and an expression of \( s \). You can use this source either in time or frequency domain.

\[ I(\text{out}) = H(s) \times V(\text{in}) \]

The transfer function is defined in frequency domain terms.

\[ H(s) = \exp(As + B) \times \frac{NUM}{DEN} \]

For the explanation of properties attached to CURRENT_FREQ, see Table 5-3 on page 467.

Example

A transfer function with a gain of 3 and a single pole at 1 KHz can be implemented with the following expressions:

\[ IOUT = 3 \times v(\text{in}) \]
\[ EXP = 0 \]
\[ NUM = 2 \times 3.142 \times 1K \]
\[ DEN = s + 2 \times 3.142 \times 1K \]
DC Motor

A DC motor is used to convert electrical energy to mechanical energy. It works on the principle that when electric current passes through a magnetic field, a torque is produced because of the magnetic force. This torque is used to run the DC motor.

**General form**

```
X AWBDCMOTOR PARAMS: R=<value> L=<value> BACK_EMF=<value> 
    + K_TORQ=<value> K_VISC=<value> 
    + INERTIA=<value> 
    + CONST_FRIC=<value>
```

R = 0
L = 1M
BACK_EMF = 1
K_TORQ = 1
K_VISC = 0
INERTIA = 1M
CONST_FRIC = 0

The electrical characteristics of the motor, synthetically winding, is modeled as a series RL circuit representing the armature inductance and resistance. Mechanical portion of motor is modeled as a parallel RC circuit. The electrical and mechanical portions are connected using controlled sources. Back EMF is also modeled and included in series with motor winding in such a manner that it opposes the input voltage. Voltage at pin C represents the Motor Torque.

Table 5-7 on page 473 lists properties attached to a DC motor symbol.

**Table 5-7 DC Motor Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Meaning</th>
<th>Measured in</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Series winding resistance</td>
<td>ohms (Ω)</td>
</tr>
<tr>
<td>L</td>
<td>Series inductance</td>
<td>Henries</td>
</tr>
</tbody>
</table>
Table 5-7  DC Motor Properties, continued

<table>
<thead>
<tr>
<th>Property..</th>
<th>Meaning..</th>
<th>Measured in..</th>
</tr>
</thead>
<tbody>
<tr>
<td>BACK_EMF</td>
<td>Back electromagnetic field constant</td>
<td>volt-sec/rad</td>
</tr>
<tr>
<td>K_TORQ</td>
<td>Torque constant</td>
<td>N-m/amp</td>
</tr>
<tr>
<td>K_VISC</td>
<td>Viscous frictional coefficient</td>
<td>N-m-sec/rad</td>
</tr>
<tr>
<td>INERTIA</td>
<td>Rotor moment of inertia</td>
<td>N-m-sec^2/rad</td>
</tr>
<tr>
<td>CONST_FRIC</td>
<td>Constant friction torque</td>
<td>N-m</td>
</tr>
</tbody>
</table>

Equations:

**Motor Current:**

\[
V_{ip} - V_{backemf} = L_i \frac{di}{dt} + R_i
\]

In this equation:

- \( V_{backemf} = BACK\_EMF \times (Motor\ speed) \)
- \( V_{ip} \) is Voltage applied to motor input terminals (A,B)
- \( L_i \) is winding inductance
- \( R_i \) is winding resistance

**Torque:**

Torque is modeled using the following equation.

**Note:** Solving the equation for voltage \((V_t)\) gives the equivalent torque.

\[
I = V_t/R_v + (C \cdot dV_t)/(dt)
\]

In this equation:

- \( C \) represents inertia
- \( I \) represents torque
- \( V_t \) represents angular velocity or speed; The angular velocity is \( K\_TORQ/R \) times of voltage applied across motor winding.
- \( R_v \) represents friction, which is \( 1/(K\_VISC) \)

The back emf is a function of param \( \{\text{back}\_emf\} \) and generated back EMF is a function of \( \{\text{velocity of motor}\} \times \{\text{back}\_emf\} \).
Generated torque is function of angular velocity (voltage drop) across the winding resistance (winding current). The motor torque equation is:

\[ T = J \frac{\mathrm{d} \omega}{\mathrm{d} t} + W.B \]

In this equation:

- \( T \) represents torque
- \( J \) represents inertia
- \( \omega \) is angular velocity
- \( B \) is friction
Tachometer

A tachometer is a device for indicating the angular (rotary) speed of a rotating shaft. It indicates the instantaneous values of speed in revolutions per minute (RPM). A tachometer converts mechanical energy into electrical energy.

**General form**

```plaintext
X AWBTACHO PARAMS: R=<value> L=<value> + BACK_EMF=<value> K_TORQ=<value> + K_VISC=<value> INERTIA=<value> + CONST_FRIC=<value>
```

R = 0
L = 1M
BACK_EMF = 1
K_TORQ = 1
K_VISC = 0
INERTIA = 1M
CONST_FRIC = 0

---

Table 5-8 on page 476 lists properties of a tachometer.

**Table 5-8 Tachometer Properties**

<table>
<thead>
<tr>
<th>Property..</th>
<th>Meaning..</th>
<th>Measured in..</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>Series winding resistance</td>
<td>ohms (Ω)</td>
</tr>
<tr>
<td>L</td>
<td>Series inductance</td>
<td>Henries</td>
</tr>
<tr>
<td>BACK_EMF</td>
<td>Back electromagnetic field voltage</td>
<td>volt-sec/rad</td>
</tr>
<tr>
<td>K_TORQ</td>
<td>Torque constant</td>
<td>N-m/amp</td>
</tr>
<tr>
<td>K_VISC</td>
<td>Viscous frictional coefficient</td>
<td>N-m-sec/rad</td>
</tr>
<tr>
<td>INERTIA</td>
<td>Rotor moment of inertia</td>
<td>N-m-sec²/rad</td>
</tr>
<tr>
<td>CONST_FRIC</td>
<td>Constant friction torque</td>
<td>N-m</td>
</tr>
</tbody>
</table>
COMPLEX_FZ

BP_GAIN = ?
HP_GAIN = ?
FREQ_HZ = ?
LP_GAIN = ?
DAMPING = ?

Purpose

Returns a second-order function.

The frequency and damping complex-pole function block is a general-purpose, second-order block that can form linear combinations of low-pass, band-pass, and high-pass functions.

Arguments

\[ FREQ\_HZ \]

Used to calculate the natural frequency, Omega (\( \omega \)) using the equation, \( \omega = 2\pi FREQ\_HZ \).

Omega is used to set the cutoff frequency for the high-pass and low-pass functions and the band center frequency for the bandpass function.

\[ DAMPING \]

This is the damping factor Zeta (\( \zeta \)). At values of \( \zeta \) below 1, the poles or zeros separate from the x-axis and form complex pole or zero pairs, leading, respectively, to ringing or peaking in the frequency domain.

\[ BP\_GAIN \]

Specifies the gain for the band pass function.

\[ HP\_GAIN \]

Specifies the gain for the high pass function.

\[ LP\_GAIN \]

Specifies the gain for the low pass function.
Determining the Low-Pass Coefficient

To determine the low-pass coefficient, $C$ (assuming that $A = B = 0$), you first decide what DC gain you want and then take the limit of the transfer function as $s$ goes to zero. The variable, $s$, is the zero-frequency or DC gain point. This equation simplifies to

$$C = \text{the desired DC gain}$$
Determining the High-Pass Coefficient

To determine the high-pass coefficient, $A$ (assuming that $B = C = 0$), you must first decide what gain you want to have at a frequency high above the cutoff frequency and then take the limit of the transfer function as $s$ goes to infinity, which is the high-frequency asymptote.

Solving this equation, you will notice that the square of infinity is involved in both the numerator and the denominator. This simplifies both terms and leads to cancellation of $j$-squared and infinity-squared. This calculation produces the following result

$$A \omega^2 = \text{the desired high-frequency gain}$$

Therefore, $A$ equals the desired gain divided by the square of $\omega$. This implies that to maintain a constant gain, $A$ must change as the natural frequency changes.

Determining the Bandpass Coefficient

To determine the band-pass coefficient, $B$ (assuming that $A = C = 0$), you first decide what gain you want at the center of the pass band. Then, you take the limit of the transfer function as $\omega$ goes to $\omega$.

Note: Remember that the natural frequency for a bandpass function is the center frequency.

With $A$ and $C$ at zero, the equation becomes

$$\frac{0j^2 \omega^2 + Bj \omega + 0}{\frac{1}{\omega^2} + \frac{2\zeta}{\omega} + 1}$$

In the denominator, the first term goes to -1 and the second term goes to $2\zeta j$.

Therefore, bandpass gain equals $Bj\omega/2j\zeta$

Cancelling the $js$ and solving for $B$, you get

$$B = \frac{2\zeta \text{(desired band-pass gain)}}{\omega}$$

The value required for $A$ depends on the natural frequency and the damping factor of the circuit. So, to maintain a constant gain when the natural frequency or $\zeta$ changes, the value of the coefficient $B$ changes.
**COMPLEX_RI**

**Description**  Complex pole, real and imaginary

**Purpose**  Using the Real and Imaginary complex pole function block you define the complex poles based on real and imaginary parameters of a second order equation.

**Arguments**

- `REAL_HZ`  
  Real part of the complex pole.

- `IMAG_HZ`  
  This is the imaginary part of the complex pole.

- `BP_GAIN`  
  Specifies the gain for the band pass function.

- `HP_GAIN`  
  Specifies the gain for the high pass function.

- `LP_GAIN`  
  Specifies the gain for the low pass function.
Comments

This is also a Laplace domain function, similar to the frequency and damping complex pole function block, COMPLEX_FZ on page 477.

Using complex-pole, real and imaginary function block, you calculate complex poles based on real and imaginary parameters of the following second order equation.

\[
\frac{\omega (A s^2 + B s + C)}{(s + 2\pi D + 2j\pi E)(s + 2\pi D - 2j\pi E)}
\]

where

- A is specified by HP_GAIN
- B is specified by BP_GAIN,
- C is specified by LP_GAIN,
- D is specified by REAL_HZ,
- and
- E is specified by IMG_HZ.
Flywheel

Use this component to simulate the flywheel effect in a PSpice simulation. A flywheel is a heavy rotating disk on a shaft and resists changes in the rotation speed.

![Diagram of Flywheel]

**INERTIA = 1**

**FLYWHEEL**

**General form**

\[ X \text{ AWFLYWHEEL PARAMS: INERTIA=<value>} \]

\[ \text{energy} = \frac{1}{2} I \omega^2 \]

For a flywheel torque is measured by the equation given below.

\[ T = J \times \frac{d\omega}{dt} \]

where:

- **T** torque
  Specified by the current

- **J** Moment of Inertia
  Specified by INERTIA

- **\( \omega \)** Angular velocity
  specified by circuit voltage

**Important**

The mechanical elements in the FUNCTION library are simulated in PSpice using a mechanical-to-electrical analogy. The mechanical properties such as torque and angular velocity are represented by current and voltage, respectively. They are calibrated in a manner such that one **volt** corresponds to one **radian per second** of shaft velocity, and one **Ampere current** is equal to one **Newton-meter** of torque.
Gearbox

Use this component to simulate a mechanical gear box in PSpice. A gearbox is an assembly of gears that allows the rotational speed of an input shaft to be changed to a different speed.

\[ \text{RATIO} = 2 \]

\[ \begin{align*} \text{GEARBOX} \end{align*} \]

The conversion is done using the equation given below.

\[ TL = \left( \frac{N_2}{N_1} \right) \cdot TM \]

where

\begin{align*} 
TL & \quad \text{Torque at load shaft} \\
N_2/N_1 & \quad \text{gear ratio specified by the RATIO property} \\
TM & \quad \text{Torque at motor shaft} 
\end{align*}

The value of the RATIO property can be obtained using one of the following ratios:

\begin{align*} 
\text{output diameter} & \quad \text{output number of teeth} \\
\text{input diameter} & \quad \text{input number of teeth} 
\end{align*}

**Important**

The mechanical elements in the FUNCTION library are simulated in PSpice using a mechanical-to-electrical analogy. The mechanical properties such as torque and angular velocity are represented by current and voltage, respectively. They are calibrated in a manner such that one volt corresponds to one radian per second of shaft velocity, and one Ampere current is equal to one Newton-meter of torque.
IN

This input comparator converts analog voltages to 0 and 1 volt levels that are compatible with the digital inputs of other functions.

\[
\begin{align*}
\text{THR} &= 2 \\
\text{HYS} &= 1.5
\end{align*}
\]

General form

\[
X \text{ AWBIN PARAMS: THR=<value> HYS=<value>}
\]

THR

Specifies threshold

HYS

Specifies hysteresis value

Note: The properties for the IN function correspond to those for the voltage-controlled switch; the ON state in the voltage-controlled switch maps to 1 Volt at the output, and the OFF state maps to 0 Volts. The open and closed resistance values for the IN function are fixed at 1 Ohm and 1 MegOhm, respectively. In most cases you can connect an external reference voltage to one of the inputs, and leave the default properties unchanged.
OUT

OUT is an output buffer. It buffers the digital outputs of other functions so they can be connected to external (analog) circuitry.

```
ROUT = 1K
VLO = 10m
VHI = 3.3
```

**General form**

`X AWBOUT1 PARAMS: ROUT=<value> VLO=<value> VHI=<value>`

- **ROUTE**: Output impedance in ohms
- **VLO**: Specifies the voltage level in volts, for the low output
- **VHI**: Specifies the voltage level in volts, for the high output

**Note**: The reference terminal, located at the bottom of the OUT or OUT1 symbol, must be connected to the reference voltage of the external circuitry. This is usually (but not necessarily) ground. To use OUT1 as a logic inverter, use the default property values and connect the reference terminal to ground.
ONE_SHOT

This is a monostable multivibrator.

![ONE_SHOT schematic]

**General form**

\[ X \text{ AWBONE_SHOT PARAMS: DELAY=<value>} \]

**DELAY**

Specifies the maximum pulse width of the output waveform

A monostable multivibrator has two inputs: a clock input labeled >, and a reset input labeled \( R \). The \( R \) input is used to initialize and reset the output. When the \( R \) input is high, the output is forced low. With \( R \) low, a positive-going edge at the clock input causes the output to step from low to high. The pulse is terminated by a high at the \( R \) input.

These relationships are summarized in the truth table shown below.

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock (&gt;)</td>
<td>Reset (R)</td>
</tr>
<tr>
<td>X</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

**Caution**

*If the output terminal of the monostable multivibrator is loaded with a finite impedance, it malfunctions.*
VCO

This is a *Square Wave* Voltage Controlled Oscillator that has a differential analog control voltage input that controls the frequency of the oscillator. The linear frequency transfer characteristic is determined by four parameters that are specified with the properties A Voltage ($A_V$), A Frequency ($A_F$), B Voltage ($B_V$), and B Frequency ($B_F$).

\[ \frac{F_{OUT}}{B_V - A_V} \frac{B_V \times A_F - (A_V \times B_F)}{(B_F - A_F) \times V_{IN}} \]

**Parameters**

- $A_V = \?$
- $A_F = \?$
- $B_V = \?$
- $B_F = \?$

**Note:** Positive and negative terminals are, respectively, represented as A and B.

- $A_F$ specifies A frequency
- $A_V$ specifies A voltage
- $B_F$ specifies B frequency
- $B_V$ specifies B voltage

**Caution**

*A frequency and B frequency must be greater than zero but not equal to 1. And A voltage must not equal B voltage.*
Where,

- $F_{OUT}$: Output Frequency of the VCO
- $V_{IN}$: Input Voltage \([V_{IN1} - V_{IN2}]\)

**Note:** The Square Wave VCO malfunctions if its output terminal is loaded with a finite impedance.
ILIM

This current limiter limits the output current within the range specified by the user.

IU  = 1
IL  = -1
RS  = 1M

![Diagram of ILIM model]

**Parameters**

**IU**

Specifies the upper limit of the output current. If the input current is more than the value of the IU parameter, the input current is clipped at this value.

**IL**

Specifies the lower limit of the output current. If the input current is less than IL, the output current is maintained at IL.

**RS**

Specifies device resistance

**Note:** The input current depends on the input voltage $V_{in}$, and the device resistance $RS$. The input current $I_{in}$ is calculated using the equation given below:

$$I_{in} = \frac{V_{in}}{RS}$$
VLIM

This is a voltage limiter that is used to maintain the output voltage within the range specified by the user.

\[ V_U = 5 \]
\[ V_L = 2 \]

Parameters

\( V_U \)

Specifies the upper limit of the output voltage. If the input voltage is more than the value of the \( V_U \) parameter, the output voltage is clipped at this value.

\( V_L \)

Specifies the lower limit of the output voltage. If the input voltage is less than \( V_L \), the output voltage is maintained at \( V_L \).
INTEGRATOR

The Integrator block models the transfer function \( \frac{k}{s} \), but with a finite DC gain. For a gain of 1, the DC gain is 240 dB, but as the gain is changed, the DC gain varies. This can affect DC convergence if the gain is set too high.

**Figure 5-2 Integrator Function**

The integrator has an ideal buffered output on the right side of the symbol and a special initial condition (IC) pin on top. The GAIN property affects the unity gain frequency. For instance, to obtain a unity gain point of 1 Hz, enter a value of 6.28 (~2\(\pi\)).

The rise time of the signal fed through an Integrator function must not be less than 0.1% of the total simulation time. If a DC voltage is applied to the input of the Integrator block and the output wire is not connected, you must assign initial conditions to the IC pin.

To set an initial condition, you can either use an IC part or a NODESET, by connecting it to the IC pin on the top of the symbol. Without the initial conditions, the output attempts to reach an infinite voltage.
DIFFERENTIATOR

Differentiator output is calculated using the equation given below.

\[ output = GAIN \times \frac{d}{dt} \text{input} \]
Transfer Functions

A transfer function between an input variable, \( u(t) \), and an output variable, \( y(t) \), of a system is defined as the ratio of the Laplace transform of the output to the Laplace transform of the input:

\[
U(s) = \text{Laplace}\{u(t)\}
\]
\[
Y(s) = \text{Laplace}\{y(t)\}
\]
\[
H(s) = \text{transfer function} = \frac{Y(s)}{U(s)}
\]

In these equations, \( s \) is the complex variable.

The four types of transfer functions supported in PSpice are FY1, FY2, FY3, and FY4, shown in Figure 5-3 on page 493.

**Figure 5-3 Transfer Functions**

\[
H(s) = \frac{B1s + B0}{s + A0}
\]

FY1  
1\(^{st}\) order function

\[
H(s) = \frac{B2s^2 + B1s + B0}{s^2 + A1s + A0}
\]

FY2  
2\(^{nd}\) order function

\[
H(s) = \frac{B3s^3 + B2s^2 + B1s + B0}{s^3 + A2s^2 + A1s + A0}
\]

FY3  
3\(^{rd}\) order function

\[
H(s) = \frac{B4s^4 + B3s^3 + B2s^2 + B1s + B0}{s^4 + A3s^3 + A2s^2 + A1s + A0}
\]

FY4  
4\(^{th}\) order function
The following rules apply to transfer functions:

- Transfer functions are defined only for linear, time-invariant systems.
- The transfer function model assumes that all initial conditions are zero.
- Transfer functions are independent of input excitation.

The equations for the following fourth-order transfer function are

\[
Y(s) = B_4 s^4 + B_3 s^3 + B_2 s^2 + B_1 s + B_0
\]

\[
U(s) = s^4 + A_3 s^3 + A_2 s^2 + A_1 s + A_0
\]

where

- \( Y(s) \) Output
- \( U(s) \) Input
- \( S \) The complex frequency \((s + jw)\)
- \( B_4, B_3, B_2, B_1, B_0, A_3, A_2, A_1, \text{ and } A_0 \) Constant coefficients
REALPOLE2

The Real Pole function models a single pole on the real axis. The frequency response has constant gain from DC to about one decade below the pole frequency, by which point the gain rolls off to 3 dB below the DC gain. Above the pole, the frequency response rolls off at about 20 dB per decade.

Figure 5-4 Symbol and Properties for the Real Pole Function

\[ \text{Transfer Function} = \frac{k}{1 + \frac{s}{\omega}} \]

\[ f = \frac{\omega}{2\pi} \]

The following are the properties for this function.

**General form**

\[ X \text{ AWBREALPOLE2 PARAMS: FREQ_HZ=<value> GAIN=<value>} \]

**FREQ_HZ**

Specifies operating frequency in Hertz (Hz). This value cannot be negative.

**GAIN**

Specifies the linear gain factor. Fractional numbers provide attenuation, while negative numbers provide phase reversal with respect to a positive gain factor.
REALZERO2

The Real Zero function models a single zero on the real axis. The frequency response has constant gain from DC to about one decade below the zero frequency, at which point the gain increases. Above zero, the frequency response increases at 20 dB per decade.

To avoid convergence problems in time domain analyses, use the Real Pole function block (with pole frequency equal to 1000 times the zero frequency) in series with the Real Zero function block.

Figure 5-5 Symbol and Properties for the Real Zero Function

General form

\[
\begin{align*}
\text{FREQ}_\text{HZ} &= \omega / 2\pi \\
\text{GAIN} &= 1
\end{align*}
\]

X AWBREALPOLE2 PARAMS: FREQ_HZ=<value> GAIN=<value>

FREQ_HZ

Specifies operating frequency in Hertz (Hz). For zero location, frequency can be negative. This is unlike the frequency value for a Real Pole.

GAIN

Specifies the linear gain. Fractional numbers provide attenuation, while negative numbers provide phase reversal with respect to a positive gain factor.
SLEW_LIMIT

Use this part to control the rise and fall value of the input waveforms.

General form

X AUWSLEW_LIMIT PARAMS: POS_SLEW=<value> NEG_SLEW=<value>

POS_SLEW

Specifies the positive slew rate, which control the rise time of the input waveform. The rise time, $t_r$, is equal to the reciprocal of POS_SLEW.

NEG_SLEW

Specifies the negative slew rate, which control the fall time of the input waveform. The fall time, $t_f$, is equal to the reciprocal of NEG_SLEW.

$$t_r = \frac{1}{POSSLEW} \quad \text{and} \quad t_f = \frac{1}{NEGSLEW}$$
### Spice_elem library

<table>
<thead>
<tr>
<th>Element..</th>
<th>Purpose..</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC_SWITCH</td>
<td>Current controlled switch</td>
<td>See CC_SWITCH on page 500.</td>
</tr>
<tr>
<td>CCCS</td>
<td>Current Controlled Current Source</td>
<td>See CCCS on page 511, and Current-controlled current source Current-controlled voltage source on page 174.</td>
</tr>
<tr>
<td>CCS10</td>
<td>Controlled Current Source</td>
<td>This is a dependent current source. Output current can be controlled either by the voltagesense or by the currentsense. For more information see, CCS10 on page 505.</td>
</tr>
<tr>
<td>CCS23</td>
<td>Controlled Current Source</td>
<td>This is also a dependent current source, where the output current is controlled both, by the voltagesense and by the currentsense. For more information see, CCS23 on page 508.</td>
</tr>
<tr>
<td>CCVS</td>
<td>Current Controlled Voltage Source</td>
<td>See CCCS on page 511 and Current-controlled current source Current-controlled voltage source on page 174.</td>
</tr>
<tr>
<td>CURRENT</td>
<td>Independent Current source</td>
<td>The output current is determined by the value assigned to the VALUE property attached to the symbol.</td>
</tr>
<tr>
<td>CURRENTSENSE</td>
<td>Ammeter</td>
<td>See Currentsense on page 509.</td>
</tr>
<tr>
<td>CVS10</td>
<td>Controlled Voltage Source</td>
<td>This is a dependent voltage source, where the output voltage is controlled either by the voltagesense or by the currentsense. For more information see, CVS10 on page 502.</td>
</tr>
<tr>
<td>CVS23</td>
<td>Controlled Voltage Source</td>
<td>This is also a dependent voltage source, where the output voltage is controlled by both, the voltagesense and the currentsense. For more information see, CVS23 on page 507.</td>
</tr>
</tbody>
</table>
Table 5-9 Elements in the Spice_elem library, *continued*

<table>
<thead>
<tr>
<th>Element</th>
<th>Purpose</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC</td>
<td>DC transformer</td>
<td>See DC transformer on page 516.</td>
</tr>
<tr>
<td>DELAY_2_TERM</td>
<td>Delay Line</td>
<td>See Delay lines on page 515.</td>
</tr>
<tr>
<td>DELAY_3_TERM</td>
<td>Delay Line</td>
<td>See Delay lines on page 515.</td>
</tr>
<tr>
<td>VC_CAP</td>
<td>Voltage Controlled Capacitor</td>
<td></td>
</tr>
<tr>
<td>VC_CON</td>
<td>Voltage Controlled Conductor</td>
<td></td>
</tr>
<tr>
<td>VC_IND</td>
<td>Voltage Controlled Inductor</td>
<td></td>
</tr>
<tr>
<td>VC_RES</td>
<td>Voltage Controlled Resistor</td>
<td></td>
</tr>
<tr>
<td>VC_SWITCH</td>
<td>Voltage Controlled Switch</td>
<td>See CC_SWITCH on page 500.</td>
</tr>
<tr>
<td>VCCS</td>
<td>Voltage Controlled Current Source</td>
<td>See CCCS on page 511.</td>
</tr>
<tr>
<td>VCVS</td>
<td>Voltage Controlled Voltage Source</td>
<td>See CCCS on page 511.</td>
</tr>
<tr>
<td>VOLTAGESENSE</td>
<td>Voltmeter</td>
<td>See Voltagesense on page 510.</td>
</tr>
</tbody>
</table>
CC_SWITCH

Model for current controlled switches.

\[
\begin{align*}
\text{THRESHOLD} &= \text{?} \\
\text{HYSTERESIS} &= \text{?} \\
\text{OFF\_RESIST} &= \text{?} \\
\text{ON\_RESIST} &= \text{?}
\end{align*}
\]

The properties passed as parameters to a switch model are shown in the table below.

**Table 5-10 CC_SWITCH properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>indicates...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold</td>
<td>Threshold current</td>
</tr>
<tr>
<td>Hysteresis</td>
<td>Hysteresis current</td>
</tr>
<tr>
<td>On resistance</td>
<td>Resistance when the switch on</td>
</tr>
<tr>
<td>Off resistance</td>
<td>Resistance when the switch off</td>
</tr>
<tr>
<td></td>
<td>This is of the order of 10^6 ohms.</td>
</tr>
<tr>
<td>Delay</td>
<td>time delay in seconds</td>
</tr>
</tbody>
</table>

Use of switches can cause large discontinuities to occur in the circuit node voltages and branch currents. A rapid change such as that associated with a switch changing state can cause numerical round off or tolerance problems, leading to erroneous results or time step difficulties. You can improve the situation by taking the following actions:

- Set ideal switch impedances only high and low enough to be negligible with respect to other circuit elements. Using switch impedances that are close to “ideal” in all cases aggravates the discontinuity problem. When modeling real devices such as MOSFETs, adjust the on resistance to a realistic level, depending on the size of the device being modeled.

- Set a delay not equal to 0.
**VV_SWITCH**

Model for voltage controlled switches.

```
THRESHOLD = ?
HYSTERESIS = ?
OFF_RESIST = ?
ON_RESIST = ?
```

The properties passed as parameters to VC_SWITCH model are shown in the table below.

**Table 5-11 VC_SWITCH Properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>indicates...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold</td>
<td>Threshold voltage</td>
</tr>
<tr>
<td>Hysteresis</td>
<td>Hysteresis voltage</td>
</tr>
<tr>
<td>On resistance</td>
<td>Resistance when the switch on</td>
</tr>
<tr>
<td>Off resistance</td>
<td>Resistance when the switch off</td>
</tr>
<tr>
<td>Delay</td>
<td>time delay in seconds</td>
</tr>
</tbody>
</table>
CVS10

This is a controlled voltage source. Here the output voltage is controlled either by a Currentsense or by a Voltagesense. To see how CVS10 is used with a currentsense, see Figure 5-6 on page 503. To see how to use CVS10 with a voltage sense, see Figure 5-7 on page 504.

Property

GAIN

Describes the voltage gain

VSOURCE

If the value of this parameter is set to TRUE, it indicates that the voltage source is being controlled by a voltagesense. If no value is assigned to this parameter, it indicates that the output voltage is controlled by a currentsense.

CONTROLLER

Specifies the name of the controller.
Figure 5-6  CVS10 used with CurrentSense

Table 5-12  Property values when CVS10 is controlled by a CurrentSense

<table>
<thead>
<tr>
<th>Property</th>
<th>Value assigned...</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSOURCE</td>
<td>NULL</td>
</tr>
<tr>
<td>CONTROLLER</td>
<td>IX1 (Name of the CurrentSense)</td>
</tr>
<tr>
<td>GAIN</td>
<td>2</td>
</tr>
</tbody>
</table>
Figure 5-7  CVS10 used with a VoltageSense

Table 5-13  Property values when CVS10 is controlled by a VoltageSense

<table>
<thead>
<tr>
<th>Property</th>
<th>Value assigned...</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSOURCE</td>
<td>TRUE</td>
</tr>
<tr>
<td>CONTROLLER</td>
<td>VX1 (Name of the VoltageSense)</td>
</tr>
<tr>
<td>GAIN</td>
<td>2</td>
</tr>
</tbody>
</table>
CCS10

This is a controlled current source. Here the output current is controlled either by a Currentsense, see Figure 5-9 on page 506, or by a Voltagesense, Figure 5-8 on page 505.

**Property**

Value<expression>

Indicates the gain of the current source, which is assigned using the GAIN property attached to the symbol.

**VSOURCE**

If the value of this parameter attached to the symbol, is set to TRUE, it indicates that the output current is being controlled by a voltagesense. If no value is assigned to this parameter, it indicates that the controller is a currentsense.

**Figure 5-8  CCS10 used with a VoltageSense**

![Diagram of CCS10 used with a VoltageSense](image)

**Table 5-14  Property values when CCS10 is controlled by a VoltageSense**

<table>
<thead>
<tr>
<th>Property</th>
<th>Value assigned...</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSOURCE</td>
<td>TRUE</td>
</tr>
<tr>
<td>CONTROLLER</td>
<td>VX1(Name of the VoltageSense)</td>
</tr>
<tr>
<td>GAIN</td>
<td>2</td>
</tr>
</tbody>
</table>
Figure 5-9  CCS10 used with a CurrentSense

![Diagram of CCS10 used with a CurrentSense](image)

Table 5-15  Property values when CCS10 is controlled by a VoltageSense

<table>
<thead>
<tr>
<th>Property</th>
<th>Value assigned...</th>
</tr>
</thead>
<tbody>
<tr>
<td>VSOURCE</td>
<td>NULL</td>
</tr>
<tr>
<td>CONTROLLER</td>
<td>IX1 (Name of the CurrentSense)</td>
</tr>
<tr>
<td>GAIN</td>
<td>2</td>
</tr>
</tbody>
</table>
CVS23

This is a controlled voltage source, controlled by two voltagessenses or currentsenses.

\[ \text{VX1} \]
\[ \text{VY1} \]
\[ \text{GAIN} = ? \]
\[ \text{CVS23} \]

**Property**

\[ \text{VALUE}<\text{expression}> \]

Indicates the expression used to calculate the output current. The expression used is:

\[ \text{GAIN} \times (C_0 + C_1 \cdot I_1 + C_2 \cdot I_2 + C_3 \cdot I_1^2 + C_4 \cdot I_1 \cdot I_2 + C_5 \cdot I_2^2 + C_6 \cdot I_1^3 + C_7 \cdot I_1^2 \cdot I_2 + C_8 \cdot I_1 \cdot I_2^2 + C_9 \cdot I_2^3) \]

where

- GAIN is the value of the GAIN property
- \( C_0 \) is the value of the \( C0\_VALUE \) property
- \( C_1 \) is the value of the \( C1\_VALUE \) property
- \( C_2 \) is the value of the \( C2\_VALUE \) property
- and so on

**VSOURCE**

If the value \text{VSOURCE} is set to TRUE, it indicates that the output voltage is being controlled by a voltagessense. If no value is assigned to this parameter, it indicates that the controller is a currentssense.

**XCONTROLLER**

Specifies the name of the first controlling source.

**YCONTROLLER**

Specifies the name of the second controlling source.
CCS23

This controlled current source can be controlled either by two voltage senses or by two current senses.

Note: For explanation of the parameters, see CVS23 on page 507.
Currentsense

This is a behavioral representation of an ammeter, which is an electrical device for measuring circuit current.

Ammeters are modeled using zero value voltage sources that can be inserted into the circuit for the purpose of measuring current.

Adding Currentsense has no effect on circuit operation because they represent short-circuit. These voltage sources need not be grounded.
Voltagessense

This is a behavioral representation of a voltmeter, which is an electrical device for measuring voltage drop across two points in a circuit.

![Diagram of Voltagessense]

Adding voltagessense has no effect on circuit operation because they represent an open circuit.
This is a Current Controlled Current Source.

**Arguments**

- **GAIN**

  Used to calculate the output current. The output current is calculated using the equation listed below.

  \[ I_{out} = I_{in} \times GAIN \]

  The input current \( I_{in} \) is the current flowing through the input terminal of the device.
VCCS

This is the model for a voltage controlled current source.

\[
GAIN = 10
\]

\[ I_{out} = V_{in} \times GAIN \]

The input voltage $V_{in}$ is the voltage across the input terminal of the device.

**Arguments**

- **GAIN**
  
  Used to calculate the output current. The output current is calculated using the equation listed below.

  \[ I_{out} = V_{in} \times GAIN \]
CCVS

This is the behavioral model for a Current Controlled Voltage Source.

\[ V_{out} = I_{in} \times GAIN \]

The input current \( I_{in} \) is the current flowing through the input terminal of the device.
**VCVS**

This is the behavioral model for a Voltage Controlled Voltage Source.

\[
GAIN = 10 \\
U1 \\
V_{\text{in}} \\
V_{\text{out}} \\
\text{VCVS}
\]

**Arguments**

**GAIN**

Used to calculate the output voltage. The output voltage is calculated using the equation listed below.

\[ V_{out} = V_{in} \times GAIN \]

The input voltage \( V_{in} \) is the voltage across the input terminal of the device.
Delay lines

PSpice supports two types of delay lines. These are `DELAY_2_TERM`, which is a two terminal device, and `DELAY_3_TERM`, which is a 3 terminal device.

```
DELAY_TIME = 0.7u
IMPEDANCE = 1K
INSERT_LOSS = 100
Q = 1000
```

![DELAY_2TERM_diagram]

**Table 5-16 Delay Line properties**

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DELAY_TIME</td>
<td>introduces the specified delay in the output</td>
</tr>
<tr>
<td>IMPEDANCE</td>
<td>$Z_0$</td>
</tr>
<tr>
<td>INSERT_LOSS</td>
<td>$Q = \frac{\omega L}{R}$</td>
</tr>
</tbody>
</table>

For a three terminal delay line, the third terminal is to be grounded.
DC transformer

This is the PSpice model used for state average analysis.

```
ROUT = 10m
PRI_TURNS = 1
SEC_TURNS = 1
U1
```

**Arguments**

- `PRI_TURNS` - Number of turns in the primary windings
- `SEC_TURNS` - Number of turns in the secondary windings
- `ROUT` - Output resistance

**Comments**

The output voltage is calculated as:

\[
V_{out} = V_{in} \times \left( \frac{SEC\_TURNS}{PRI\_TURNS} \right)
\]
VC_CAP

This is a voltage controller capacitor. The capacitance is a function of the input voltage.

\[
C = V_{in} \times \text{CAP}
\]

**Parameters**

- **CAP**
  - Capacitance factor

**Comments**

The output capacitance is the product of input voltage and the value of the CAP property.

\[
C = V_{in} \times \text{CAP}
\]

The output current is given by the equation:

\[
I_{out} = V_{out} \times (2\pi f C)
\]

where

- \( f \) = operating frequency

Vout is the voltage drop across VC_CAP
VC_CON

This is a voltage controller conductance. Electrical conductance is defined as the reciprocal of resistance.

\[ CON = 2 \]
\[ U1 \]

\[ G = V \times CON \]

\[ + \]
\[ V \]
\[ - \]
\[ G \]

\[ VC\_CON \]

**Parameters**  
**CON**  
Conductance factor

**Comments**  
The output conductance, \( G \), is the product of input voltage and the value of the CON property.

\[ G = V\text{in} \times CON \]

The output current is given by the equation:

\[ I_{out} = V_{out} \times G \]
VC_IND

This is a voltage controlled inductor.

\[ \text{IND} = 1M \]

\[ U1 \]

\[ L = V \cdot \text{IND} \]

\[ \text{VC_IND} \]

**Parameter**  \( \text{IND} \)

Inductance factor

**Comments**

The output inductance, \( L \), is the product of input voltage and the value of the IND property.

\[ L = V_{in} \cdot \text{IND} \]

The output current is given by the equation:

\[ I_{out} = \frac{V_{out}}{2\pi fL} \]
VC_RES

This is a voltage controlled resistor.

\[
RES = 1 \\
U1 \\
R=V\cdot RES \\
VC\_RES
\]

**Parameter**  
\(RES\)  
Resistance factor

**Comments**  
The output resistance, \(R\), is the product of input voltage and the value of the \(RES\) property.

\[R = Vin\cdot RS\]

The output current is given by the equation:

\[I_{out} = Vout/R\]
## Glossary

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABM</td>
<td>analog behavioral modeling</td>
</tr>
<tr>
<td>AKO</td>
<td>“A Kind Of” symbol. Symbols must either contain graphics or refer to an AKO symbol. The AKO defines the symbol in terms of the graphics and pins of another part. Both must exist in the same Symbol Library file.</td>
</tr>
<tr>
<td>alias</td>
<td>An alias relates local schematic names for parts and signals to netlist names (simulation devices and nodes). An alias is an exact electrical equivalent that can be used to reference a symbol. A command that sets up equivalences between pin names or net names and node names. As a command, it is the setup equivalences between node names and pin names or net names.</td>
</tr>
<tr>
<td>annotation</td>
<td>Annotation is a means by which parts are labeled when they are placed, either automatically or manually.</td>
</tr>
<tr>
<td>annotation symbol</td>
<td>An annotation symbol has no electrical significance, and is used to clarify, point out, or define items on the schematic.</td>
</tr>
<tr>
<td>argument</td>
<td>A value or an expression used with an operator or passed to a subprogram (subroutine, procedure, or function).</td>
</tr>
<tr>
<td>attributes</td>
<td>Attributes are special characteristics (a name and an associated value) contained in a part instance or definition. For example, a MOSFET may contain specific length and width parameters which are represented as attributes on the symbol or part. Attributes may be changed through the Property Editor and/or the Parts Editor.</td>
</tr>
<tr>
<td>bus</td>
<td>A bus is a collection of homogeneously named signals.</td>
</tr>
<tr>
<td>call</td>
<td>To transfer a program execution to some section of code (usually a subroutine of some sort), while saving the necessary information to allow execution to resume at the calling point when the call section has completed execution.</td>
</tr>
</tbody>
</table>
circuit  A circuit is a configuration of electrically connected components or devices.

comment  A statement written into a program for documentation purposes only and not for any functionality purposes.

compiler  Translates between high-level computer language understood by humans and machine language that is understood by computers.

component  A device or part employed in a circuit to obtain some desired action. See package.

connector  A connector is a physical device that is used for external connections to a circuit board.

construct  A computer program statement that produces a predetermined effect.

current source  A current source can be an ideal current source (no limit on the supply voltage) or a voltage source with a series resistor.

defined function  A computer instruction specifying the operation to be done with predetermined limits.

declarative statement  A computer source program instruction specifying the size, format, and kind of data elements and variables in a program for a compiler.

device  A simple or complex discrete electronic component. Sometimes, a subsystem employed as a unit and, therefore, thought of as a single component. See package.

DIBL  drain-induced barrier lowering (MOSFET device)

dot command  A type of formatting command typed into a document that is preceded by a period (dot) to distinguish from other syntax text.

doping tail  A changing amount of impurity in a semiconductor device. It is observed as a change in the bulk resistance of the semiconductor material.

ELSE  An operation used in BASIC computer programing. It specifies the operation to be performed if the conditions given in the same program line didn’t occur.

flicker noise  A repeating low-frequency noise.
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fourier analysis</td>
<td>A mathematical method of transforming a function in such a way that the data of the function is retained but the representation of that data is changed. It is used to simplify the reduction of the data.</td>
</tr>
<tr>
<td>FSTIM</td>
<td>Digital file stimulus device</td>
</tr>
<tr>
<td>gate</td>
<td>A gate is a subset of a package, and corresponds to a part instance. An electronic switch that follows a rule of Boolean logic.</td>
</tr>
<tr>
<td>glitch</td>
<td>An unwanted transient that recurs irregularly in the system.</td>
</tr>
<tr>
<td>global temperature</td>
<td>Universally applied temperature (to all elements of a circuit)</td>
</tr>
<tr>
<td>global parameter</td>
<td>Universally applied parameter (to all elements of a circuit)</td>
</tr>
<tr>
<td>hierarchical block</td>
<td>A user defined rectangle placed on a schematic. It is used to represent or hold the place for a collection of circuitry. A block is treated as a black box by the schematic editor. The schematic editor is aware of the connections going into and out of the block, but ignores the contents of the block until the netlist is generated.</td>
</tr>
<tr>
<td>icon</td>
<td>A small graphics image displayed on the screen to represent an object that can be manipulated by the user.</td>
</tr>
<tr>
<td>IF</td>
<td>An operation used in BASIC computer programing. It specifies an IF-THEN operation to be performed when a condition has changed from what was expected in a program line.</td>
</tr>
<tr>
<td>included file</td>
<td>A smaller file that is read into a larger source-code file at a specific spot and becomes part of a statement within the larger source-code file.</td>
</tr>
<tr>
<td>instance</td>
<td>The placement of a component one or more times on a schematic.</td>
</tr>
<tr>
<td>invoke</td>
<td>To call or activate; used in reference to commands and subroutines.</td>
</tr>
<tr>
<td>ionization knee</td>
<td>A bend in the response curve where ionization starts.</td>
</tr>
<tr>
<td>IS temperature</td>
<td>The temperature of the JFET and other transistor types junction saturation current or the input leakage current</td>
</tr>
<tr>
<td>iteration</td>
<td>A repeating series of arithmetic operations to arrive at a solution.</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Jiles-Atherton model</td>
<td>A state equation model rather than an explicit function for an inductor</td>
</tr>
<tr>
<td>junction</td>
<td>A junction graphically indicates that wires, buses, and/or pins are electrically connected.</td>
</tr>
<tr>
<td>keyword</td>
<td>The significant word in a syntax statement that directs the process of the operation.</td>
</tr>
<tr>
<td>labels</td>
<td>Is a word or symbol used to identify a file or other element defined in a computer program.</td>
</tr>
<tr>
<td>LIBPATH</td>
<td>A variable that specifies the directory that the model library is in, and is first set in the PSpice.ini file.</td>
</tr>
<tr>
<td>link</td>
<td>A branch instruction, or an address in such an instruction, used to leave a subroutine to return to some point in the main program.</td>
</tr>
<tr>
<td>lot tolerance</td>
<td>The tolerance of a group of items taken as one unit.</td>
</tr>
<tr>
<td>lsb</td>
<td>least significant bit</td>
</tr>
<tr>
<td>metafile</td>
<td>A file that contains or defines other files.</td>
</tr>
<tr>
<td>mobility</td>
<td>movement of electrons in semiconductor devices such as MOSFETs</td>
</tr>
<tr>
<td>model library</td>
<td>Consists of electrical model definitions for the parts used in PSpice. Usually, the term model library refers to a simulation model library that has .lib extension.</td>
</tr>
<tr>
<td>mouse</td>
<td>A common pointing device used in a windows environment. The physical movement of the mouse will move the pointer (cursor) on the screen.</td>
</tr>
<tr>
<td>msb</td>
<td>most significant bit</td>
</tr>
<tr>
<td>msim.ini</td>
<td>Legacy MicroSim configuration file that has the default elements that are used to complete a simulation.</td>
</tr>
<tr>
<td>nesting</td>
<td>The embedding of one construct (such as a table in a database; a data structure, a control structure) inside another—for example, a nested procedure is a procedure declared within a procedure.</td>
</tr>
<tr>
<td>NETLIST</td>
<td>The netlist provides the circuit definition and connectivity information in simulation netlist format.</td>
</tr>
<tr>
<td>NODESET</td>
<td>A noderset symbol contains one or two pins, permitting you to initialize a node voltage for simulation.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>---------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>NOREUSE flag</td>
<td>A piece of information that tells the simulator that the automatic saving and restoring of bias point information between different temperatures, Monte Carlo runs, worst-case runs, or parametric analyses is suppressed. It is one of the options in the <code>.OPTIONS (analysis options)</code> on page 71 command.</td>
</tr>
<tr>
<td>NOSUBCKT</td>
<td>A variable that tells the simulator not to save the node voltages and inductor currents for subcircuits.</td>
</tr>
<tr>
<td>NUMDGT</td>
<td>An option that tells the simulator the number of digits that will be printed for the analog values. It is one of the options in the <code>.OPTIONS (analysis options)</code> on page 71 command.</td>
</tr>
<tr>
<td>object</td>
<td>A variable comprising both routines and data that is treated as a discrete entity, in object-oriented programming.</td>
</tr>
<tr>
<td>operator</td>
<td>A symbol (mathematical, as an example) or other character indicating an operation that acts on one or more elements.</td>
</tr>
<tr>
<td>OUTPUT ALL</td>
<td>An option that asks for an output from the sensitivity runs, after the nominal (first) run. The output from any run is governed by the <code>.PRINT (print)</code> on page 84, <code>.PLOT (plot)</code> on page 82, and <code>.PROBE (Probe)</code> on page 86 command in the file. If OUTPUT ALL is omitted, then only the nominal and worst-case runs produce output. OUTPUT ALL ensures that all sensitivity information is saved for Probe.</td>
</tr>
<tr>
<td>package</td>
<td>A package is an enclosure for an electronic device or subsystem. A physical device consisting of one or more gates.</td>
</tr>
<tr>
<td>page</td>
<td>A page may contain both parts (represented by symbols), port instances, connectors, and annotation symbols. A page may or may not have a title. Each schematic page represents a single page of a circuit design.</td>
</tr>
<tr>
<td>parameter</td>
<td>A property used in model definition to define the electrical behavior of the model.</td>
</tr>
<tr>
<td>part</td>
<td>A part is an electrical component that is represented by a schematic symbol. The term refers to the logical, rather than the physical, component.</td>
</tr>
<tr>
<td>part definition</td>
<td>See symbol.</td>
</tr>
<tr>
<td>part instance</td>
<td>A part instance refers to an occurrence of a symbol in a schematic.</td>
</tr>
<tr>
<td>Term</td>
<td>Definition</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>pin</td>
<td>Pins are contained in parts, ports, and offpage connectors. Parts can contain multiple pins. Each part contains specific pin names associated with the part. Pins may connect to a wire, a bus, or another pin.</td>
</tr>
<tr>
<td>pin current</td>
<td>The current that flows into or out-of a defined pin.</td>
</tr>
<tr>
<td>POLY</td>
<td>Specifies the number of dimensions of the polynomial.</td>
</tr>
<tr>
<td>port</td>
<td>A port provides connectivity across schematic pages. A port provides the anchor for a single pin. Ports are chosen from library files, placed, moved, and deleted in the same way as are parts. Ports may have multiple connections. Ports consist of three types: global, interface, and offpage.</td>
</tr>
<tr>
<td>run</td>
<td>The execution of a computer routine or operation.</td>
</tr>
<tr>
<td>SCBE</td>
<td>substrate current induced body effect (MOSFET device)</td>
</tr>
<tr>
<td>schematic</td>
<td>A schematic consists of the following components: one or more pages, a set of symbols representing local part definitions or parts in a library file, and/or text.</td>
</tr>
<tr>
<td>setpoint</td>
<td>A setpoint provides a graphical way of introducing .IC (initial bias point condition) on page 49 or .NODESET (set approximate node voltage for bias point) on page 66 commands for each instance of a symbol. These commands set one or more node voltages for the bias point calculation.</td>
</tr>
<tr>
<td>SIMLIBPATH</td>
<td>A variable that defines the environment that the simulator is working in (path to the directory that the library is in).</td>
</tr>
<tr>
<td>simulation</td>
<td>The use of a mathematical model to represent a physical device or process.</td>
</tr>
<tr>
<td>skipbp</td>
<td>(skip bias point)</td>
</tr>
<tr>
<td>statement</td>
<td>The smallest executable entity within a programming language. In general, each line of a program is an individual statement and is considered an individual instruction. (Examples: command statements, option statements, control statements, assignment statements, comment statements.)</td>
</tr>
<tr>
<td>Statz model</td>
<td>A GaAsFET model</td>
</tr>
</tbody>
</table>
**subcircuit model**
A model that is not built in PSpice. These models use the .SUBCKT and .ENDS statements along with variable input parameters to define the structure and function of a part.

**symbol**
A symbol consists of the graphical representation of a logical or physical electronic part on the schematic page, and its definition. Symbols can be created either for a specific schematic or extracted from a library file, and may contain schematic pages nested within them.

**symbol library**
Contains symbols or graphical representation of off-the-shelf PSpice parts that can directly be used in circuits being developed. In Capture, a symbol library has .OLB extension.

**syntax**
The grammar of a particular computer language, with rules that govern the structure and content of the statement.

**TEXTINT**
A function which returns a text string which is the integer value closest to the value of the `<value or expression>`; `<value or expression>` is a floating-point value

**tick number**
The number generated from a regular recurring signal emitted by a clocking circuit, or from the interrupt generated by this signal.

**TOM model**
a GaAsFET device

**VARY BOTH**
The default option is VARY BOTH. When VARY BOTH is used, sensitivity to parameters using both DEV and LOT specifications is checked only with respect to LOT variations. The parameter is then maximized or minimized using both DEV and LOT tolerances for the worst-case. All devices referencing the model have the same parameter values for the worst-case simulation.

**VARY DEV**
See VARY BOTH.

**VARY LOT**
See VARY BOTH.

**VTO temperature**
The temperature of the JFET or MOSFET device when there is zero-bias threshold (pinchoff) voltage.

**window**
An area on the screen in a graphical computer interface that contains instructional documentation or a message.
Appendix A: Battery Models
Battery Model

General form

```
.X awbflooded_cell PARAMS VOC=<value> AH=<value> SOC=<value>
```

```
.X awbvalve_regulated_cell PARAMS VOC=<value> AH=<value> SOC=<value>
```

Examples

```
.X awbflooded_cell PARAMS VOC=5 AH=15 SOC=0.8
```

Arguments and options

**awbflooded_cell**

PSpice model for modelling the flooded cell batteries. In the flooded cells batteries since the gases created during charging are vented to the atmosphere, distilled water must be added occasionally to bring the electrolyte back to its required level.

Example: 12-V automobile battery.

**awbvalve_regulated_cell**

PSpice model for modelling the valve regulated batteries.

**VOC**

Indicates the open circuit voltage. This is the voltage across the two terminals of the battery when the battery is not connected to a circuit.

**AH**

It is the ampere hour of the battery. This is the amount of time for which a battery operates without having to recharge it. For example, if a battery is marked 300Ah then it is assumed that the battery can supply 20A current for 15 hours or 10A current for 30 hrs.

**Note:** An ampere hour (Ah) indicates the amount of energy charge in a battery that will allow one ampere of current to flow for one hour.

**SOC**

Indicates the state of charge in a battery. For a completely charged battery, SOC is 100% and for a fully discharged battery, SOC is 0%.
Capture Parts

FLOODED CELL

VOC = 5
AH = 300

VALVE REGULATED CELL

VOC = 5
AH = 300
Index

Symbols

; (in-line comment) 126
* (comment) 125

Numerics

74181 model 399
74393 subcircuit example 347

A

A/D and D/A converters 393
ABM 168
defined 521
polynomial transfer function 171
ABS 460
absolute function 460
absolute value (ABS) 14
ABSTOL 453
ABSTOL (.OPTIONS) 74
.AC 32
AC analysis 32
ACCT (.OPTIONS) 71
ACOS(x) 14
ADC 393
ADVCONV (.OPTIONS) 71
AFFECTS 408
air-gap 209, 210
AKO 140, 196, 222, 240, 282, 305
defined 521
alias
defined 521
.ALTER 129
ammeter 509
amplitude, peak 190, 191
analog devices 63
passive
semiconductor
analog parts
breakout parts 158, 207, 221, 303
capacitors 303
ideal switches 308, 324
inductor coupling (KBREAK) 208
inductors 303
resistors 303
semiconductor parts 228
analog switch function
asw1 468
analog-to-digital converter 60
analyses
AC 32
bias point 70
DC 37
Fourier 47
Monte Carlo 54
noise 67
parametric sensitivity 98
sensitivity/worst-case 121
temperature 109
transient 113
analysis options
flag options 71
AND 358
AND3 360
anhysteric 209, 210
annotation
defined 521
annotation symbol
defined 521
AO 358
arbitrary current source 461
arbitrary voltage source
single pin 470
two pin 466
arc tangent (ATAN and ARCTAN) 14
arccosine function 14
arccosine hyperbolic function 14
ARCOS(x) 14
arcsine 14
arcsine hyperbolic function 14
arctan hyperbolic function 14
arctangent 14
argument 106, 107, 329, 521
arithmetic expressions 17
arithmetic functions
ABS 460
DIFFERENCE2 461
DIVIDER2 463
EXP 463
MAX 464
MIN 464
MULTIPLIER2 465
SQRT 465
SUM2 465
ASIN(x) 14
ASINH(x) 14
asw1 468
ATAN2 14
ATANH(x) 14
attributes
definition 63, 73, 521

B

Battery Model 530
BEHAV_FREQ 469
BEHAV_GEN 470
behavior
static 213
behavioral primitives 347, 397
constraint check 409
logic expression 397
pin-to-pin delay 400
bias point .NODESET 66
.SKIPBP 113
small-signal 49
biaspoint
transient 49
bidirectional delay line 312
bidirectional transfer gates 346, 362
binary notation format 421
bipolar transistor 272
summary 131, 134
bipolar transistor model
quasi-saturation effect 290
bipolar transistor, PNP 60
block 329, 523
BOOLEAN 401, 402, 410
Boolean expression IF 14
break() 452
breakout parts 158, 221, 303
inductor coupling 207
BSIM3 model
advanced parameters 232
BUF 358
BUF3 361
buffer 465
buffer, output 485
bulk, MOSFET terminal (substrate) 90
bus 79
defined 521

C

call 86, 106, 133, 329, 529
CAP device model 60
capacitor 60, 156
summary 131, 134
capacitors 303
voltage coefficients 128
CASE 405
ccs10 505
ccs23 508
CHANGED
reference function 403
CHANGED_HL
reference function 403, 406
CHANGED_LH
reference function 403
CHEBYSHEV 165
CHGTOL 454
CHGTOL (.OPTIONS) 74
circuit 133, 156, 329, 332, 522
circuit topology 97
CLEAR 411
CLOCK 410
CLRBAR 406
coil spring function 471
collector 90
command
reference 29
syntax format 12
command files
Probe 19
command line options
PSpice 25
PSpice A/D 25
comment 125, 522
bias point files 53
included files 50
in-line 126
comment line 125
common simulation data file (CSDF) 86
compiler 522
complex digital devices 397
complex pole, frequency and damping, complex pole
frequency and damping 477
complex pole, real and imaginary, complex pole
real and imaginary 480
component 133, 219, 223, 302, 332, 522
conductance 75
connectors 522
CONSTRAINT 399, 407, 409, 414
PSpice messages 78
constraint check 409
FREQ 413
GENERAL 414
SETUP_HOLD 410
WIDTH 412
continuation line 382, 386, 389
controlled current source
cs10 505
cs23 508
controlled voltage source
cvs10 502
cvs23 507
conventions 29
expression 29
numeric value 29
convergence
real pole and zero function blocks 496
convergence hazard 78
convergence problems 97
corrections 129
converters 393, 395
convolution 315
CORE device model 60
core model 211, 219
cores
defining static behavior 213
cosine (COS) 14
counter function
divide by 2 462
divide by 3 462
divide by 4 462
divide by 5 462
CPTIME (.OPTIONS) 74
CPU time 74
current limiter 489
current source 33, 522
EXP parameters 182
PULSE parameters 184
SIN parameters 191
SSFM parameters 190
CURRENT_GEN 461
current-controlled
current source 131, 134, 174
switch 60, 132, 135, 322
voltage source 131, 135, 174
currentsense 509
cvs10 502
cvs23 507
D
D device model 60
DAC 395
damping factor
current source 191
data
data file 24
.DC 37
DC analysis 37
DC gain 112
DC sweep 37
Ddt(x) 14
declarative statement 522
DEFAD (.OPTIONS) 74
DEFAS 74
default temperature (TNOM) 76
defined function 41, 43, 48, 522
defining
static behavior 213
DEFL (.OPTIONS) 74
DEFW (.OPTIONS) 74
DEG 167
DELAY 165, 461
delay 405
delay line 346, 378
delay values 352
DELAY1 461
delta() 450
DEV tolerance 61, 124
device 229, 230, 232, 522
device temperatures, customized 63
device terminal abbreviation 90
devices 131, 135
A/D converter 393
bipolar transistor 272
capacitor 156
complex digital 397
current-controlled current source 174
current-controlled switch 322
current-controlled voltage source 174
D/A converter 393
digital input 434
October 2012 535 Product Version 16.6
digital output 439
digital primitive 131
digital stimulus 132
digital-to-analog interface 434
diode 160
GaAsFET 137
independent current source & stimulus 180
independent current source & stimulus (sinusoidal waveform) 191
independent voltage source & stimulus 180
inductor 219
inductor coupling
inductor coupling (transformer core) 202
input/output model 431
insulated gate bipolar transistor 332
interface 344
JFET 194
MOSFET 132, 136, 224
passive 63
primitives 345
programmable array logic 446
resistor 302
semiconductor 63
stimulus 344
subcircuit instantiation 329
subcircuits 132
transmission line 312, 322
transmission line coupling 132, 136, 216
voltage-controlled current source 165
voltage-controlled switch
voltage-controlled voltage source 165
DFF 368
DIBL 232, 247, 248, 522
differential function 14
DIGDRV (OPTIONS) 74
DIGDRVF (OPTIONS) 74
DIGDRVZ (OPTIONS) 74
DIGERRDEFAULT (OPTIONS) 74, 415
DIGERRLIMIT (OPTIONS) 74, 415
DIGFREQ (OPTIONS) 74
DIGINITSTATE (OPTIONS) 74
DIGIOLVL 352
DIGIOLVL (OPTIONS) 74
digital delay line 60
digital input 60, 434
digital input model parameters 435
  C (capacitance) 435
  FILE 435
  FORMAT 435
  Sn (state "n") 435
  TIMESCALE 436
digital libraries 444
digital output 60, 439
  .VECTOR 116
digital output model parameters 439
  CHGONLY 439
  CLOAD 439
  FILE 439
  FORMAT 439
  RLOAD 439
  Sn (state "n") 439
  SXNAME 440
  TIMESCALE 440
  TIMESCALE 440
digital power supplies 431, 433, 445
digital primitives 345
  format 349
digital simulation
  results in files 116
  worst-case timing 79
digital time step 74
digital worst-case timing 78
  convergence hazard 78
  cumulative ambiguity hazard 79
  digital input voltage 79
  glitch suppression 79
  net state conflict 79
  persistent hazard 79
  zero-delay-oscillation 79
digital-to-analog
  converter 60
  interface devices 434
  DIGMNTX (OPTIONS) 74, 352
  DIGMNTYSCALE (OPTIONS) 75, 353
  DIGOVDR (OPTIONS) 75
  DIGTYMXSCALE (OPTIONS) 75
  DINPUT device model 60, 435
diode model 60, 131, 135, 160
display file 25
  .DISTO (small-signal distortion) 128
  .DISTRIBUTION 41, 73
distribution
  user-defined 41
  using an include file 41
distribution name
  GAUSS 62
  user-defined 62
distributions
  UNIFORM tolerances name 61
DIV_BY_2 462
DIV_BY_3 462
DIV_BY_4 462
DIV_BY_5 462
divide function 463
DLTCH 374
DLYLINE 378
DMFACTOR (.OPTIONS) 72
doping tail 248, 280, 522
dot command 156, 181, 522
DOUTPUT device model 60
drain 89
area 74
bulk 226
induced 241
drive resistance 74

E

edge-triggered flip-flops 60, 367
truth tables 371, 372, 373
ELSE 14, 522
emitter 90
ENABLE 406
.END 44
end of circuit 44
end subcircuit definition 105
ENDREPEAT 420
. ENDS 105
environment variables
LIBPATH 51
ERRORLIMIT 415
examples
CONSTRANT primitive 415
EXPAND (.OPTIONS) 53, 72
exponential (EXP) 14
exponential function 463
exponential temperature coefficient (TCE) 305
expressions 17, 48
conventions 13
numeric conventions 13
text 111
.EXTERNAL 46
external port specifications 46

F

FALSE 412

ferromagnetic 218
Ferroxcube 205
FET 60
file 24, 425
data 24
header 425
input 23
output 24
stimulus 425
transitions 425
files
log 20, 24
stimulus 118
filter shift 101
final time value 113, 181
first order transfer function 493
flicker noise 154, 164, 201, 268, 292, 522
flip-flops and latches 346, 366
initialize 366
timing constraints 367
X-level handling 366
flush interval 23
FORMAT 437
format array 424
.FOUR 47
Fourier analysis 29, 47, 113, 523
fourth order transfer function 493
FREQ (constraint check) 413
frequency
expression 165
modulation 181
response, AC analysis 33
frequency domain
current source 460
expression block 469
voltage source 466
FSTIM 107, 111, 425, 427, 523
.FUNC 48
function definition 48
functions
absolute value (ABS) 14
arctangent (ATAN and ARCTAN) 14
arccosine (ACOS) 14
arccosine hyperbolic (ACOSH) 14
arc sine hyperbolic (ASINH) 14
arctan hyperbolic (ATANH) 14
arctangent (ARCTAN) 14
arsine (ASIN) 14
ATAN2 14
break() 452
cosine (COS) 14
cosine hyperbolic 14
delta 450
differential (Ddt) 14
exponential (EXP) 14
hyperbolic tangent (TANH) 16
IF 14
imaginary (IMG) 15
integral (Sdt) 15
inverse cosine hyperbolic (ACOSH) 14
inverse sine hyperbolic (ASONH) 14
inverse tan hyperbolic (ATANH) 14
limit (LIMIT) 15
log base 10 (LOG10) 15
log base E (LOG) 15
MAX 15
MIN 15
phase (P) 15
power (PWR) 15, 17
real (R) 15
signed power (PWRS) 15
signum (SGN) 15
sine (SIN) 15
square root (SQRT) 15
state() 451
step (STP) 15
table (TABLE) 16
tangent (TAN) 16
time() 450
FY1 493
FY2 493
FY3 493
FY4 493

G
GaAs MESFET 60
GaAsFET 131, 135, 137, 141
device model 60
Level 1 parameters 141
Level 2 parameters 141
Level 3 parameters 143
Level 4 parameters 140
parameters for all levels
number of inputs 356
standard 356
tri-state 360
gate-source 88
Gaussian 62
GENERAL (constraint check) 414
Glitch 523
suppression 79
global
node names 107
parameters 523
ports 523
GMIN 454
GMIN (.OPTIONS) 75
GMIN_FROM_SOURCESTEP (.OPTIONS) 72
GMINSTEMPS (.OPTIONS) 75
goal functions
command line 25
AC analysis 32
Gummel-Poon transistor model
quasi-saturation effect 290

H
H(s) source
current source 460
expression block 469
voltage source 466
harmonics 47
header
file 425
HEX radix functions 425
hexadecimal notation 422
HEXFET 270
HOLDTIME 411
hyperbolic tangent (TANH) 16
hysteresis 209

I
I/O model 418
.IC 49, 66
bias point
.IC setting 49
IC= 114, 156, 219
icon 11, 523
ideal transmission line 313

IGBT
  extracting model parameters from data sheets 334

imaginary part 92

IMG(x) 15

.INC 48, 50, 58

included file 30, 106, 123, 523

.DISTRIBUTION command 74

including files 48, 50

INCR BY 423

IND device model 60

independent current source & stimulus 132, 135, 180

sinusoidal waveform 191

independent sources
  AC analysis 33

independent voltage source & stimulus 132, 136, 180

inductor 60, 132, 136, 219

  coupling 136, 203

  coupling (transformer core) 202

inductors 303

  current coefficients 128

initial bias point condition 49

in-line comment 126

input file 23

input/output model parameters 351, 431

  AtoD 431
  DIGPOWER 431
  DRV 431
 DtoA 431
  OUT 431
  TPWRT 432

instance name 134, 523

instantiate 105

insulated gate bipolar transistor
  (IGBT) 132, 136, 332

integral (Sdt) 15

integrator function 491

Intel hex format 384, 388

interface 344

internal time step 114

INV 358

INV3 361

invoke 523

IO_LEVEL 352, 419

IO_STM 418, 421, 428

ionization knee 199, 523

IS = 437

IS temperature 141, 162, 197, 282, 523

ISWITCH device model 60

iteration 189, 523

ITL3 128

ITL6 (.OPTIONS) 75

ITLn (.OPTIONS) 75

J

JEDEC file 107, 111, 379, 382, 383, 446

JFET 132, 136, 194

Jiles-Atherton model 206, 209, 524

JKFF 368

job statistics (ACCT) 71

junction definition 226, 229, 524

K

KBREAK (inductor coupling) 208

KEYWORD 165, 188, 524

  DEG 167
  MAG 167
  PARAMS 329

  R_I 167

  RAD 167

L

labels 524

Laplace domain function
  GAIN2 463

Laplace domain functions
  complex pole, frequency and damping 477
  complex pole, real and imaginary 480

laplace domain functions
  real pole 495

  realzero 496

Laplace Domain Sources 469

laplace domain functions
  FY1 493
  FY2 493
  FY3 493
  FY4 493

Laplace variable 48, 165

latch 366

  gated 374
lateral PNP 60
.LIB 51
LIBPATH 51, 524
LIBRARY (.OPTIONS) 72
library file 51
LIMIT (.OPTIONS) 75
limit (LIMIT) 15
limiter
  current, ILIM 489
  voltage, VLIM 490
LIMPTS (.OPTIONS) 75
LIMTIM 128
linear
  temperature coefficient (TC1) 305
link 524
LIST (.OPTIONS) 72
.LOADBIAS 96
  load bias point file 53
log (logarithmic)
  base 10 (LOG10) 15
  base E (LOG) 15
log files 20
PSpice 24
log Probe commands 20
logic expression 397
LOGICEXP 397, 399, 400
lossy transmission line 60, 314, 315
lot tolerance 61, 123, 124, 524
LPNP device model 60
lsb 118, 524
LVLCOD 128
LVLTIM 128

M

macro file 25
MAG 167
magnetic core 204, 206
magnitude 76, 92
  M(x), functions 15
master library file 51
MAXFREQ 413
maximum
  (MAX) 15
maximum value function 464
MAXORD 128
.MC 54
mechanical elements
  viscous friction 466
memory primitives 347
RAM 388
ROM 384
messages 78, 414
  DIGITAL INPUT VOLTAGE 79
  FREQUENCY 78
  GENERAL 78
  hazard and timing violation 78
  HOLD 78
  NET-STATE CONFLICT 79
  PERSISTENT HAZARD 79
  persistent hazards 46
  RELEASE 78
  SETUP 78
  Timing Violations 78
  WIDTH 78
  ZERO–DELAY– OSCILLATION 79
metafile 524
METHOD 128
method (.OPTIONS) 75
MIN_LO 413
MINFREQ 413
minimum (MIN) 15
minimum value function 464
MNTYMXDLY 352
mobility 232, 241, 242, 249, 336, 524
.MODEL 58
model
  library 51, 524
  temperature customization 63
model breakout parts
resistors 303
D (diode) 161
GASFET (GaAsFET) 138
IGBT 333
ISWITCH (current-controlled switch) 324
LPNP (bipolar transistor) 279
NJF (JFET) 195
NMOS (MOSFET) 228
NPN (bipolar transistor) 279
PJF (JFET) 195
PMOS (MOSFET) 228
PNP (bipolar transistor) 279
VSWITCH (voltage-controlled switch) 308
X (diode) 161
models, using in circuit designs 133
monostable multivibrator 486
Monte Carlo analysis 41, 54, 95
default distribution values 74
read-in error 101

October 2012 540 Product Version 16.6
MOSFET 132, 224
  BSIM3 version 3.1 model
description 233
  BSIM3 version 3.1 model
parameters 252
device declaration 136
  EKV version 2.6 model description 230
  EKV version 2.6 model parameters 243
Level 4 description 229
Level 6 advanced parameters 232
Levels 1, 2, and 3 descriptions 229
model declaration 60
model parameters 229

noise
analysis 67
  bipolar transistor 292
diode 164
  flicker 154, 164, 201, 268, 292
  JFET 201
MOSFET 268
  shot 154, 201, 269, 292
thermal 154, 164, 201, 269, 292, 328
NOM.LIB 51
nominal temperature 109, 146, 159, 163,
  198, 222, 264, 288, 306
NOMOD (.OPTIONS) 73
nonlinear controlled sources 70
NOOPITER (.OPTIONS) 73
NOOUTMSG (.OPTIONS) 73
NOPAGE (.OPTIONS) 73
NOPRBMSG (.OPTIONS) 73
no-print value 113
NOR 358
NOR3 361
NOREUSE (.OPTIONS) 73
NOREUSE flag 73, 97, 525
NOSTEPDEP (.OPTIONS) 71
NOSUBCKT 94, 525
NPN bipolar transistor 60
number of times to repeat (n) 420
NUMDGT (.OPTIONS) 76, 84, 525
numeric expression convention 13
NXOR 359
NXOR3 361

N

N device 434
NAND 358
NAND3 361
NBTG 363
N-channel 60
  GaAsMESFET 60
  IGBT 60
  JFET 60
  NMOS 60
nested subcircuits 128
nesting 329, 524
netlist
definition 524
device declarations 134
intrinsic device types 134
proper syntax 133
NIGBT device model 60
NJF device model 60
NMOS device model 60
NOBIAS (.OPTIONS) 72
NODE (.OPTIONS) 72, 413
.NODESET 53, 66, 96
ndeset 29, 53, 66, 97, 524
NOECHO (.OPTIONS) 72
NOGMIN (.OPTIONS) 72
NOICTRANSLATE (.OPTIONS) 73
.NOISE 67

O

OA 359
object 525
OCT radix functions 425
ONE_SHOT 486
.OP 70
operator 110, 525
.OPTIONS 71, 74
  ABSTOL 74
  ADVCONV 71
  CHGTOL 74
  CPTIME 74
  DEFAD 74
  DEFFAS 74
  DEFL 74
  DEFW 74
  DIGDRVF 74
DIGDRVZ  74
DIGERRDEFAULT  415
DIGERRLIMIT  415
DIGFREQ  74
DIGINITSTATE  74
DIGIOLVL  74
DIGMNTYMNX  74
DIGMNTYSCL  353
DIGOVRDVR  75
DIGTYMXSCL  75
DISTRIBUTION  73
DMFACTOR  72
EXPAND  53
GMIN  75
GMIN_FROM_SOURCESTEP  72
GMINSTEPS  75
ITL1  75
ITL2  75
ITL4  75
ITL5  75
ITL6  75
LIBRARY  72
LIMIT  75
LIMPTS  75
LIST  72
method  75
NOBIAS  72
NODE  72
NOECHO  72
NOGMIN  72
NOICTRANS  73
NOMOD  73
NOOPITER  73
NOOUTMSG  73
NOPAGE  73
NOPRBMSG  73
NOREUSE  73
NOSTEPDEP  71
NUMDG  84
OPTS  73
PIVREL  76
PIVTOL  76
PSEUDOTRAN  73
PTRANSTEP  76
RELTOL  76
SOLVER  76
STEPGMIN  73
THREAD  76
TNOM  76
TRTOL  76
VNTOL  76
WIDTH  84
ACCT  71
options  23, 71
options not available in PSpice  128
MAXORD  128
OPTS (.OPTIONS)  73
OR  359
OR3  361
OUT1  465
OUTPUT ALL  54, 121, 123, 525
output buffer  485
inverted output  465
output file  24
and flag options  71
resistances  112
output files
digital simulation results  116
output variables
DC sweep  87
transient analysis  87

P

package  525
page options for simulation output  73
page, definition  525
.PARAM  80
parameter  525
definition  80
global  140, 159, 162, 523
parametric analysis  99
PARAMS  105
subcircuits  329
part  525
definition  525
instance  525
path definition  402
PBTG  363
P-channel  60
JFET  60
MOSFET  60
peak amplitude  190, 191
performance package (SOLVER)  76
phase  92
phase (P)  15
piecewise linear  41
pin  302, 397, 526
pin current  526
pin names equivalent to node names  34
pins
and capacitor node polarity 156
and inductor node polarity 219, 223
as external interface points of
subcircuit nodes 105
voltage out of range 79
pin-to-pin delay (PINDLY) 399, 400, 401, 402, 408
PIVREL (.OPTIONS) 76
PIVTOL (.OPTIONS) 76
PJF device model 60
PLAND 380
PLD 107, 111, 379
overview 379
PLNAND 380
PLNOR 381
PLNXOR 381
PLOR 381
.PLOT 82
plot 82
PLXOR 381
PMOS device model 60
PNP device model 60
POLY 526
polynomial transfer function (for ABM
controlled sources) 171
ports 526
global 523
power (PWR) 15, 17
.prb file 25
primitives 344, 345, 400
.PRINT 84
print 84
step value 113, 181
tables 84
Probe 86
data files 70
output files 44
.PROBE 86
Probing command line options
-c 24
-p 25
programmable logic array PLD 446
data values 382
overview 379
PLAND 380
PLANDC 380
PLNAND 380
PLNANDC 381
PLNOR 381
PLNORC 381
PLNXOR 381
PLNXORC 381
PLOR 381
PLORC 381
PLXOR 381
PLXORC 381
syntax 380
timing model 383
types 347, 380
propagation delay 353
PSEUDOTRAN (.OPTIONS) 73
PSpice
log file 24
PSpice command line options
-d 24
-I 24
-o 24
PSpice messages
hazard and timing violations 78
Persistent Hazards 46
PTRANSTEP (.OPTIONS) 76
PULLDN 377
PULLUP 377
pullup and pulldown resistors 346, 377
Q
QBAR 86
quadratic temperature coefficient
(TC2) 305
quasi-saturation effect (BJT) 290
R
R device 302
R_I 167
RAD 167
RAM 388
random access memory
timing model 390
read only memory 384
timing model 386
real function (R) 15
real part 92
real pole function 495
real zero function 496
reference functions 403
CHANGED 403
CHANGED_HI 403
CHANGED_LH 403
relative accuracy 76
RELEASETIME 411
RELTOL 453
RELTOL (.OPTIONS) 76
REPEAT 189, 420
ENDREPEAT 189
FOREVER 189
RES device model 60
resistance multiplier 305
resistor 60, 131, 136, 302
model definition 302
pullup and pulldown 377
RMS 68
roll-off
current 280
ROM 384
run 167, 307, 526
Probes commands 20

S

S device 307
save
bias point to file 94
.SAVEBIAS 53, 94
scale factor 75
SCBE 232, 249, 526
schematic 526
Schmitt trigger 431
Sdt(x) integral function 15
second order transfer function 493
semiconductor models 129
semiconductor parts 228
.SENS 98
sensitivity
analysis 98
worst-case analysis 121
setpoint 526
setting initial conditions 49
SETUP_HOLD
constraint check 410
SETUPTIME 411
SGN(X) signum function 15
shot noise 154, 201, 269, 292
shunt conductance 80, 104
SIGNAME 429, 438
signed power (PWRS) 15
SIMLIBPATH 526
simulation 167, 314, 526
algorithm choices (SOLVER) 76
lossy transmission lines 314
transmission line 312
simulation command line options
-bf 23
-bn 24
-bs 24
-e 24
-i 24
-wONLY 24
SIN(x) function 15
sine (SIN) 15
skip bias point, SKIPBP 113
skipbp 73, 113, 114, 526
small-signal 70
bias point 49
DC gain 112
small-signal distortion analysis 128
solution algorithm (SOLVER) 76
SOLVER (.OPTIONS) 76
source 89
SPICE2 options 128
SPICE2G
PSpice differences 128
spring function 471
square root (SQRT) 15
square root function 465
square wave voltage controlled
oscillator 487
SRFF 374
standard gates 60, 345, 356, 358
AND 358
ANDA 358
AO 358
AOI 359
BUF
BUFA 358
INV 358
INVA 358
NAND 358
NANDA 358
NANDA 358
NOR 358
NORA 359
NXOR 359
NXORA 359
OA 359
OAI 359
OR 359
ORA 359
XOR 359
XORA 359
<table>
<thead>
<tr>
<th>Keyword</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>state()</td>
<td>451</td>
</tr>
<tr>
<td>statement</td>
<td>80, 97, 169, 203, 204, 225, 526</td>
</tr>
<tr>
<td>static behavior</td>
<td>213</td>
</tr>
<tr>
<td>Statz model</td>
<td>138, 152, 526</td>
</tr>
<tr>
<td>.STEP</td>
<td>99</td>
</tr>
<tr>
<td>usage examples</td>
<td>101</td>
</tr>
<tr>
<td>step ceiling value</td>
<td>113</td>
</tr>
<tr>
<td>step function (STP)</td>
<td>15</td>
</tr>
<tr>
<td>STEPMIN (.OPTIONS)</td>
<td>73</td>
</tr>
<tr>
<td>stepping a resistor</td>
<td>101</td>
</tr>
<tr>
<td>STIM</td>
<td>418</td>
</tr>
<tr>
<td>.STIMLIB</td>
<td>103</td>
</tr>
<tr>
<td>.STIMULUS</td>
<td>104</td>
</tr>
<tr>
<td>stimulus definition</td>
<td>104</td>
</tr>
<tr>
<td>stimulus devices</td>
<td>344, 417</td>
</tr>
<tr>
<td>examples</td>
<td>421</td>
</tr>
<tr>
<td>file stimulus</td>
<td>425</td>
</tr>
<tr>
<td>stimulus generator</td>
<td>418</td>
</tr>
<tr>
<td>stimulus library files</td>
<td>103</td>
</tr>
<tr>
<td>STP(x)</td>
<td>15</td>
</tr>
<tr>
<td>subcircuit</td>
<td>132, 134, 136, 314</td>
</tr>
<tr>
<td>definition</td>
<td>105</td>
</tr>
<tr>
<td>device declarations</td>
<td>105, 329</td>
</tr>
<tr>
<td>instantiation</td>
<td>105, 329</td>
</tr>
<tr>
<td>intrinsic device types</td>
<td>51</td>
</tr>
<tr>
<td>library</td>
<td>51</td>
</tr>
<tr>
<td>usage examples</td>
<td>107</td>
</tr>
<tr>
<td>subcircuit model</td>
<td>527</td>
</tr>
<tr>
<td>.SUBCKT</td>
<td>105</td>
</tr>
<tr>
<td>usage examples</td>
<td>107</td>
</tr>
<tr>
<td>substrate</td>
<td>90</td>
</tr>
<tr>
<td>sum function</td>
<td>465</td>
</tr>
<tr>
<td>sweep variable</td>
<td>56, 95</td>
</tr>
<tr>
<td>DC analysis</td>
<td>40</td>
</tr>
<tr>
<td>switch</td>
<td>132, 135</td>
</tr>
<tr>
<td>current-controlled</td>
<td>132, 135</td>
</tr>
<tr>
<td>voltage-controlled</td>
<td>131, 136</td>
</tr>
<tr>
<td>switches, ideal</td>
<td>308, 324</td>
</tr>
<tr>
<td>SXNAME</td>
<td>441</td>
</tr>
<tr>
<td>symbol</td>
<td>527</td>
</tr>
<tr>
<td>symbols</td>
<td></td>
</tr>
<tr>
<td>ideal switches</td>
<td>308, 324</td>
</tr>
<tr>
<td>BBREAK (GaAsFET)</td>
<td>138</td>
</tr>
<tr>
<td>C (capacitor)</td>
<td>157</td>
</tr>
<tr>
<td>C_VAR (capacitor)</td>
<td>158</td>
</tr>
<tr>
<td>CBREAK (capacitor)</td>
<td>158</td>
</tr>
<tr>
<td>DBREAKx (diodes)</td>
<td>161</td>
</tr>
<tr>
<td>E (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>EPOLY (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>F (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>FPOLY (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>G (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>GPOLY (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>H (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>HPOLY (ABM controlled analog source)</td>
<td>171</td>
</tr>
<tr>
<td>JBREAKx (JFET)</td>
<td>195</td>
</tr>
<tr>
<td>K_LINEAR (transformer)</td>
<td>207, 220</td>
</tr>
<tr>
<td>KBREAK (inductor coupling)</td>
<td>208</td>
</tr>
<tr>
<td>KCOUPLEn (transmission line coupling matrix)</td>
<td>318</td>
</tr>
<tr>
<td>L (inductor)</td>
<td>220</td>
</tr>
<tr>
<td>LBREAK (inductor)</td>
<td>221</td>
</tr>
<tr>
<td>MBREAKx (MOSFET)</td>
<td>228</td>
</tr>
<tr>
<td>QBREAKx (bipolar transistor)</td>
<td>279</td>
</tr>
<tr>
<td>R (resistor)</td>
<td>303</td>
</tr>
<tr>
<td>R_VAR (resistor)</td>
<td>303</td>
</tr>
<tr>
<td>RBREAK (resistor)</td>
<td>304</td>
</tr>
<tr>
<td>SBREAK (voltage-controlled switch)</td>
<td>308</td>
</tr>
<tr>
<td>T (transmission line)</td>
<td>316</td>
</tr>
<tr>
<td>TLOSSY (transmission line)</td>
<td>316</td>
</tr>
<tr>
<td>TnCOUPLED (transmission line)</td>
<td>316</td>
</tr>
<tr>
<td>TnCOUPLEDX (transmission line)</td>
<td>317</td>
</tr>
<tr>
<td>XFRM_LINEAR (transformer)</td>
<td>207, 220</td>
</tr>
<tr>
<td>XFRM_NONLINEAR (nonlinear transformer)</td>
<td>207</td>
</tr>
<tr>
<td>ZBREAKN (IGBT)</td>
<td>333</td>
</tr>
<tr>
<td>syntax</td>
<td>12, 62, 104, 527</td>
</tr>
</tbody>
</table>

**T**

<table>
<thead>
<tr>
<th>Keyword</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>T device</td>
<td>312</td>
</tr>
<tr>
<td>T_ABS (.MODEL)</td>
<td>63</td>
</tr>
<tr>
<td>T_MEASURED (.MODEL)</td>
<td>63</td>
</tr>
<tr>
<td>T_REL_GLOBAL (.MODEL)</td>
<td>63</td>
</tr>
<tr>
<td>T_REL_LOCAL (.MODEL)</td>
<td>63</td>
</tr>
<tr>
<td>table (TABLE)</td>
<td>16</td>
</tr>
<tr>
<td>tangent (TAN)</td>
<td>16</td>
</tr>
<tr>
<td>tangent hyperbolic (TANH)</td>
<td>16</td>
</tr>
<tr>
<td>TC</td>
<td>305</td>
</tr>
<tr>
<td>1 (linear temperature coefficient)</td>
<td>305</td>
</tr>
<tr>
<td>2 (quadratic temperature coefficient)</td>
<td>305</td>
</tr>
<tr>
<td>E (exponential temperature coefficient)</td>
<td>305</td>
</tr>
<tr>
<td>TEMP</td>
<td>453</td>
</tr>
</tbody>
</table>
.TEMP 109
temperature 80, 109
customization 63
temperature customizing 63
terminal abbreviation 90
TEXT
   subcircuit 329
.TEXT 110
text 110
text parameter definition 110
TEXTINT 110, 527
.TF 112
thermal noise 154, 164, 201, 269, 292, 328
thermal voltage 80
third order transfer function 493
THREAD (.OPTIONS) 76
tick number 437, 527
TIME 453
time domain function
   TIME 465
time function 465
      DELAY 461
      DELAY1 461
      ONE_SHOT 486
      VCO 487
time() 450
TIMESTEP 425, 442
TIMESCALE 425, 442
timing constraint 354
timing hazards
   convergence 78
cumulative ambiguity 79
timing model 351
delay line 378
gated latch 374
general discussion 353
multi-bit A/D converter 393
programmable logic array 383
random access memory 390
read only memory 386
TNOM (.OPTIONS) 76
tolerances 61, 124
   DEV 61
distribution name 61, 62
   GAUSS 62
   LOT 61
   specification 61
   UNIFORM 61
   user-defined 62
TOM model 138, 527
topology 97
.TRAN 113
   transfer function 112
   transfer functions 491, 493
   restrictions 494
   transformer 60
   transient analysis 47, 113
      final time value
      internal time step 114
      no-print value 113
      print step value 113
   SKIPBP 113
   step ceiling value 113
transient bias point 49
   transistor, bipolar 272
   transition functions 404
      TRN_$H 404
      TRN_$L 404
      TRN_H$ 404
      TRN_HL 404
      TRN_HZ 404
      TRN_L$ 404
      TRN_LH 404
      TRN_LZ 404
      TRN_Z$ 404
      TRN_ZH 404
      TRN_ZL 404
transitions
   file 425
   transmission line
      ideal line 313
      illustration 313, 314
      lossy line 314
   transmission line coupling 131, 132, 136, 216, 312, 322
   transmission line device 312
   transmission lines
      coupled 318
      ideal 316
      lossy 316
      simulating 318
TRISTATE 401, 406
tri-state gates 60, 346, 360
   AND3 360
   AND3A 361
   BUF3 361
   BUF3A 361
   INV3 361
   INV3A 361
   NAND3 361
   NAND3A 361
   NOR3 361
NOR3A  361
NXOR3  361
NXOR3A  361
OR3  361
OR3A  361
XOR3  361
XOR3A  361
TRN device model  60
TRTOL (.OPTIONS)  76
TSTEP  181
TSTOP  181
typographical conventions  12

U

U.C. Berkeley
address  271
UADC device model  60, 393
UBTG  363
UDAC device model  60, 395
UDLY device model  60, 378
UEFF device model  60, 368
UGATE  399
UGATE device model  60, 356, 397
UGFF device model  60, 374
UIO device model  60, 431
UPLD  380
URAM  388
UROM  384
user-defined distribution  41
UTGATE device model  60, 360
current source  131, 134, 165
flux source  165
switch  60, 131, 136
voltage source  131, 134, 165
VSWITCH device model  60
VTO temperature  527

W

.WATCH  119
watch analysis results  119
.WCASE  121
WHEN  414
.WIDTH  128
.IN= option  128
WIDTH (.OPTIONS)  76, 84, 125
WIDTH (constraint check)  412
wildcard characters  23
window menu  527
Windows  11, 78
worst-case analysis  41, 95, 121

X

X devices  105
XOR  359
XOR3  361

Z

zero impedance voltage source, DAC  395