# Virtuoso® RelXpert Reliability Simulator User Guide

Product Version 13.1.1
April 2014

© 2003–2014 Cadence Design Systems, Inc. All rights reserved.
Printed in the United States of America.

Cadence Design Systems, Inc. (Cadence), 2655 Seely Ave., San Jose, CA 95134, USA.

Product MMSIM contains technology licensed from, and copyrighted by: C. L. Lawson, R. J. Hanson, D. Kincaid, and F. T. Krogh © 1979, J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson © 1988, J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling © 1990, University of Tennessee, Knoxville, TN and Oak Ridge National Laboratory, Oak Ridge, TN © 1992-1996; Brian Paul © 1999-2003; M. G. Johnson, Brisbane, Queensland, Australia © 1994; Kenneth S. Kundert and the University of California, 1111 Franklin St., Oakland, CA 94607-5200 © 1985-1988; Hewlett-Packard Company, 3000 Hanover Street, Palo Alto, CA 94304-1185 USA © 1994; Silicon Graphics Computer Systems, Inc., 1140 E. Arques Ave., Sunnyvale, CA 94085 © 1996-1997; Moscow Center for SPARC Technology, Moscow, Russia © 1997; Regents of the University of California, 1111 Franklin St., Oakland, CA 94607-5200 © 1990-1994; Sun Microsystems, Inc., 4150 Network Circle Santa Clara, CA 95054 USA © 1994-2000; Scriptics Corporation and other parties © 1998-1999; Aladdin Enterprises, 35 Efal St., Kiryat Arye, Petach Tikva, Israel 49511 © 1999; and Jean-loup Gailly and Mark Adler © 1995-2005, RSA Security, Inc., 174 Middlesex Turnpike Bedford, MA 01730 © 2005.

**Trademarks**: Trademarks and service marks of Cadence Design Systems, Inc. contained in this document are attributed to Cadence with the appropriate symbol. For queries regarding Cadence's trademarks, contact the corporate legal department at the address shown above or call 800.862.4522. All other trademarks are the property of their respective holders.

**Restricted Permission:** This publication is protected by copyright law and international treaties and contains trade secrets and proprietary information owned by Cadence. Unauthorized reproduction or distribution of this publication, or any portion of it, may result in civil and criminal penalties. Except as specified in this permission statement, this publication may not be copied, reproduced, modified, published, uploaded, posted, transmitted, or distributed in any way, without prior written permission from Cadence. Unless otherwise agreed to by Cadence in writing, this statement grants Cadence customers permission to print one (1) hard copy of this publication subject to the following conditions:

1. The publication may be used only in accordance with a written agreement between Cadence and its customer.
2. The publication may not be modified in any way.
3. Any authorized copy of the publication or portion thereof must include all original copyright, trademark, and other proprietary notices and this permission statement.
4. The information contained in this document cannot be used in the development of like products or software, whether for internal or external use, and shall not be used for the benefit of any other party, whether or not for consideration.

**Disclaimer:** Information in this publication is subject to change without notice and does not represent a commitment on the part of Cadence. Except as may be explicitly set forth in such agreement, Cadence does not make, and expressly disclaims, any representations or warranties as to the completeness, accuracy or usefulness of the information contained in this document. Cadence does not warrant that use of such information will not infringe any third party rights, nor does Cadence assume any liability for damages or costs of any kind that may result from use of such information.

**Restricted Rights:** Use, duplication, or disclosure by the Government is subject to restrictions as set forth in FAR52.227-14 and DFAR252.227-7013 et seq. or its successor

# Contents




© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

© 2003-2014
All Rights Reserved.

© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

© 2003-2014
All Rights Reserved.

# Preface

The *Virtuoso® RelXpert Reliability Simulator User Guide* is intended for integrated circuit designers who want to perform transistor-level reliability simulation and analysis.

## Licensing for Reliability Analysis

RelXpert is a standalone product that is used for performing reliability analysis. In addition, the reliability analysis functionality is natively integrated into the MMSIM portfolio of simulators, which include UltraSim, Spectre, and Accelerated Parallel Simulator (APS). Table -1 lists the base products required to run reliability analysis and enable use of the proprietary AgeMos model. Similar to other analyses in MMSIM, native reliability analysis is also enabled with MMSIM tokens.

**Note:** The proprietary AgeMos models are also made available in a standalone configuration based on the simulators used for the reliability analysis task.

**Table -1**

| Function | RelXpert Standalone | Integrated in Spectre | Integrated in APS | Integrated in UltraSim |
|---|---|---|---|---|
| Reliability | RELXPRT | Virtuoso Spectre Simulator | Virtuoso Accelerated Parallel Simulator | Virtuoso UltraSim Simulator |
| Proprietary AgeMos Model | RELXPRT | Virtuoso Spectre Simulator | Virtuoso Accelerated Parallel Simulator | Virtuoso UltraSim Simulator |

## Related Documents

For additional information about the Virtuoso RelXpert Reliability simulator and related products, refer to the following manuals:

- *Virtuoso RelXpert Reliability Simulator What’s New*

  Contains information about new, changed, and deleted features, compatibility issues, and documentation changes.

- *Virtuoso UltraSim Simulator User Guide*


© 2003-2014 All Rights Reserved.

Contains information about the Virtuoso UltraSim™ simulator.

# Typographic and Syntax Conventions

The following typographic and syntax conventions are used in this manual.

| | |
|---|---|
| `text` | Indicates text you must type exactly as it is presented. |
| *`argument`* | Indicates text that you must replace with an appropriate argument. |
| `[ ]` | Denotes optional arguments. When used with vertical bars, they enclose a list of choices from which you can choose one. |
| `{ }` | Used with vertical bars and encloses a list of choices from which you must choose one. |
| \| | Separates a choice of options. |
| ... | Indicates that you can repeat the previous argument. |

The language requires many characters not included in the preceding list. You must type these characters exactly as they are shown in the syntax.


© 2003-2014
All Rights Reserved.

# 1

# Introducing the RelXpert Reliability Simulator




© 2003-2014 All Rights Reserved.

# Overview

The Virtuoso® RelXpert Reliability Simulator provides transistor-level reliability simulation and analysis functionality, which enable you to consider reliability effects in the early stages of design and ensure that circuits have sufficient margins to function correctly over their entire lifetimes.

The following reliability simulation features are supported:

- Hot carrier injection (HCI)
- Negative bias temperature instability (NBTI)
- Positive bias temperature instability (PBTI)
- Cadence ageMOS models
- User-defined degradation models through the Virtuoso Unified Reliability Interface (URI)

This chapter provides an overview of the Virtuoso® RelXpert reliability simulator and describes Virtuoso RelXpert simulator features and capabilities. A flow chart on how to use the Virtuoso RelXpert simulator is also included.

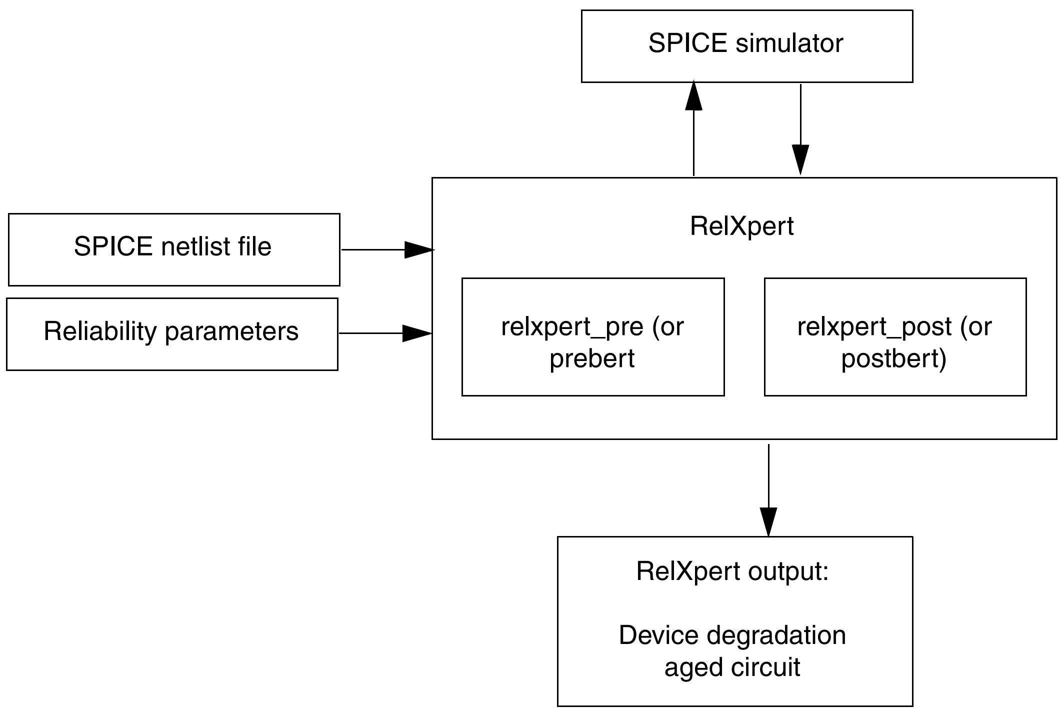
> **Note:** Starting in the 7.1.1 release, the prebert, postbert, and prebert2 binary names were replaced as follows:
>
> - prebert was replaced with relxpert_pre
> - postbert was replaced with relxpert_post
> - prebert2 was replaced with relxpert_pre -age
>
> However, to maintain backward compatibility, the RelXpert software will continue to support the prebert, postbert, and prebert2 binary names.


© 2003-2014 All Rights Reserved.

# Virtuoso RelXpert Reliability Simulator Flow

The Virtuoso RelXpert simulator consists of pre- and post-processors (relxpert_pre and relxpert_post, respectively) linked by a SPICE simulator, as illustrated in Figure 1-1.

**Figure 1-1 Overview of the Virtuoso RelXpert Simulator**

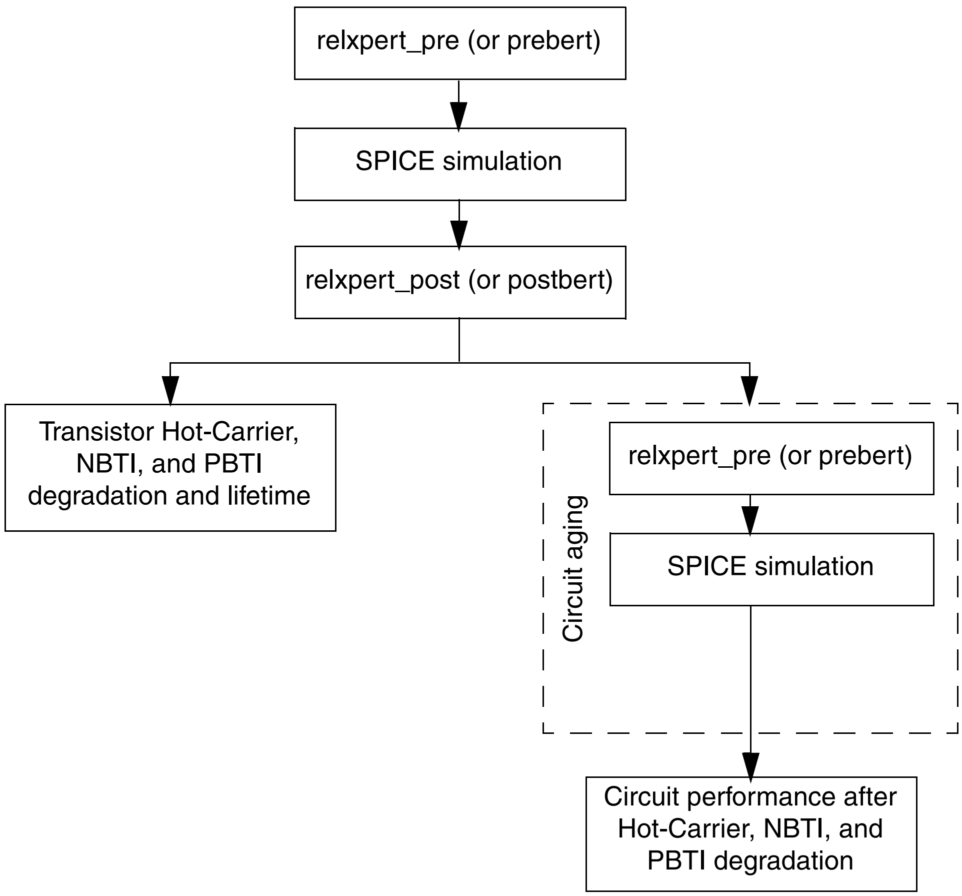



The Virtuoso RelXpert simulator reads the SPICE input netlist file. The reliability parameters are included in the netlist file. The SPICE simulator generates the voltage and current waveforms used in the reliability calculation.

The sequence of a Virtuoso RelXpert simulation is illustrated in Figure 1-2 on page 16. A single-pass execution of relxpert_pre (or prebert), SPICE, and relxpert_post (or postbert) yields the results for transistor Hot-Carrier/NBTI degradation and lifetime. Second-pass simulations simulate circuit performance due to degraded transistors.


© 2003-2014 All Rights Reserved.

### Figure 1-2 Sequence of Virtuoso RelXpert Simulations





© 2003-2014 All Rights Reserved.

# Virtuoso RelXpert Simulator Features

The Virtuoso RelXpert simulator features include:

- Three reliability models (see Figure 1-3):
  - MOSFET Hot-Carrier Injection (HCI) module

    Predicts transistor and circuit performance degradation due to HCI effects
  - PMOSFET Negative Bias Temperature Instability (NBTI) module

    Predicts PMOS transistor and circuit performance degradation due to NBTI effects
  - NMOSFET Positive Bias Temperature Instability (PBTI) module

    Predicts NMOS transistor and circuit performance degradation due to PBTI effects
- SPICE netlist file input

  No modifications of the netlist file are required.
- Reliability results from postprocessing of available SPICE output, which minimizes the added overhead in the design cycle
- Support for UC Berkeley series models (BSIM3, BSIM3V3, BSIM4, BSIMSOI, BSIMCMG), Philips MOSFET model (MOS9, MOS11), TI Spice3 MOSFET models (BSIM1, BSIM2, BSIM3, BSIM3v3, BSIM4, BSIMSOI), PSP102, PSP103, HVMOS, HISIM_HV, and HISIM2 models.
- Support for equation and table-lookup methods for Hot-Carrier lifetime calculation
- Support for binned models
- Support for the Virtuoso RelXpert simulator in the ADE interface
- Accepts parameters extracted by the BSIMProPlus® model extractor (from ProPlus Design Solutions Inc) or BSIMProPlus RelET, the automatic Hot-Carrier stressing, and the characterization system (BSIMProPlus® and BSIMProPlus RelET are from ProPlus Design Solutions Inc)
- Support for HCI, NBTI, and PBTI Reliability simulation
- Support for the agemos model
- Support for Unified Reliability Interface (URI) (Refer to the *Virtuoso Unified Reliability Interface* manual)
- Support for TMI Aging


© 2003-2014 All Rights Reserved.

# Using the Virtuoso RelXpert Simulator

Figure 1-3 on page 19 illustrates how to use Virtuoso RelXpert simulator and the Virtuoso RelXpert/ADE interface. The tasks include:

1. Characterizing and extracting reliability parameters

   Detailed procedures are in Chapter 3, "Extracting Hot-Carrier Parameters for MOS Transistors." Cadence also provides the BSIMProPlus® model extractor (from ProPlus Design Solutions Inc) and BSIMProPlus RelET for characterization and extraction of reliability parameters.

2. Generating model files for the Virtuoso RelXpert simulator

   Detailed procedures are in Chapter 6, "Setting Up Models for Simulation."

3. Preparing the SPICE input netlist file for the Virtuoso RelXpert simulator

   The Virtuoso RelXpert simulator commands are entered into the SPICE input netlist file in batch mode. The Virtuoso RelXpert simulator control statements are documented in Chapter 7, "Simulator Control Statements." The Virtuoso Spectre, TI Spice3, and TITAN interfaces are documented in Chapter 8, "Virtuoso Spectre Interface," Chapter 9, "TI Spice3 Interface," and Chapter 10, "TITAN Interface."

4. Running the Virtuoso RelXpert simulation

   Operating instructions are in Chapter 11, "Running the Simulator."

5. Interpreting the Virtuoso RelXpert simulation results

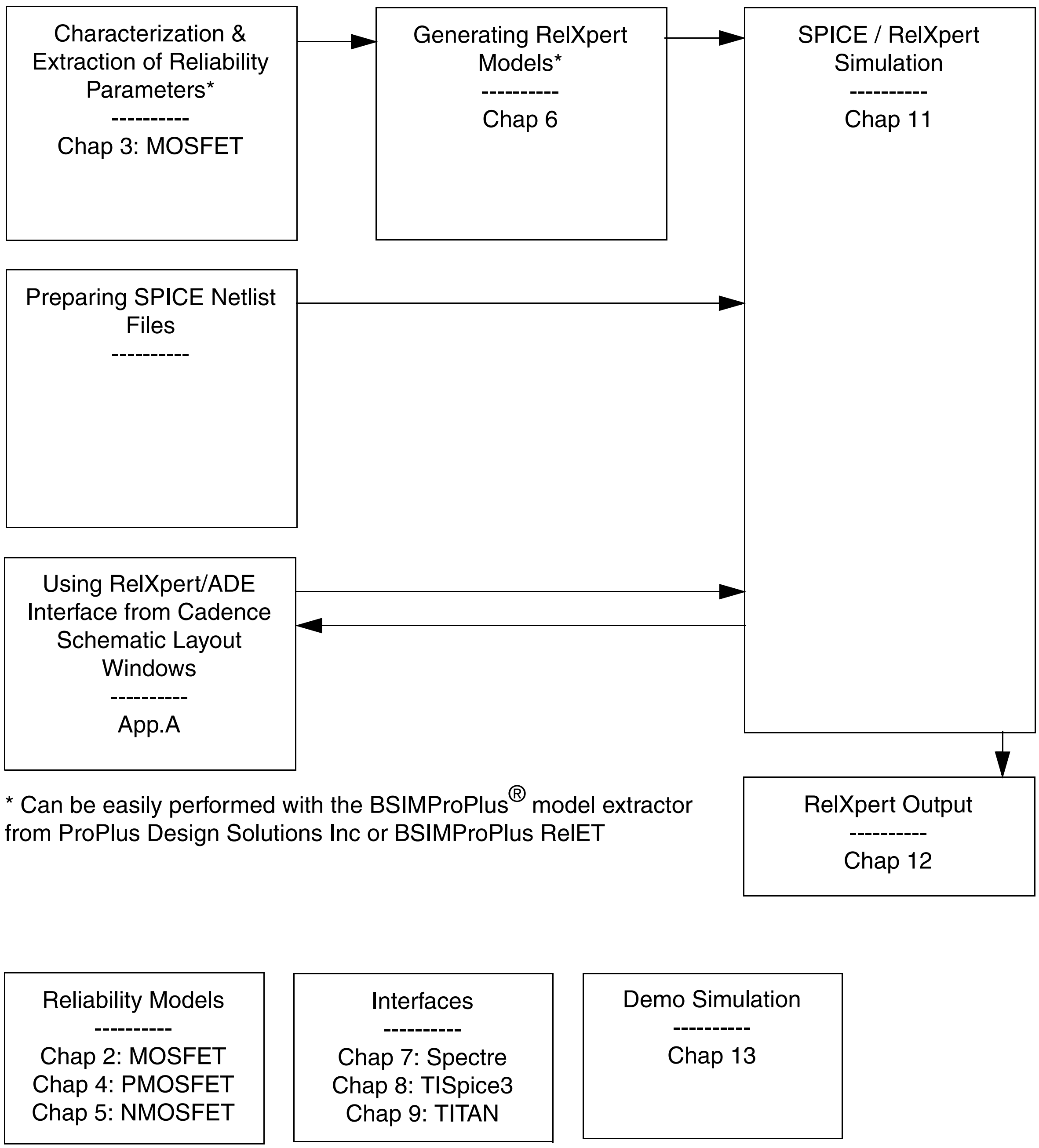
   The simulation results are explained in Chapter 12, "Simulator Output."

You will find it useful to go through the demo simulation described in Chapter 13, "Examples of Simulations."

The physics and derivation of reliability models are discussed in Chapter 2, "Hot-Carrier Degradation Model for MOS Transistors," Chapter 4, "Negative Bias Temperature Instability Degradation Model for PMOS Transistors.", and Chapter 5, "Positive Bias Temperature Instability Degradation Model for NMOS Transistors."


© 2003-2014 All Rights Reserved.

**Figure 1-3  Flowchart for Using the Virtuoso RelXpert Simulator**




# Installing Virtuoso RelXpert Simulator

To install the Virtuoso RelXpert simulator software:


© 2003-2014 All Rights Reserved.

1. Do one of the following:

   a. If you have downloaded the software from downloads.cadence.com, follow the instructions in the *Cadence Installation Guide* to extract and install the software.

   b. If you receive the installation as a tar.gz file, extract the program by typing

   ```
   gunzip -c relxpert_v10.1.102._lnx86.tar.gz | tar xvf -
   ```

2. In the `.cshrc` file, include the executable path in your search path:

   ```
   set path = ($path /relxpert_installation/tools.sun4v/bin)
   ```

   To use the Virtuoso RelXpert reliability simulator 64-bit software, set the `CDS_AUTO_64BIT` environment variable to `relxpert` or `ALL` in your `.cshrc` file. For example:

   `setenv CDS_AUTO_64BIT ALL` or `setenv CDS_AUTO_64BIT relxpert`

**Note:** Do not launch the executables directly from the *`your_install_dir`*`/bin/64bit` or *`your_install_dir`*`/bin/32bit` directory.

The Virtuoso RelXpert Simulator has the following file structure:

| | |
|---|---|
| `bin/` | executables |
| `relxpert/bin/` | RelXpert software |
| `relxpert/lib` | libraries |
| `relxpert/example` | Example test cases |
| `relxpert/uri` | URI templates and example |

# Common Problems in Running the Virtuoso RelXpert Simulator

- Cannot find the Virtuoso RelXpert simulator

  Make sure `RelXpert` is in the search path.

- Cannot start the Virtuoso RelXpert simulator because of a license problem

  - License unavailable because someone else is using the Virtuoso RelXpert simulator

    Type `lmstat -a` to check license usage.


© 2003-2014 All Rights Reserved.

- License unavailable because the license server is not up and running

  Check the license server status, or start the license server.
- License has expired

- The Virtuoso RelXpert simulator cannot find the SPICE output file
  - Make sure the output file from the SPICE simulator is available for postprocessing.
  - Virtuoso Spectre or TI Spice3 is not in the search path.
  - The `.rhosts` file, or `/etc/hosts.equiv` in the remote host, is not set up correctly to allow silent logins.
  - The raw data file from SPICE simulation (PSF file in Virtuoso Spectre and `.pun` file in Spice3) is not available because the rawfile output option in the `.option` statement is not set up properly.

For more information about the `.options` statement, refer to Chapter 8, “Virtuoso Spectre Interface,” Chapter 9, “TI Spice3 Interface,” or Chapter 10, “TITAN Interface.”


© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 2

# Hot-Carrier Degradation Model for MOS Transistors

The models for Hot-Carrier degradation in MOS transistors include:

- A model for calculating substrate current (NMOSFET, PMOSFET) and gate (PMOSFET) current
- A lifetime model, which calculates the Hot-Carrier lifetime under circuit operating conditions from experimental results obtained under accelerated test conditions
- An aging model, which describes the degradation of transistor characteristics as a function of stress: this model generates degraded model parameters for SPICE simulation of degraded circuit performance

## MOSFET Substrate and Gate Current Models

Hot-Carrier induced degradation in n-channel MOSFETs is correlated to substrate current $I_{sub}$. The correlation exists because both Hot-Carriers and substrate current are driven by a common factor—the maximum channel electric field $E_m$, which occurs at the drain end of the channel. In p-channel MOSFETs, where the dominant driving force for degradation is charge trapping in the gate oxide, the degradation is found to be correlated to gate current $I_g$. The Virtuoso® RelXpert reliability simulator calculates these currents using the following models.

### Substrate Current Model

The substrate current is generated from impact ionization and therefore is related to the maximum channel electric field $E_m$[1]

**(2-1)** $$I_{sub} = \frac{A_i}{B_i} E_m l_c I_d \exp\left(-\frac{B_i}{E_m}\right)$$


© 2003-2014
All Rights Reserved.

Both $A_i$ and $B_i$ are physical constants. $l_c$ is the characteristic length of the saturation region. $E_m$ can be solved using a pseudo two-dimensional approach[2].

**(2-2)** $$E_m \approx \frac{(V_{ds} - V_{dsat})}{l_c} \quad \text{and} \quad V_{dsat} = \frac{E_{crit} L_{eff} (V_{gs} - V_t)}{E_{crit} L_{eff} + V_{gs} - V_t}$$

The threshold voltage $V_t$ can be calculated from SPICE model parameters or obtained from a SPICE output rawfile (by setting `.vthmethod spice`). If $V_t$ is calculated by the Virtuoso RelXpert simulator (`.vthmethod calc`, which is also the default), you can select a $V_t$ that is dependent on a $V_{ds}$ voltage (by setting hcilevel=1 which is the default) or a $V_t$ value that is fixed at the value calculated from $V_{ds} = 0$ (by setting hcilevel=2).

Substituting 2-2 in 2-1:

**(2-3)** $$I_{sub} = \frac{A_i}{B_i}(V_{ds} - V_{dsat}) I_d \exp\left(-\frac{B_i l_c}{(V_{ds} - V_{dsat})}\right)$$

$E_{crit}$ and $l_c$ are bias dependent:

**(2-4)** $$E_{crit} = E_{crit0} + E_{critg} V_{gs} + E_{critb} V_{bs} + E_{critd} V_{ds}$$

**(2-5)** $$l_c = (l_{c0} + l_{c1} V_{ds} + l_{c2} V_{ds} V_{gs}) \sqrt{t_{ox}}$$

## Temperature Substrate Current Model

The temperature dependence of MOSFET substrate current is modeled using the activation energy $E_a$:

**(2-6)** $$\frac{I_{sub}(T)}{I_{ds}(T)} = \frac{I_{sub}(Tnom)}{I_{ds}(Tnom)} \times \exp\left(\frac{E_a}{8.62e-5}\left(\frac{1}{T} - \frac{1}{Tnom}\right)\right)$$

$T$ is the circuit operating temperature (K) set in the `.temp` statement. $T_{nom}$ is the nominal temperature (K) of the model (set using `.option tnom` or `.model tref`). $E_a$ (in eV) is the activation energy.

The temperature model compares the ratio of $I_{sub}/I_{ds}$ at two temperatures rather than $I_{sub}$ alone. This is because the drain current itself is changing with temperature.


© 2003-2014 All Rights Reserved.

The parameters for substrate current used by the Virtuoso RelXpert simulator and their default values are shown in Table 2-1.

**Table 2-1 Default Parameters for Substrate Current**

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $A_i$ | Ai | 1/cm | $2.0 \times 10^6$ (NMOSFET) |
| | | | $1.0 \times 10^7$ (PMOSFET) |
| $B_i$ | Bi[a] | V/cm | $1.7 \times 10^6$ (NMOSFET) |
| | | | $3.7 \times 10^6$ (PMOSFET) |
| $E_{crit0}$ | Ecrit0 | V/cm | $1.0 \times 10^4$ |
| $E_{critg}$ | Ecritg | 1/cm | 0.0 |
| $E_{critb}$ | Ecritb | 1/cm | 0.0 |
| $E_{critd}$ | Ecritd | 1/cm | 0.0 |
| $l_{c0}$ | lc0 | $\mu m^{1/2}$ | 1.0 |
| $l_{c1}$ | lc1 | $\mu m^{1/2}/V$ | 0.0 |
| $l_{c2}$ | lc2 | $\mu m^{1/2}/V$ | 0.0 |
| $E_a$ | Eai | eV | 0.0 |
| | hcilevel[b] | | 1 |

a Values for $B_i$ cannot be changed.
b hcilevel 1: $V_t$ is dependent on $V_{ds}$; hcilevel 2: $V_t$ value is fixed at the value calculated from $V_{ds} = 0$.

The parameters can have length- and width-dependent sensitivity parameters. These sensitivity parameter names are the parameter names in the table with prefix `l` or `w`: for example, `lAi`, `lEcrit0`, `lEcritg`, `lEcritb`, `lEcritd`, `llc0`, `llc1`, `llc2`, `wAi`, `wEcrit0`, `wEcritg`, `wEcritb`, `wEcritd`, `wlc0`, `wlc1`, `wlc2`, `pAi`, `pEcrit0`, `pEcritb`, `plc0`, `plc1`, and `plc2`. The final value of any parameter is based on the SPICE model binning equation:

**(2-7)** For bsim3v3,

$$P = P_0 + P_l \cdot \left(\frac{1}{L_{eff}} - \frac{1}{L_{ref}}\right) + P_w \cdot \left(\frac{1}{W_{eff}} - \frac{1}{W_{ref}}\right) + P_p \cdot \left(\frac{1}{L_{eff}} - \frac{1}{L_{ref}}\right)\left(\frac{1}{W_{eff}} - \frac{1}{W_{ref}}\right)$$


© 2003-2014 All Rights Reserved.

**(2-8)** For bsim4

$$P = P_0 + \frac{P_l}{L_{eff}} + \frac{P_w}{W_{eff}} + \frac{P_p}{(L_{eff}W_{eff})}$$

**(2-9)** For BSIMCMG

$$P = P_0 + \frac{1.0e-6}{L_{eff} + DLBIN} \cdot P_l + \frac{1.0}{NFIN} \cdot P_n + \frac{1.0e-6}{NFIN \cdot (L_{eff} + DLBIN)} \cdot P_p$$

where $P_0$ is the length and width independent parameter. $P_l$, $P_w$, and $P_p$ are the length and width dependent parameters respectively. $L_{eff}$ and $W_{eff}$ are the effective channel length and width from SPICE in µm. $L_{ref}$ and $W_{ref}$ are the reference channel length and width in the SPICE parameter.

## PMOSFET Gate Current Model

The gate current is given by[3]:

**(2-10)**

$$I_g = I_{sub} \int_0^{\Delta L} 0.25 \frac{E\lambda}{\phi_b} \exp\left(-\frac{\phi_b}{E\lambda}\right) P(E_{ox}) \frac{dx}{\lambda_r}$$

where $\Delta L$ is the portion of channel length where velocity saturation occurs. Equation 2-10 can be explained as follows:

1. The electrons are generated from impact ionization: that is, proportional to the substrate current $I_{sub}$.
2. The probability that the electrons do not suffer energy loss from scattering during the journey from channel to the *Si*-*SiO*$_2$ barrier is proportional to $P(E_{ox})dx/\lambda_r$, where $E_{ox}$ is the electric field in the oxide and $\lambda_r$ is the redirection mean free path.
3. The probability that these electrons have energy larger than the *Si*-*SiO*$_2$ barrier height $\phi_b$ in a field of $E$ is proportional to $E\lambda/\phi_b \exp(-\phi_b/E\lambda)$ , where $\lambda$ is the scattering mean free path for electrons.

The integral in 2-10 can be evaluated to:


© 2003-2014
All Rights Reserved.

**(2-11)** $$I_g = 0.5 I_{sub} t_{ox} \frac{P(E_{ox})}{\lambda_r} \left(\frac{E_m \lambda}{\phi_b}\right)^2 \exp\left(-\frac{\phi_b}{E_m \lambda}\right)$$

Substituting expressions for $I_{sub}$ and $E_m$ from 2-2 and 2-3:

**(2-12)** $$I_g = 0.5 I_d t_{ox} \frac{A_i}{B_i} \frac{P(E_{ox})}{\lambda_r \phi_b^2} \frac{\lambda^2}{l_c^2} (|V_{ds}| - V_{dsat})^3 \exp\left(-\left(\frac{\phi_b}{\lambda} + B_i\right) \frac{l_c}{(|V_{ds}| - V_{dsat})}\right)$$

$V_{dsat}$ is calculated from 2-2 using the absolute values for $V_{gs}$ and $V_t$.

The electric field in the gate-drain region $E_{ox}$ is given by:

**(2-13)** $$E_{ox} = \frac{(V_{gd} - V_{fb})}{t_{ox}}$$

where $V_{fb}$ is the flat-band voltage from the SPICE model. Usually, $V_{fb} = V_t - \phi_s - K1\sqrt{\phi_s}$ where $V_t$, $\phi_s$, and $K1$ are the threshold voltage, surface potential at inversion, and body effect factor respectively. The calculation of $V_{fb}$ is dependent on the level of MOSFET model used. If you provide $V_{fb}$, the Virtuoso RelXpert simulator uses the provided $V_{fb}$.

The barrier height $\phi_b$ is a function of $E_{ox}$:

**(2-14)**

$$\phi_b = 3.2 - 2.6 \times 10^{-4} \sqrt{E_{ox}} - \nu E_{ox}^{2/3}$$

The expression for $P(E_{ox})$ is complicated and depends on the direction of $E_{ox}$:

- For $E_{ox}$ greater than 0:


© 2003-2014 All Rights Reserved.

**(2-15)**

$$P(E_{ox}) = \left[\frac{5.66 \times 10^{-6} E_{ox}}{1 + 6.90 \times 10^{-6} E_{ox}} \cdot \frac{1}{1 + \frac{2 \times 10^{-3}}{L_{eff}} \exp(-E_{ox} t_{ox} / 1.5)} + 2.5 \times 10^{-2}\right] \times \exp(-300 / \sqrt{E_{ox}})$$

- For $E_{ox}$ less than 0:

**(2-16)** $$P(E_{ox}) = 2.5 \times 10^{-2} \exp\left(\frac{-t_{ox}}{\lambda_{ox}}\right)$$

$E_{ox}$ is in units of V/cm. $L_{eff}$ is in units of cm.

The values for the constants in equations 2-10 through 2-16 are shown in Table 2-2.

**Table 2-2 Constants Used for Gate Current Calculation**

| Parameters | Units | Values |
|---|---|---|
| $\lambda$ | $\mu m$ | 0.0105 |
| $\lambda_r$ | $\mu m$ | 0.0616 |
| $\lambda_{ox}$ | $\mu m$ | 0.0320 |
| $\upsilon$ | $V^{1/3}cm^{2/3}$ | $4 \times 10^{-5}$ |

Equation 2-11 can be simplified to:

**(2-17)** $$I_g = I_d \frac{A_i^g}{B_i} \frac{P(E_{ox})}{\phi_b^2} (|V_{ds}| - V_{dsat})^3 \exp\left(-\left(\frac{\phi_b}{\lambda} + B_i\right) \frac{l_c^g}{(|V_{ds}| - V_{dsat})}\right)$$

$B_i$ is set to 3.7 MV/cm. $A^g{}_i$ and $l^g{}_c$ are extracted from measured data. $l^g{}_c$ is bias dependent:

**(2-18)** $$l_c^g = \left(l_{c0}^g + l_{c1}^g |V_{ds}|\right) \sqrt{t_{ox}}$$


© 2003-2014
All Rights Reserved.

**(2-19)** $A_i^g = (A_{g0} + A_{g1}|V_{ds}|)$

The parameters $E_{crit0}$, $E_{critg}$, $E_{critb}$, and $E_{critd}$ for calculating $V_{dsat}$ and the gate current parameters used by the Virtuoso RelXpert simulator and their default values are shown in Table 2-3.

## Temperature Gate Current Model

The temperature dependence of PMOSFET gate current is modeled using the activation energy $E_{ag}$:

**(2-20)** $$\frac{I_{gate}(T)}{I_{ds}(T)} = \frac{I_{gate}(Tnom)}{I_{ds}(Tnom)} \times \exp\left(\frac{E_{ag}}{8.62e-5}\left(\frac{1}{T} - \frac{1}{Tnom}\right)\right)$$

$T$ is the circuit operating temperature (K) set in the `.temp` statement. $T_{nom}$ is the nominal temperature (K) of the model (set using `.option tnom` or `.model tref`). $E_{ag}$ (in eV) is the activation energy.

The temperature model compares the ratio of $I_{gate}/I_{ds}$ at two temperatures rather than $I_{gate}$ alone. This is because the gate current itself is changing with temperature.

The parameters for gate current used by the Virtuoso RelXpert simulator and their default values are shown in Table 2-3.

**Table 2-3 Default Parameters for PMOSFET Gate Current**

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $A_{0i}^g$ | Ag0 | 1/cm | 3.2 |
| $A_i^g$ | Agi | 1/cm | 0 |
| $B_i$ | Bi[a] | V/cm | 3.7 x $10^6$ |
| $l_{c0}^g$ | lcg0 | $\mu m^{1/2}$ | 1.00 |
| $l_{c1}^g$ | lcg1 | $\mu m^{1/2}$/V | 0.0 |
| $Eag$ | Eag | eV | 0.0 |

a Values for $B_i$ cannot be changed.


© 2003-2014 All Rights Reserved.

The parameters can have length-dependent sensitivity parameters. These sensitivity parameter names are the parameter names in the above table with prefix `l` or `w`: `lAg0`, `lAgi`, `llcg0`, `llcg1`, `wAg0`, `wAgi`, `wlcg0`, `wlcg1`, `pAg0`, `pAgi`, `plcg0`, `plcg1`. The final value of any parameter is based on the SPICE model binning equation:

**(2-21)** For bsim3v3

$$P = P_0 + P_l \cdot \left(\frac{1}{L_{eff}} - \frac{1}{L_{ref}}\right) + P_w \cdot \left(\frac{1}{W_{eff}} - \frac{1}{W_{ref}}\right) + P_p \cdot \left(\frac{1}{L_{eff}} - \frac{1}{L_{ref}}\right)\left(\frac{1}{W_{eff}} - \frac{1}{W_{ref}}\right)$$

**(2-22)** For bsim4

$$P = P_0 + \frac{P_l}{L_{eff}} + \frac{P_w}{W_{eff}} + \frac{P_p}{(L_{eff}W_{eff})}$$

**(2-23)** For BSIMCMG

$$P = P_0 + \frac{1.0e-6}{L_{eff} + DLBIN} \cdot P_l + \frac{1.0}{NFIN} \cdot P_n + \frac{1.0e-6}{NFIN \cdot (L_{eff} + DLBIN)} \cdot P_p$$

where $P_0$ is the length and width independent parameter. $P_l$, $P_w$, and $P_p$ are length and width dependent parameters respectively. $L_{eff}$ and $W_{eff}$ are the effective channel length and width from SPICE in µm. $L_{ref}$ and $W_{ref}$ are the reference channel length and width in the SPICE parameter.

# Hot-Carrier Lifetime Model

This section describes the model used to predict Hot-Carrier induced degradation from substrate/gate current calculated from "MOSFET Substrate and Gate Current Models" on page 23.

Hot-Carrier induced device degradation in MOSFET is usually measured by the change in transconductance $\Delta g_m/g_m$ , drain current $\Delta I_d/I_d$ , and threshold voltage shift $\Delta V_t$ . This discussion generalizes the degradation by using the symbol $\Delta D$, which may be replaced by any of the above quantities or other transistor parametric shifts in the following equations.


© 2003-2014
All Rights Reserved.

## NMOSFET DC Lifetime Model

For the n-channel MOSFET under DC stress condition, the amount of degradation as a function of time is given by:

**(2-24)**

$$\Delta D = f_N(At)$$

The function $f_N(x)$ is often approximated with $x^n$. In fact, the Virtuoso RelXpert simulator calculates transistor degradation and lifetime using $\Delta D = (At)^n$. This will cause an error in the degradation and lifetime results when there is a saturation in the degradation at long stress time (see “Extracting Hot-Carrier Lifetime Parameters” on page 49).

The proportionality constant $A$ describes the degradation rate as a function of Hot-Carriers in the MOSFET channel and is related to substrate current:

**(2-25)** $$A = \frac{I_{ds}}{HW}\left(\frac{I_{sub}}{I_{ds}}\right)^m$$

or substituted in 2-24:

**(2-26)**

$$\Delta D = \left[\frac{I_{ds}}{HW}\left(\frac{I_{sub}}{I_{ds}}\right)^m \cdot t\right]^n$$

$H$ is a constant, $W$ is the device width. The exponent $n$ is ~0.3 for reaction limited rate. $m$ is the acceleration factor and is ~3.0. Using 2-25, the Virtuoso RelXpert simulator can predict degradation or lifetime if either is given and the substrate and drain currents (or their effective average under AC conditions) are known.

## PMOSFET DC Lifetime Model

For the p-channel MOSFET under DC stress condition, the amount of degradation as a function of time is given by[4]:

**(2-27)** $$\Delta D = f_p(Bt)$$


© 2003-2014 All Rights Reserved.

The function $f_p(x)$ is often approximated with $x^n$. In fact, the Virtuoso RelXpert simulator calculates transistor degradation and lifetime using $\Delta D = (Bt)^n$ (the exponent $n$ for PMOSFET is generally not equal to the $n$ for NMOSFET). This will cause an error in the degradation and lifetime results when there is a saturation in the degradation at long stress time (see "Extracting Hot-Carrier Lifetime Parameters" on page 49).

The expression for $B$ is

**(2-28)** $$B = W_g \times \frac{1}{H_g}\left(\frac{I_{gate}}{W}\right)^{m_g} + (1 - W_g) \times \frac{I_{ds}}{HW}\left(\frac{I_{sub}}{I_{ds}}\right)^{m}$$

The formula for $B$ allows the degradation in PMOSFET to be either caused by gate current injection into oxide ($W_g$=1) or channel hot electron injection ($W_g$=0). $W_g$ is the weighting coefficient.

If we assume the degradation is well modeled by the gate current (that is, $W_g$=1), 2-27 becomes:

**(2-29)**

$$\Delta D = \left[\frac{1}{H_g} \cdot \left(\frac{I_g}{W}\right)^{m_g} \cdot t\right]^n$$

$H_g$ is a constant, $W$ is the device width, and $m_g$ is the acceleration factor. Using 2-25, the Virtuoso RelXpert simulator can predict degradation or lifetime if either is given and the gate current (or its effective average under AC conditions) is known.

The parameters $H$ and $m$ in 2-25 and $H_g$ and $m_g$ in 2-28 are allowed to have a $V_{gd}$ , $V_{gs}$, $V_{ds}$, and $V_{bs}$ dependences[5]. There are two implementations of the three terminals' voltage dependence equations:

- First implementation[5]:


© 2003-2014 All Rights Reserved.

**(2-30)**

$$H = H_0 + H_{gd}V_{gd} + H_{gs}V_{gs} + H_{ds}V_{ds} + H_{bs}V_{bs}$$

$$m = m_0 + m_{gd}V_{gd} + m_{gs}V_{gs} + m_{ds}V_{ds} + m_{bs}V_{bs}$$

$$H_g = H_{g0} + H_{ggd}V_{gd} + H_{ggs}V_{gs} + H_{gds}V_{ds} + H_{gbs}V_{bs}$$

$$m_g = m_{g0} + m_{ggd}V_{gd} + m_{ggs}V_{gs} + m_{gds}V_{ds} + m_{gbs}V_{bs}$$

- Second implementation[6]:

**(2-31)**

$$H = H_0 + H_{gd}V_{gd} + H_{gs}V_{gs} + H_{ds}V_{ds} + H_{bs}V_{bs} + \sum_{i=1}^{9} H_{gdi}V^{i+1}{}_{gd} + \sum_{i=1}^{9} H_{gsi}V^{i+1}{}_{gs} + \sum_{i=1}^{9} H_{dsi}V^{i+1}{}_{gdi} + \sum_{i=1}^{9} H_{bsi}V^{i+1}{}_{bs}$$

$$m = m_0 + m_{gd}V_{gd} + m_{gs}V_{gs} + m_{ds}V_{ds} + m_{bs}V_{bs} + \sum_{i=1}^{9} m_{gdi}V^{i+1}{}_{gd} + \sum_{i=1}^{9} m_{gsi}V^{i+1}{}_{gs} + \sum_{i=1}^{9} m_{dsi}V^{i+1}{}_{gdi} + \sum_{i=1}^{9} m_{bsi}V^{i+1}{}_{bs}$$

$$mg = mg_0 + mg_{gd}V_{gd} + mg_{gs}V_{gs} + mg_{ds}V_{ds} + mg_{bs}V_{bs} + \sum_{i=1}^{9} mg_{gdi}V^{i+1}{}_{gd} + \sum_{i=1}^{9} mg_{gsi}V^{i+1}{}_{gs} + \sum_{i=1}^{9} mg_{dsi}V^{i+1}{}_{gdi} + \sum_{i=1}^{9} mg_{bsi}V^{i+1}{}_{bs}$$

$$Hg = Hg_0 + Hg_{gd}V_{gd} + Hg_{gs}V_{gs} + Hg_{ds}V_{ds} + Hg_{bs}V_{bs} + \sum_{i=1}^{9} Hg_{gdi}V^{i+1}{}_{gd} + \sum_{i=1}^{9} Hg_{gsi}V^{i+1}{}_{gs} + \sum_{i=1}^{9} Hg_{dsi}V^{i+1}{}_{gdi} + \sum_{i=1}^{9} Hg_{bsi}V^{i+1}{}_{bs}$$

The Virtuoso RelXpert simulator supports both implementations. For the second implementation, the Virtuoso RelXpert simulator supports up to 10th power of $V_{gd}$, $V_{gs}$, $V_{ds}$ , and $V_{bs}$. The default is the first implementation. However, if any of the $H_{gd}$, $m_{gd}$, $Hg_{gd}$, $mg_{gd}$; $H_{gs}$, $m_{gs}$, $Hg_{gs}$, $mg_{gs}$, $H_{ds}$, $m_{ds}$, $Hg_{ds}$, $mg_{ds}$ or $H_{bs}$, $m_{bs}$, $mg_{bs}$, $Hg_{bs}$ parameters is present, the respective equation in the second implementation will be used.

The Hot-Carrier lifetime parameters used by the Virtuoso RelXpert simulator and their default values are shown in Table 2-4.

**Table 2-4 Default Values for MOSFET Hot-Carrier Degradation Lifetime**

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $H_0$[a] | H0 | $C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{gd}$[1] | Hgd | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{gd1}$~$H_{gd9}$[b] | Hgd1~Hgd9 | - | 0.0 |
| $H_{gs}$ | Hgs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |


© 2003-2014 All Rights Reserved.

**Table 2-4 Default Values for MOSFET Hot-Carrier Degradation Lifetime,** *continued*

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $H_{gs1}$~$H_{gs9}$ | Hgs1~Hgs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ds}$ | Hds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ds1}$~$H_{ds9}$ | Hds1~Hds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_0$ | m0 | - | 3.0 |
| $m_{gd}$ | mgd | $V^{-1}$ | 0.0 |
| $m_{gd1}$~$m_{gd9}$[2] | mgd1~mgd9 | - | 0.0 |
| $m_{gs}$ | mgs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{gs1}$~$m_{gs9}$ | mgs1~mgs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{ds}$ | mds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{ds1}$~$m_{ds9}$ | mds1~mds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $n$ | nn0 | - | 0.3 |
| $H_{g0}$[1] | Hg0 | $(A/m)^{m_g} \cdot s \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ggd}$[1] | Hggd | $V^{-1} \cdot (A/m)^{m_g} \cdot s \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ggd1}$~$H_{ggd9}$[2] | Hggd1~Hggd9 | - | 0.0 |
| $Hg_{gs}$ | Hggs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{gs1}$~$Hg_{gs9}$ | Hggs1~Hggs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{ds}$ | Hgds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{ds1}$~$Hg_{ds9}$ | Hgds1~Hgds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{g0}$ | mg0 | - | 1.5 |
| $m_{ggd}$ | mggd | $V^{-1}$ | 0.0 |
| $m_{ggd1} - m_{ggd9}$[2] | mggd1~mggd9 | - | 0.0 |
| $mg_{gs}$ | mggs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{gs1}$~$mg_{gs9}$ | mggs1~mggs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{ds}$ | mgds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{ds1}$~$mg_{ds9}$ | mgds1~mgds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $W_g$ | wg | - | 1.0 |


© 2003-2014
All Rights Reserved.

**Table 2-4 Default Values for MOSFET Hot-Carrier Degradation Lifetime,** *continued*

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $H_{bs}$ | Hbs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{bs1}$~$H_{bs9}$ | Hbs1~Hbs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{bs}$ | mbs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{bs1}$~$m_{bs9}$ | mbs1~mbs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{bs}$ | mgbs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{bs1}$~$mg_{bs9}$ | mgbs1~mgbs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{bs}$ | Hgbs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{bs1}$~$Hg_{bs9}$ | Hgbs1~Hgbs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |

[a] The units depend on the quantity chosen for $\Delta D$.
[b] Applicable only in equation 2-27.

In implementation one (2-31), the parameters can have length- or width-dependent sensitivity parameters. These sensitivity parameter names are the parameter names in the above table with prefix `l`, `w`, or `p`, such as Vgd dependence part: `lH0`, `lHgd`, `lm0`, `lmgd`, `lHg0`, `lHggd`, `lmg0`, `lmggd`, `wH0`, `wHgd`, `wm0`, `wmgd`, `wHg0`, `wHggd`, `wmg0`, `wmggd`, `pH0`, `pHgd`, `pm0`, `pmgd`, `pHg0`, `pHggd`. The `Vgs` and `Vds` parts have parameter names that are similar to `Vgd`. For the The final value of any parameter is based on the SPICE model binning equation:

**(2-32)** For bsim3v3

$$P = P_0 + P_1 \cdot \left(\frac{1}{L_{eff}} - \frac{1}{L_{ref}}\right) + P_w \cdot \left(\frac{1}{W_{eff}} - \frac{1}{W_{ref}}\right) + P_p \cdot \left(\frac{1}{L_{eff}} - \frac{1}{L_{ref}}\right)\left(\frac{1}{W_{eff}} - \frac{1}{W_{ref}}\right)$$

**(2-33)** For bsim4

$$P = P_0 + \frac{P_l}{L_{eff}} + \frac{P_w}{W_{eff}} + \frac{P_p}{(L_{eff}W_{eff})}$$


© 2003-2014 All Rights Reserved.

**(2-34)** For BSIMCMG

$$P = P_0 + \frac{1.0e-6}{L_{eff} + DLBIN} \cdot P_l + \frac{1.0}{NFIN} \cdot P_n + \frac{1.0e-6}{NFIN \cdot (L_{eff} + DLBIN)} \cdot P_p$$

where $P_0$ is the length and width independent parameter. $P_l$, $P_w$, and $P_p$ are length- and width-dependent parameters respectively. $L_{eff}$ and $W_{eff}$ are the effective channel length and width from SPICE in µm. $L_{ref}$ and $W_{ref}$ are the reference channel length and width in the SPICE parameter.

Instead of using the above equations, you can select to have $H$, $m$, $H_g$, or $m_g$ values interpolated from a lookup table by specifying `.lifetime_method table`. If the lifetime parameters cannot be modelled well by a linear relationship (2-30), the table lookup method can be used to generate more accurate results. See ".deltad" on page 112 for the explanation and usage of this statement.

## AC Lifetime Model

First, let us define a parameter called $Age$. Under DC conditions, $Age$ is calculated using:

**(2-35)** $$Age(t) \equiv \frac{I_{ds}}{WH}\left(\frac{I_{sub}}{I_{ds}}\right)^m t$$

where $I_{sub}$ and $I_d$ are the substrate and drain currents respectively during stress (in A), $t$ is the stress time (in sec), $W$ is the width (in m). Both $H$ and $m$ are lifetime parameters defined in the previous section. The parameter $Age$ will now be used to quantify the amount of Hot-Carrier stress.

From 2-24, the amount of degradation $\Delta D$, is:

**(2-36)** $$\Delta D(t) = f(Age)$$

Using a quasi-static argument, under an AC bias condition $Age$ is modified to

**(2-37)**

$$Age \equiv N \int_0^T \frac{I_{ds}}{WH}\left(\frac{I_{sub}}{I_{ds}}\right)^m dt$$


© 2003-2014
All Rights Reserved.

where $I_{ds}$ and $I_{sub}$ are time-varying quantities and the integral is evaluated for any period $T$ you specify. $N$ is the number of times this period is repeated. Using 2-36 and 2-37, you can determine the amount of degradation under the AC bias condition after a given time $t$ or determine the AC lifetime $\tau$ (that is, time to reach a predefined failure level $\Delta D_f$).

For PMOSFET, $Age$ under AC conditions is a weighted sum of degradation caused by channel hot electron and gate current injection:

**(2-38)**

$$Age \equiv \int_0^T \left[ W_g \times \frac{1}{H_g}\left(\frac{I_g}{W}\right)^{m_g} + (1 - W_g) \times \frac{I_{ds}}{WH}\left(\frac{I_{sub}}{I_{ds}}\right)^{m} \right] dt$$

# Degraded Model Generation

$\Delta D$ can be quite general and not necessarily limited to transconductance, drain current degradation, or threshold voltage shift. This discussion defines $\Delta D$ as degradation in any of the transistor model parameters for SPICE simulation. The degradation $\Delta D$ as a function of $Age$ can be extended to "aging" of transistor model parameters. Thus, degraded SPICE models can be constructed for circuit aging simulation to examine circuit behavior after Hot-Carrier degradation in MOS transistors. There are two ways to go about doing this:

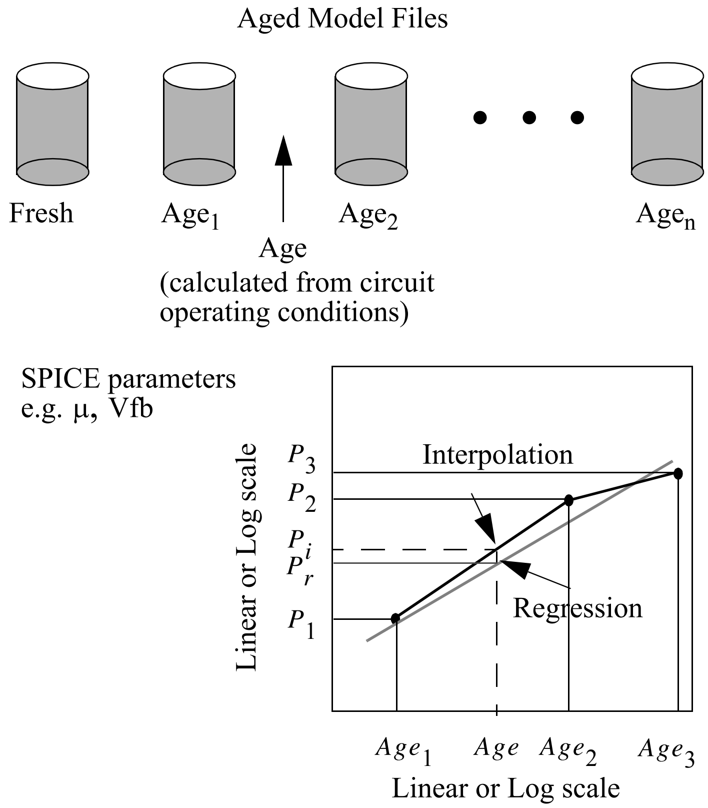
- By using aged model files
- By using the agemos model

## Aged Model Files

SPICE model parameters are extracted from a fresh device and at a number of stress intervals. These model parameters form a set of aged model files. Each of the files represents the transistor behavior after Hot-Carrier stress. The amount of stress is given by the $Age$ parameter calculated using 2-37. During the aging simulation, the Virtuoso RelXpert simulator calculates age for each individual device. Using age as a basis, the Virtuoso RelXpert simulator constructs a degraded transistor model (for each device) from the aged model files. It can do this by either interpolation or regression from these files in the linear-linear, linear-log, or log-log domain in the calculation. The aged model file method of calculating aged SPICE model parameters is shown in Figure 2-1 on page 38. The values $P_1$, $P_2$, and $P_3$ are the degraded model parameters in SPICE model files with $Age_1$, $Age_2$, and $Age_3$ respectively. $P_i$ and $P_r$ are the respective model parameters if interpolation or regression is selected. The recommended method (which is also the default) is interpolation in the log-log domain. If there is a sign change in the parameter $P$, linear-linear interpolation is suggested.


© 2003-2014 All Rights Reserved.

**Figure 2-1 Calculating Aged Model Parameters from Aged Model Files**




## Agemos Model

The Virtuoso RelXpert simulator accepts agemos parameters for BSIM3v3 and BSIM4 and supports the agemos method for aged `.model` card generation.

### “Ageable” Parameters

The degraded model parameter is a function of its fresh model parameter and its agemos parameters:

**(2-39)** $$\Delta P = f(P0, sign, age, d1, d2, n1, n2, s)$$


© 2003-2014 All Rights Reserved.

where delta $P$ is the change of parameter $P$; $P0$ is the fresh model parameter; $sign$ is the parameter changing direction; $age$ is the degradation age value; and $d1$, $d2$, $n1$, $n2$, and $s$ are agemos parameters.

**Table 2-5 Agemos Model Parameters for HCI**

| Parameter | Name Used in RelXpert | Default |
|---|---|---|
| a0 | hd1_a0 | 0.0 |
| | hd2_a0 | 0.0 |
| | hn1_a0 | 1.0 |
| | hn2_a0 | 1.0 |
| | hs_a0 | 3.0 |
| | hsign_a0 | 1.0 |
| cgdo | hd1_cgdo | 0.0 |
| | hd2_cgdo | 0.0 |
| | hn1_cgdo | 1.0 |
| | hn2_cgdo | 1.0 |
| | hs_cgdo | 3.0 |
| | hsign_cgdo | 1.0 |
| cgso | hd1_cgso | 0.0 |
| | hd2_cgso | 0.0 |
| | hn1_cgso | 1.0 |
| | hn2_cgso | 1.0 |
| | hs_cgso | 3.0 |
| | hsign_cgso | 1.0 |
| cj | hd1_cj | 0.0 |
| | hd2_cj | 0.0 |
| | hn1_cj | 1.0 |
| | hn2_cj | 1.0 |
| | hs_cj | 3.0 |
| | hsign_cj | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 2-5 Agemos Model Parameters for HCI,** *continued*

| Parameter | Name Used in RelXpert | Default |
|---|---|---|
| cjsw | hd1_cjsw | 0.0 |
| | hd2_cjsw | 0.0 |
| | hn1_cjsw | 1.0 |
| | hn2_cjsw | 1.0 |
| | hs_cjsw | 3.0 |
| | hsign_cjsw | 1.0 |
| cjswg | hd1_cjswg | 0.0 |
| | hd2_cjswg | 0.0 |
| | hn1_cjswg | 1.0 |
| | hn2_cjswg | 1.0 |
| | hs_cjswg | 3.0 |
| | hsign_cjswg | 1.0 |
| dlc | hd1_dlc | 0.0 |
| | hd2_dlc | 0.0 |
| | hn1_dlc | 1.0 |
| | hn2_dlc | 1.0 |
| | hs_dlc | 3.0 |
| | hsign_dlc | 1.0 |
| dsub | hd1_dsub | 0.0 |
| | hd2_dsub | 0.0 |
| | hn1_dsub | 1.0 |
| | hn2_dsub | 1.0 |
| | hs_dsub | 3.0 |
| | hsign_dsub | 1.0 |
| dwc | hd1_dwc | 0.0 |
| | hd2_dwc | 0.0 |
| | hn1_dwc | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 2-5 Agemos Model Parameters for HCI,** *continued*

| Parameter | Name Used in RelXpert | Default |
|---|---|---|
| | hn2_dwc | 1.0 |
| | hs_dwc | 3.0 |
| | hsign_dwc | 1.0 |
| eta0 | hd1_eta0 | 0.0 |
| | hd2_eta0 | 0.0 |
| | hn1_eta0 | 1.0 |
| | hn2_eta0 | 1.0 |
| | hs_eta0 | 3.0 |
| | hsign_eta0 | 1.0 |
| etab | hd1_etab | 0.0 |
| | hd2_etab | 0.0 |
| | hn1_etab | 1.0 |
| | hn2_etab | 1.0 |
| | hs_etab | 3.0 |
| | hsign_etab | 1.0 |
| k1 | hd1_k1 | 0.0 |
| | hd2_k1 | 0.0 |
| | hn1_k1 | 1.0 |
| | hn2_k1 | 1.0 |
| | hs_k1 | 3.0 |
| | hsign_k1 | 1.0 |
| k2 | hd1_k2 | 0.0 |
| | hd2_k2 | 0.0 |
| | hn1_k2 | 1.0 |
| | hn2_k2 | 1.0 |
| | hs_k2 | 3.0 |
| | hsign_k2 | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 2-5 Agemos Model Parameters for HCI,** *continued*

| Parameter | Name Used in RelXpert | Default |
|---|---|---|
| nfactor | hd1_nfactor | 0.0 |
| | hd2_nfactor | 0.0 |
| | hn1_nfactor | 1.0 |
| | hn2_nfactor | 1.0 |
| | hs_nfactor | 3.0 |
| | hsign_nfactor | 1.0 |
| pclm | hd1_pclm | 0.0 |
| | hd2_pclm | 0.0 |
| | hn1_pclm | 1.0 |
| | hn2_pclm | 1.0 |
| | hs_pclm | 3.0 |
| | hsign_pclm | 1.0 |
| rdsw | hd1_rdsw | 0.0 |
| | hd2_rdsw | 0.0 |
| | hn1_rdsw | 1.0 |
| | hn2_rdsw | 1.0 |
| | hs_rdsw | 3.0 |
| | hsign_rdsw | 1.0 |
| rdw | hd1_rdw | 0.0 |
| | hd2_rdw | 0.0 |
| | hn1_rdw | 1.0 |
| | hn2_rdw | 1.0 |
| | hs_rdw | 3.0 |
| | hsign_rdw | 1.0 |
| rsw | hd1_rsw | 0.0 |
| | hd2_rsw | 0.0 |
| | hn1_rsw | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 2-5 Agemos Model Parameters for HCI,** *continued*

| Parameter | Name Used in RelXpert | Default |
|---|---|---|
| | hn2_rsw | 1.0 |
| | hs_rsw | 3.0 |
| | hsign_rsw | 1.0 |
| u0 | hd1_u0 | 0.0 |
| | hd2_u0 | 0.0 |
| | hn1_u0 | 1.0 |
| | hn2_u0 | 1.0 |
| | hs_u0 | 3.0 |
| | hsign_u0 | 1.0 |
| ua | hd1_ua | 0.0 |
| | hd2_ua | 0.0 |
| | hn1_ua | 1.0 |
| | hn2_ua | 1.0 |
| | hs_ua | 3.0 |
| | hsign_ua | 1.0 |
| ub | hd1_ub | 0.0 |
| | hd2_ub | 0.0 |
| | hn1_ub | 1.0 |
| | hn2_ub | 1.0 |
| | hs_ub | 3.0 |
| | hsign_ub | 1.0 |
| uc | hd1_uc | 0.0 |
| | hd2_uc | 0.0 |
| | hn1_uc | 1.0 |
| | hn2_uc | 1.0 |
| | hs_uc | 3.0 |
| | hsign_uc | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 2-5 Agemos Model Parameters for HCI,** *continued*

| Parameter | Name Used in RelXpert | Default |
|---|---|---|
| voff | hd1_voff | 0.0 |
| | hd2_voff | 0.0 |
| | hn1_voff | 1.0 |
| | hn2_voff | 1.0 |
| | hs_voff | 3.0 |
| | hsign_voff | 1.0 |
| vsat | hd1_vsat | 0.0 |
| | hd2_vsat | 0.0 |
| | hn1_vsat | 1.0 |
| | hn2_vsat | 1.0 |
| | hs_vsat | 3.0 |
| | hsign_vsat | 1.0 |
| vth0 | hd1_vth0 | 0.0 |
| | hd2_vth0 | 0.0 |
| | hn1_vth0 | 1.0 |
| | hn2_vth0 | 1.0 |
| | hs_vth0 | 3.0 |
| | hsign_vth0 | 1.0 |

The prefix *h* indicates HCI parameters, the prefix *n* indicates NBTI parameters, and the prefix *p* indicates PBTI parameters. The Virtuoso RelXpert simulator uses these agemos parameters to generate aged (or degraded) `.model` cards in circuit simulation.

For example

```
*relxpert: +hd1_vth0 = 4.5 hd2_vth0 = 0 hn1_vth0 = 0.3 hn2_vth0 = 0.36488
*relxpert: +hs_vth0 = 1.2777 hd1_ua = 0.11812 hd2_ua = 13.12 hn1_ua = 0.2684
*relxpert: +hn2_ua = 0.50428 hs_ua = 3 hd1_ub = 372.6 hd2_ub = 1 hn1_ub = 0.44
*relxpert: +hn2_ub = 1 hs_ub = 1 hd1_a0 = 0.40162 hd2_a0 = 0 hn1_a0 = 0.08392
*relxpert: +hn2_a0 = 1 hs_a0 = 1
```


© 2003-2014 All Rights Reserved.

In this example, `vth0`, `ua`, `ub`, and `a0` change with different age values. If both `d1` and `d2` equal 0.0, the corresponding model parameter will keep constant during the whole stressing. If `d1` or `d2` does not equal 0.0, the corresponding model parameter will change with stressing.

In order to include `age` value in the aged `.model` card, add a small `age` value in the fresh `.model` card: for example,

```
*relxpert: +age = 1e-12
```

### Advantages of the Agemos Model

The agemos model has the following advantages over the aged model file method:

- The agemos method is more accurate than the aged model file method

  In the agemos method, the aged parameters at any age values can be calculated using Equation 2-39 on page 38. You do not need to do any interpolation or regression.

- The agemos method keeps the aged parameters monotonic
- Simulation with the agemos method is simpler than with the aged model file method

  No degraded `.model` cards are needed in the netlist file: Put agemos parameters in the fresh `.model` card and the aged parameters are calculated from these agemos parameters.

- Simulation with the agemos method is faster than with the aged model file method

  The aged model parameters are calculated directly, so no interpolation or regression is needed.

## Hot-Carrier Simulation

The procedures used in the Virtuoso RelXpert simulator to simulate Hot-Carrier degradation are

1. Substrate and gate current waveforms are calculated from the voltage waveforms on the terminals of the transistor.
2. From the substrate and gate current waveforms, *Age* for the transistor is calculated. Lifetime and projected degradation can then be made.
3. If aging simulation is desired, the calculated *Age* is used to create degraded SPICE model parameters for the aged SPICE netlist file. A second-pass circuit simulation on the aged circuit is performed.


© 2003-2014 All Rights Reserved.

# References

1 C. Hu, *et al*., “Hot-electron induced MOSFET degradation — model, monitor and improvement,” *IEEE Trans. Electron Devices*, vol. ED-32, p. 375, 1985.

2 P. K. Ko, *et al*., “A unified model for hot-electron currents in MOSFETs,” *IEDM Tech. Dig.*, p. 600, 1981.

3 S. Tam, *et al.*, “Lucky-electron model of channel hot-electron injection in MOSFET’s,” *IEEE Trans. Electron Devices*, vol. ED-31, p. 1116, 1984.

4 T-C. Ong, *et al*., “Hot-carrier current modeling and device degradation in surface-channel p-MOSFET’s,” *IEEE Trans. Electron Devices*, p. 1658, 1990.

5 M. M. Kuo, *et al*., “Simulation of MOSFET Lifetime under AC hot-electron stress,” *IEEE Trans. Electron Devices*, p. 1004, 1988.

6 W. J. Jiang, *et al*., “Key Hot-Carrier Degradation Model Calibration and Verification Issues for Accurate AC Circuit-Level Reliability Simulation,” IEEE International Reliability Physics Symposium, p. 300, 1997.

7 J. Y. Choi, *et al*., “Hot-carrier-induced MOSFET degradation under AC stress,” *IEEE Electron Device Letter*, EDL-8, p. 333, 1987.


© 2003-2014 All Rights Reserved.

# 3

# Extracting Hot-Carrier Parameters for MOS Transistors

This chapter describes how to extract parameters for Hot-Carrier degradation simulation in MOS transistors. The extraction task is divided into three steps:

1. Extraction of substrate and gate current parameters
2. DC stress experiment and extraction of lifetime parameters
3. Extraction of SPICE model parameters from stressed transistor I-V data

These tasks can be performed using the BSIMProPlus® (from ProPlus Design Solutions Inc) and BSIMProPlus RelET device stressing, characterization, and extraction tools. Refer to the *Virtuoso BSIMProPlus Basic Operations Guide*, *Virtuoso BSIMProPlus Model Extractor Device Modeling Guide*, and BSIMProPlus RelET manual for more details.

## Test Structures

The test structures needed for parameter extraction are MOS transistors with different channel lengths. Transistors with minimum channel lengths should be included.

## Extracting MOSFET Substrate and Gate Current Parameters

### Substrate Current Parameters

The substrate current is given by 3-1:

**(3-1)** $$I_{sub} = \frac{A_i}{B_i}(V_{ds} - V_{dsat}) I_d \exp\left(-\frac{B_i l_c}{(V_{ds} - V_{dsat})}\right)$$


All Rights Reserved.

and $V_{dsat}$ from 2-2 is[1]

**(3-2)** $$V_{dsat} = \frac{E_{crit}L_{eff}(V_{gs} - V_t)}{E_{crit}L_{eff} + V_{gs} - V_t}$$

Solve for $E_{crit}$ in and explicitly show the bias dependence:

**(3-3)** $$(E_{crit0} + E_{critg}V_{gs} + E_{critb}V_{bs} + E_{critd}V_{ds})L_{eff} = \frac{(V_{gs} - V_t)V_{dsat}}{V_{gs} - V_t - V_{dsat}}$$

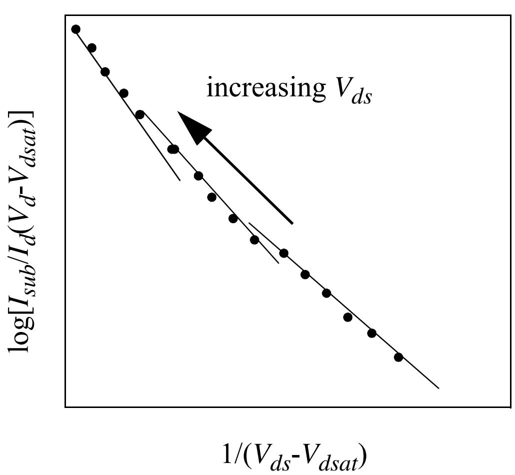
One $I_{sub}$, $I_d$ vs $V_{ds}$ for a number of $V_{gs}$ steps at zero body bias is sufficient to extract $E_{crit0}$ and $E_{critg}$. $E_{critb}$. $E_{critd}$ can be found by repeating this measurement at another fix-body bias. Usually, $E_{critb}$ is insensitive to changes in body bias and can be set to zero.

Next, $Ai$ and the bias-dependent components of $l_c$, $l_{c0}$, $l_{c1}$, and $l_{c2}$ are found by plotting $I_{sub}/I_d(V_{ds} - V_{dsat})$ versus $1/(V_{ds} - V_{dsat})$ in a semi-log plot (see Figure 3-1). $B_i$ is 1.7 MV/cm for NMOSFET and 3.7 MV/cm for PMOSFET. $l_c$ is dependent on $V_{ds}$:

**(3-4)**

$$l_c = (l_{c0} + l_{c1}V_{ds} + l_{c2}V_{gs}V_{ds})\sqrt{t_{ox}}$$

**Figure 3-1 Plotting Equation 3-1**




## PMOSFET Gate Current Parameters

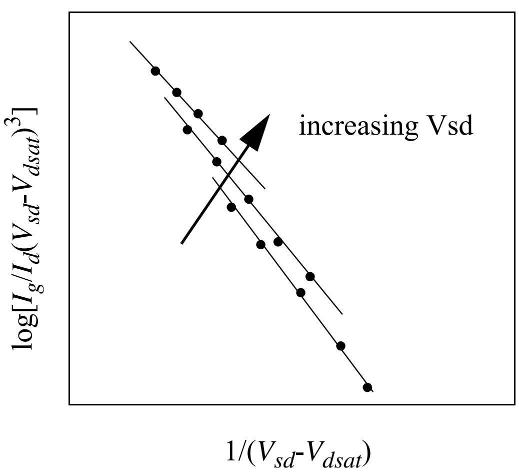
The PMOSFET gate current is given by 2-11:


© 2003-2014 All Rights Reserved.

**(3-5)** $$I_g = 0.5 I_d \frac{A_i^g}{B_i} \frac{P(E_{ox})}{\phi_b^2} (|V_{ds}| - V_{dsat})^3 \exp\left(-\left(\frac{\phi_b}{\lambda} + B_i\right) \frac{l_c^g}{(|V_{ds}| - V_{dsat})}\right)$$

$A^g{}_i$ and the bias-dependent components of $l^g{}_c$, $l^g{}_{c0}$, $l^g{}_{c1}$ are found by plotting $I_g/[I_d(V_{sd} - V_{dsat})^3 P(E_{ox})/\phi^2{}_b]$ versus $1/(V_{ds} - V_{dsat})$ in a semi-log plot (see Figure 3-2). $B_i$ is 3.7 MV/cm.

**Figure 3-2 Plotting Equation 3-5**




All the parameters can have length- and width-dependent sensitivity parameters. You can also extract length or width dependent parameters to fit the data from devices of different lengths or widths. Refer to “MOSFET Substrate and Gate Current Models” on page 23 for details.

# Extracting Hot-Carrier Lifetime Parameters

The Hot-Carrier lifetime parameters are found by measuring transistor degradation versus stress time. The degradation to be monitored can be one of the following:

- Linear transconductance, linear or saturation current degradation, or threshold voltage shift
- Lifetime $\tau$ is then defined to be the stress time required to reach a chosen level of degradation $\Delta D_f$; for example, 10% linear transconductance degradation

The Virtuoso RelXpert reliability simulator calculates the time dependence of degradation using Equation 2-24 on page 31 and the approximation $f_N(x) = x^n$:


© 2003-2014 All Rights Reserved.

**(3-6)**

$$\Delta D = (At)^n$$

In most cases, $\Delta D$ tends to saturate at longer stress time, making $n$ a function of stress time. Cadence recommends that $n$ be calculated from data at longer stress times (when saturation occurs). This will allow $\Delta D$ to be calculated accurately after longer stress intervals, but $\Delta D$ is overstated initially. In the $\Delta I_d$ model, two values of $n$ can be extracted at low and high range of $\Delta D$ to account for the saturation effect.

The error from using single value of $n$ occurs only when the Virtuoso RelXpert simulator calculates projected $\Delta D$ or lifetime. However, there is no error in the aging simulation because the more general function $f(At)$ or $f(Age)$ is used, which does not depend on $n$ (see "AC Lifetime Model" on page 36) to quantify the age dependence of degradation.

## NMOSFET Lifetime Parameters

Using the lifetime data ($\tau$) obtained under a number of stress conditions, the $m$ and $H$ parameters are calculated from the slope and intercept of $\tau \times I_{ds}$ versus $I_{sub}/I_{ds}$ curve (see Figure 3-3):

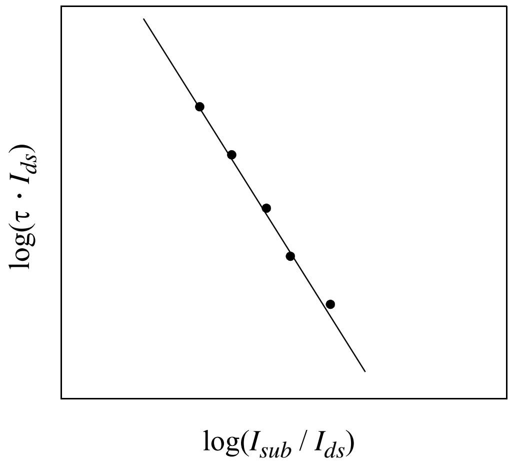
**(3-7)**

$$\tau \cdot I_d = \Delta D_f^{1/n} \cdot HW \cdot \left(\frac{I_{sub}}{I_d}\right)^{-m}$$

$\Delta D_f$ is the predefined criterion for lifetime (it can be transconductance degradation, drain current degradation, or threshold voltage shift), $W$ is the width of transistor (in m). Because both $m$ and $H$ are allowed to have $V_{dg}$ dependence, the gate-drain voltage can be varied in the stress experiment to examine and extract this dependence using Equation 2-30 on page 33.


© 2003-2014 All Rights Reserved.

**Figure 3-3 Plotting Equation 3-7**




3-7 can be rewritten as

**(3-8)**

$$\tau = \Delta D_f^{1/n} \cdot H \cdot \left(\frac{I_d}{W}\right)^{m-1}\left(\frac{I_{sub}}{W}\right)^{-m}$$

Because $I_d/W$ is roughly a constant at the stress biases and of known values, a $\log\tau$ versus $\log(I_{sub}/W)$ plot can also yield the $H$ and $m$ values.

## PMOSFET Lifetime Parameters

You can use $I_{sub}$ or $I_g$ in the PMOSFET degradation model (Equation 2-28 on page 32). If the degradation correlates with $I_{sub}$ (that is, $W_g = 0$) in Equation 2-28 on page 32), the $m$ and $H$ parameters can be extracted as in the case for NMOSFET. If the $I_g$ model is chosen, the parameters are calculated from the slope and intercept of $\log\tau$ vs l$\log(I_g/W)$ (see Figure 3-4):

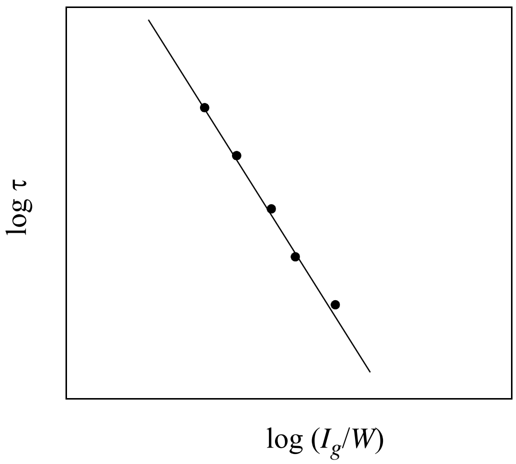
**(3-9)**

$$\tau = \Delta D_f^{1/n} \cdot H_g \cdot \left(\frac{I_g}{W}\right)^{-m_g}$$

$\Delta D_f$ is the predefined criterion for lifetime (it can be transconductance degradation, drain current degradation, or threshold voltage shift), $W$ is the width of transistor (in m). Because both $m_g$ and $H_g$ are allowed to have $V_{gd}$ bias-dependent terms, the gate-drain voltage can be varied in the stress experiment to examine and extract this dependence using Equation 2-30 on page 33.


© 2003-2014 All Rights Reserved.

**Figure 3-4  Plotting Equation 3-9**




All the parameters can have length- and width-dependent sensitivity parameters. You can extract additional length or width parameters to fit devices of different lengths and widths. See "Hot-Carrier Lifetime Model" on page 30 for details.

# Building Aged Model Files

The aged model files are built by extracting transistor SPICE model parameters from one device under stress at a number of intervals. At each time interval, $Age$ for the device is calculated from the following equations:

- For NMOSFET

**(3-10)** $$Age = \frac{I_{ds}}{HW}\left(\frac{I_{sub}}{I_{ds}}\right)^{m} t$$

- For PMOSFET

**(3-11)** $$Age = \left[W_g \times \frac{1}{H_g}\left(\frac{I_g}{W}\right)^{mg} + (1 - W_g) \times \frac{I_{ds}}{HW}\left(\frac{I_{sub}}{I_{ds}}\right)^{m}\right] t$$

To ensure that the simulator interpolates aged parameter from aged model files, rather than extrapolates from the last aged model (which can cause significant error), Cadence recommends that you run a preliminary Virtuoso RelXpert simulation with fresh model parameters, which will estimate the highest $Age$ value in the circuit. Stress experiments can then be planned such that the final aged model extracted from the stress device has at least this $Age$. Stress experiments should also include very small value of $Age$, so that small


© 2003-2014 All Rights Reserved.

degradation can be simulated accurately. *Age* for fresh models is usually set to a small and non-zero value (for example, 1e-12) to avoid underflow in the simulation.

If length- and width-sensitivity factors of transistor model parameters (as in BSIM3) are to be extracted from transistors of different lengths and widths, the extraction must be done using degraded I-V data from these transistors having the same *Age* value (by varying stress time according to 3-10 and 3-11). For digital circuits, the circuit sensitivity due to Hot-Carrier stress is mostly governed by the degradation in the shortest channel length transistors. Therefore, it is reasonable to extract aged model file from the shortest device.

# References

1 T. Y. Chan, *et al*., “A Simple method to characterize substrate current in MOSFET’s,” *IEEE Electron Device Letter,* vol. EDL-5, p.505, 1984.


© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 4

# Negative Bias Temperature Instability Degradation Model for PMOS Transistors

Negative bias temperature instability (NBTI) is simulated similar to Hot-Carrier injection (HCI), but with different lifetime parameters and degraded model sets. If NBTI lifetime parameters are specified in the `.model` card, NBTI effects will be simulated; otherwise, they are skipped. Both NBTI and HCI effects can be simulated together or independently.

**Note:** To calculate the total lifetime, use the `Lifetime=1/(1/HciLifetime + 1/NbtiLifetime)` equation.

To simulate NBTI, the following models are needed:

- NBTI lifetime model parameters
- For the aged model file method, NBTI degraded SPICE `.model` cards
- For the agemos method, agemos parameters in the fresh `.model` card

To simulate NBTI recovery, NBTI recovery model parameters listed in Table 4-3 on page 63 are needed.

The NBTI lifetime model parameters and the agemos parameters, degraded models, or NBTI recovery model parameters can be extracted by the BSIMProPlus® model extractor from ProPlus Design Solutions Inc. Refer to the *Virtuoso BSIMProPlus Basic Operations Guide* or *Virtuoso BSIMProPlus Model Extractor Device Modeling Guide* for additional information on extraction and the models.


© 2003-2014
All Rights Reserved.

# NBTI Lifetime Model

The following are NBTI lifetime model parameters:

**Table 4-1 NBTI Lifetime Model Parameters**

| Parameter | Default |
|---|---|
| nbemod | 1 |
| nba | none |
| nbeta0 | none |
| nbeta1 | none |
| nbeta2 | none |
| nbeta3 | none |
| nbea | none |
| nbgamma | none |
| nbgammab | 0 |
| nbgammad | none |
| nbalpha | none |
| nbn | none |

When nbemod=1, the lifetime is a function of (1/Vg, L, Temperature, Time)

**(4-1)**

$$Age(NBTI)) = f\left(\frac{1}{V_g}, L, Temperature, Time, NBTIparameters\right)$$

When nbemod=2, the lifetime is a function of (Vg, L, Temperature, Time)

**(4-2)**

$$Age(NBTI) = f\left(V_g, L, Temperature, Time, NBTIparameters\right)$$


© 2003-2014 All Rights Reserved.

Except the parameter `nbemod`, all other parameters are binnable.

NBTI lifetime model parameters should be included in the fresh `.model` card similar to HCI lifetime parameters or $I_{sub}$ parameters.

For example:

```
.model pmos pmos level=53 …
…
*relxpert: +nba=0.004162166 nbeta0=1 nbeta1=0.0 nbeta2=0.0 nbea=0.15
*relxpert: +nbgamma=1.6616 nbn=0.25 nbemod=2
*relxpert: +age=1e-12
```

# NBTI Degraded Models

The degraded models are extracted from stress results. These model files should be specified in the `.nbtiageproc` statement. The `.nbtiageproc` statement is similar to the `.ageproc` statement for HCI simulation.

For example:

```
*relxpert: .nbtiageproc PCH files = model/pch model/pch.1
*relxpert: +model/pch.2 model/pch.3 model/pch.4
```

# NBTI Agemos Method

The aged `.model` cards will be generated automatically by the Virtuoso RelXpert reliability simulator if you put NBTI agemos parameters in the fresh `.model` card. Agemos parameters can be extracted through the BSIMProPlus® model extractor from ProPlus Design Solutions Inc. NBTI agemos parameters are shown in Table 4-2.

**Table 4-2 Agemos Model Parameters for NBTI**

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| a0 | nd1_a0 | 0.0 |
| | nd2_a0 | 0.0 |
| | nn1_a0 | 1.0 |
| | nn2_a0 | 1.0 |
| | ns_a0 | 3.0 |
| | nsign_a0 | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 4-2 Agemos Model Parameters for NBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| cgdo | nd1_cgdo | 0.0 |
| | nd2_cgdo | 0.0 |
| | nn1_cgdo | 1.0 |
| | nn2_cgdo | 1.0 |
| | ns_cgdo | 3.0 |
| | nsign_cgdo | 1.0 |
| cgso | nd1_cgso | 0.0 |
| | nd2_cgso | 0.0 |
| | nn1_cgso | 1.0 |
| | nn2_cgso | 1.0 |
| | ns_cgso | 3.0 |
| | nsign_cgso | 1.0 |
| cj | nd1_cj | 0.0 |
| | nd2_cj | 0.0 |
| | nn1_cj | 1.0 |
| | nn2_cj | 1.0 |
| | ns_cj | 3.0 |
| | nsign_cj | 1.0 |
| cjsw | nd1_cjsw | 0.0 |
| | nd2_cjsw | 0.0 |
| | nn1_cjsw | 1.0 |
| | nn2_cjsw | 1.0 |
| | ns_cjsw | 3.0 |
| | nsign_cjsw | 1.0 |
| cjswg | nd1_cjswg | 0.0 |
| | nd2_cjswg | 0.0 |


© 2003-2014 All Rights Reserved.

**Table 4-2 Agemos Model Parameters for NBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | nn1_cjswg | 1.0 |
| | nn2_cjswg | 1.0 |
| | ns_cjswg | 3.0 |
| | nsign_cjswg | 1.0 |
| dlc | nd1_dlc | 0.0 |
| | nd2_dlc | 0.0 |
| | nn1_dlc | 1.0 |
| | nn2_dlc | 1.0 |
| | ns_dlc | 3.0 |
| | nsign_dlc | 1.0 |
| dsub | nd1_dsub | 0.0 |
| | nd2_dsub | 0.0 |
| | nn1_dsub | 1.0 |
| | nn2_dsub | 1.0 |
| | ns_dsub | 3.0 |
| | nsign_dsub | 1.0 |
| dwc | nd1_dwc | 0.0 |
| | nd2_dwc | 0.0 |
| | nn1_dwc | 1.0 |
| | nn2_dwc | 1.0 |
| | ns_dwc | 3.0 |
| | nsign_dwc | 1.0 |
| eta0 | nd1_eta0 | 0.0 |
| | nd2_eta0 | 0.0 |
| | nn1_eta0 | 1.0 |
| | nn2_eta0 | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 4-2 Agemos Model Parameters for NBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | ns_eta0 | 3.0 |
| | nsign_eta0 | 1.0 |
| etab | nd1_etab | 0.0 |
| | nd2_etab | 0.0 |
| | nn1_etab | 1.0 |
| | nn2_etab | 1.0 |
| | ns_etab | 3.0 |
| | nsign_etab | 1.0 |
| k1 | nd1_k1 | 0.0 |
| | nd2_k1 | 0.0 |
| | nn1_k1 | 1.0 |
| | nn2_k1 | 1.0 |
| | ns_k1 | 3.0 |
| | nsign_k1 | 1.0 |
| k2 | nd1_k2 | 0.0 |
| | nd2_k2 | 0.0 |
| | nn1_k2 | 1.0 |
| | nn2_k2 | 1.0 |
| | ns_k2 | 3.0 |
| | nsign_k2 | 1.0 |
| nfactor | nd1_nfactor | 0.0 |
| | nd2_nfactor | 0.0 |
| | nn1_nfactor | 1.0 |
| | nn2_nfactor | 1.0 |
| | ns_nfactor | 3.0 |
| | nsign_nfactor | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 4-2 Agemos Model Parameters for NBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| pclm | nd1_pclm | 0.0 |
| | nd2_pclm | 0.0 |
| | nn1_pclm | 1.0 |
| | nn2_pclm | 1.0 |
| | ns_pclm | 3.0 |
| | nsign_pclm | 1.0 |
| rdsw | nd1_rdsw | 0.0 |
| | nd2_rdsw | 0.0 |
| | nn1_rdsw | 1.0 |
| | nn2_rdsw | 1.0 |
| | ns_rdsw | 3.0 |
| | nsign_rdsw | 1.0 |
| rdw | nd1_rdw | 0.0 |
| | nd2_rdw | 0.0 |
| | nn1_rdw | 1.0 |
| | nn2_rdw | 1.0 |
| | ns_rdw | 3.0 |
| | nsign_rdw | 1.0 |
| rsw | nd1_rsw | 0.0 |
| | nd2_rsw | 0.0 |
| | nn1_rsw | 1.0 |
| | nn2_rsw | 1.0 |
| | ns_rsw | 3.0 |
| | nsign_rsw | 1.0 |
| u0 | nd1_u0 | 0.0 |
| | nd2_u0 | 0.0 |


© 2003-2014 All Rights Reserved.

**Table 4-2 Agemos Model Parameters for NBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | nn1_u0 | 1.0 |
| | nn2_u0 | 1.0 |
| | ns_u0 | 3.0 |
| | nsign_u0 | 1.0 |
| ua | nd1_ua | 0.0 |
| | nd2_ua | 0.0 |
| | nn1_ua | 1.0 |
| | nn2_ua | 1.0 |
| | ns_ua | 3.0 |
| | nsign_ua | 1.0 |
| ub | nd1_ub | 0.0 |
| | nd2_ub | 0.0 |
| | nn1_ub | 1.0 |
| | nn2_ub | 1.0 |
| | ns_ub | 3.0 |
| | nsign_ub | 1.0 |
| uc | nd1_uc | 0.0 |
| | nd2_uc | 0.0 |
| | nn1_uc | 1.0 |
| | nn2_uc | 1.0 |
| | ns_uc | 3.0 |
| | nsign_uc | 1.0 |
| voff | nd1_voff | 0.0 |
| | nd2_voff | 0.0 |
| | nn1_voff | 1.0 |
| | nn2_voff | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 4-2 Agemos Model Parameters for NBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | ns_voff | 3.0 |
| | nsign_voff | 1.0 |
| vsat | nd1_vsat | 0.0 |
| | nd2_vsat | 0.0 |
| | nn1_vsat | 1.0 |
| | nn2_vsat | 1.0 |
| | ns_vsat | 3.0 |
| | nsign_vsat | 1.0 |
| vth0 | nd1_vth0 | 0.0 |
| | nd2_vth0 | 0.0 |
| | nn1_vth0 | 1.0 |
| | nn2_vth0 | 1.0 |
| | ns_vth0 | 3.0 |
| | nsign_vth0 | 1.0 |

**Table 4-3 Model Parameters for NBTI Recovery**

| Parameters | Default Value |
|---|---|
| nbrnvgeff | 1e30 |
| nbrvdsth | -0.5 |
| nbrvgeffth | -0.5 |
| nbreaa | 1e30 |
| nbreab | 1e30 |
| nbraag | 1e30 |
| nbrbag | 1e30 |
| nbrabg | 1e30 |


© 2003-2014 All Rights Reserved.

**Table 4-3 Model Parameters for NBTI Recovery**

| Parameters | Default Value |
|---|---|
| nbrbbg | 1e30 |
| nbral | 1e30 |
| nbrbl | 1e30 |
| nbrn | 1.0 |
| nbrmod | 1.0 |


© 2003-2014 All Rights Reserved.

# 5

# Positive Bias Temperature Instability Degradation Model for NMOS Transistors

Positive bias temperature instability (PBTI) is simulated similar to Hot-Carrier injection (HCI), but with different lifetime parameters and degraded model sets. If PBTI lifetime parameters are specified in the `.model` card, PBTI effects will be simulated; otherwise, they are skipped. Both PBTI and HCI effects can be simulated together or independently.

**Note:** To calculate the total lifetime, use the `Lifetime=1/(1/HciLifetime + 1/PbtiLifetime)` equation.

To simulate PBTI, the following models are needed:

- PBTI lifetime model parameters
- For the aged model file method, PBTI degraded SPICE `.model` cards
- For the agemos method, agemos parameters in the fresh `.model` card

Both the PBTI lifetime model parameters and the agemos parameters, or degraded models, can be extracted by the BSIMProPlus® model extractor from ProPlus Design Solutions Inc.

## PBTI Lifetime Model

The following are PBTI lifetime model parameters:

**Table 5-1 PBTI Lifetime Model Parameters**

| Parameter | Default |
|---|---|
| pbemod | 1 |
| pba | none |
| pbbeta0 | none |
| pbbeta1 | none |


© 2003-2014 All Rights Reserved.

**Table 5-1 PBTI Lifetime Model Parameters,** *continued*

| Parameter | Default |
|---|---|
| pbbeta2 | none |
| pbbeta3 | none |
| pbea | none |
| pbgamma | none |
| pbgammad | none |
| pbalpha | none |
| pbn | none |

**Note:** Because there are some parameters named `pbeta0`, `pbeta1`, and so on, in the UC Berkeley BSIM3v3 or BSIM4 model syntax, the RelXpert software specially names the PBTI lifetime parameters as `pbbeta0`, `pbbeta1`, `pbbeta2`, and `pbbeta3`.

When pbemod=1, the lifetime is a function of (1/Vg, L, Temperature, Time)

**(5-1)**

$$Age(PBTI)) = f\left(\frac{1}{V_g}, L, Temperature, Time, PBTIparameters\right)$$

When pbemod=2, the lifetime is a function of (Vg, L, Temperature, Time)

**(5-2)**

$$Age(PBTI) = f\left(V_g, L, Temperature, Time, PBTIparameters\right)$$

Except the parameter `pbemod`, all other ten parameters are binnable.

PBTI lifetime model parameters should be included in the fresh `.model` card similar to HCI lifetime parameters or $I_{sub}$ parameters.

For example:

```
*relxpert: .agemodel nch agelevel=2
*relxpert: +pbn=0.20225 pba=1 pbea=0.055226 pbgamma=1.6
*relxpert: +pbgammad=0.58759 pbalpha=1 pbbeta0=0.0015588 pbbeta1=20.621
```


© 2003-2014 All Rights Reserved.

```
*relxpert: +pbbeta2=-1 pbemod=2
```

# PBTI Degraded Models

The degraded models are extracted from stress results. These model files should be specified in the `.pbtiageproc` statement. The `.pbtiageproc` statement is similar to the `.ageproc` statement for HCI simulation.

For example:

```
*relxpert: .pbtiageproc PCH files = model/nch model/nch.1
*relxpert: +model/nch.2 model/nch.3 model/nch.4
```

# PBTI Agemos Method

The aged `.model` cards will be generated automatically by the Virtuoso RelXpert reliability simulator if you put PBTI agemos parameters in the fresh `.model` card. Agemos parameters can be extracted through the BSIMProPlus® model extractor. PBTI agemos parameters are shown in Table 4-2.

**Table 5-2 Agemos Model Parameters for PBTI**

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| a0 | pd1_a0 | 0.0 |
| | pd2_a0 | 0.0 |
| | pn1_a0 | 1.0 |
| | pn2_a0 | 1.0 |
| | ps_a0 | 3.0 |
| | psign_a0 | 1.0 |
| cgdo | pd1_cgdo | 0.0 |
| | pd2_cgdo | 0.0 |
| | pn1_cgdo | 1.0 |
| | pn2_cgdo | 1.0 |
| | ps_cgdo | 3.0 |
| | psign_cgdo | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 5-2 Agemos Model Parameters for PBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| cgso | pd1_cgso | 0.0 |
| | pd2_cgso | 0.0 |
| | pn1_cgso | 1.0 |
| | pn2_cgso | 1.0 |
| | ps_cgso | 3.0 |
| | psign_cgso | 1.0 |
| cj | pd1_cj | 0.0 |
| | pd2_cj | 0.0 |
| | pn1_cj | 1.0 |
| | pn2_cj | 1.0 |
| | ps_cj | 3.0 |
| | psign_cj | 1.0 |
| cjsw | pd1_cjsw | 0.0 |
| | pd2_cjsw | 0.0 |
| | pn1_cjsw | 1.0 |
| | pn2_cjsw | 1.0 |
| | ps_cjsw | 3.0 |
| | psign_cjsw | 1.0 |
| cjswg | pd1_cjswg | 0.0 |
| | pd2_cjswg | 0.0 |
| | pn1_cjswg | 1.0 |
| | pn2_cjswg | 1.0 |
| | ps_cjswg | 3.0 |
| | psign_cjswg | 1.0 |
| dlc | pd1_dlc | 0.0 |
| | pd2_dlc | 0.0 |


© 2003-2014
All Rights Reserved.

**Table 5-2 Agemos Model Parameters for PBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | pn1_dlc | 1.0 |
| | pn2_dlc | 1.0 |
| | ps_dlc | 3.0 |
| | psign_dlc | 1.0 |
| dsub | pd1_dsub | 0.0 |
| | pd2_dsub | 0.0 |
| | pn1_dsub | 1.0 |
| | pn2_dsub | 1.0 |
| | ps_dsub | 3.0 |
| | psign_dsub | 1.0 |
| dwc | pd1_dwc | 0.0 |
| | pd2_dwc | 0.0 |
| | pn1_dwc | 1.0 |
| | pn2_dwc | 1.0 |
| | ps_dwc | 3.0 |
| | psign_dwc | 1.0 |
| eta0 | pd1_eta0 | 0.0 |
| | pd2_eta0 | 0.0 |
| | pn1_eta0 | 1.0 |
| | pn2_eta0 | 1.0 |
| | ps_eta0 | 3.0 |
| | psign_eta0 | 1.0 |
| etab | pd1_etab | 0.0 |
| | pd2_etab | 0.0 |
| | pn1_etab | 1.0 |
| | pn2_etab | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 5-2 Agemos Model Parameters for PBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | ps_etab | 3.0 |
| | psign_etab | 1.0 |
| k1 | pd1_k1 | 0.0 |
| | pd2_k1 | 0.0 |
| | pn1_k1 | 1.0 |
| | pn2_k1 | 1.0 |
| | ps_k1 | 3.0 |
| | psign_k1 | 1.0 |
| k2 | pd1_k2 | 0.0 |
| | pd2_k2 | 0.0 |
| | pn1_k2 | 1.0 |
| | pn2_k2 | 1.0 |
| | ps_k2 | 3.0 |
| | psign_k2 | 1.0 |
| nfactor | pd1_nfactor | 0.0 |
| | pd2_nfactor | 0.0 |
| | pn1_nfactor | 1.0 |
| | pn2_nfactor | 1.0 |
| | ps_nfactor | 3.0 |
| | psign_nfactor | 1.0 |
| pclm | pd1_pclm | 0.0 |
| | pd2_pclm | 0.0 |
| | pn1_pclm | 1.0 |
| | pn2_pclm | 1.0 |
| | ps_pclm | 3.0 |
| | psign_pclm | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 5-2 Agemos Model Parameters for PBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| rdsw | pd1_rdsw | 0.0 |
| | pd2_rdsw | 0.0 |
| | pn1_rdsw | 1.0 |
| | pn2_rdsw | 1.0 |
| | ps_rdsw | 3.0 |
| | psign_rdsw | 1.0 |
| rdw | pd1_rdw | 0.0 |
| | pd2_rdw | 0.0 |
| | pn1_rdw | 1.0 |
| | pn2_rdw | 1.0 |
| | ps_rdw | 3.0 |
| | psign_rdw | 1.0 |
| rsw | pd1_rsw | 0.0 |
| | pd2_rsw | 0.0 |
| | pn1_rsw | 1.0 |
| | pn2_rsw | 1.0 |
| | ps_rsw | 3.0 |
| | psign_rsw | 1.0 |
| u0 | pd1_u0 | 0.0 |
| | pd2_u0 | 0.0 |
| | pn1_u0 | 1.0 |
| | pn2_u0 | 1.0 |
| | ps_u0 | 3.0 |
| | psign_u0 | 1.0 |
| ua | pd1_ua | 0.0 |
| | pd2_ua | 0.0 |


© 2003-2014 All Rights Reserved.

**Table 5-2 Agemos Model Parameters for PBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | pn1_ua | 1.0 |
| | pn2_ua | 1.0 |
| | ps_ua | 3.0 |
| | psign_ua | 1.0 |
| ub | pd1_ub | 0.0 |
| | pd2_ub | 0.0 |
| | pn1_ub | 1.0 |
| | pn2_ub | 1.0 |
| | ps_ub | 3.0 |
| | psign_ub | 1.0 |
| uc | pd1_uc | 0.0 |
| | pd2_uc | 0.0 |
| | pn1_uc | 1.0 |
| | pn2_uc | 1.0 |
| | ps_uc | 3.0 |
| | psign_uc | 1.0 |
| voff | pd1_voff | 0.0 |
| | pd2_voff | 0.0 |
| | pn1_voff | 1.0 |
| | pn2_voff | 1.0 |
| | ps_voff | 3.0 |
| | psign_voff | 1.0 |
| vsat | pd1_vsat | 0.0 |
| | pd2_vsat | 0.0 |
| | pn1_vsat | 1.0 |
| | pn2_vsat | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 5-2 Agemos Model Parameters for PBTI,** *continued*

| Parameters | Name Used in RelXpert | Default Values |
|---|---|---|
| | ps_vsat | 3.0 |
| | psign_vsat | 1.0 |
| vth0 | pd1_vth0 | 0.0 |
| | pd2_vth0 | 0.0 |
| | pn1_vth0 | 1.0 |
| | pn2_vth0 | 1.0 |
| | ps_vth0 | 3.0 |
| | psign_vth0 | 1.0 |


© 2003-2014
All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 6

# Setting Up Models for Simulation

This chapter describes the procedure for setting up the Virtuoso® RelXpert reliability simulator models for simulation.

## General Rules

Model parameters for the Virtuoso RelXpert simulator are embedded in the SPICE `.model` card and are specified the same way that SPICE parameters are specified. Cadence recommends that the Virtuoso RelXpert simulator parameters be specified in lines that begin with `*relxpert:` , which maintains compatibility with SPICE simulators. SPICE simulators treat these lines as comment lines but the Virtuoso RelXpert simulator sees these as input lines containing parameters. Therefore, this procedure allows the same models to be used for SPICE circuit simulation and Virtuoso RelXpert simulation.

Starting in the 7.1.1 release, if the control statements are written using SPICE language and start with "`.`" (such as `*relxpert: .age 10y`) in the Spectre netlist or the Spectre language context, the "`simulator lang = spice`" statement should be added before the statements. On the contrary, if the control statements are written using Spectre language and without "." (such as *relxpert: age 10y) in the Spectre netlist or the Spectre language context, the "`simulator lang = spectre`" statement is not needed.

The following example shows how to add the `simulator lang = spice` statement when the control statements are written using SPICE language in the Spectre netlist:

```
simulator lang = spice
*relxpert: .age 10y
*relxpert: .deltad 0.1
*relxpert: .agemethod agemos
simulator lang = spectre
...
simulator lang = spice
*relxpert: .agemodel nch.1 agelevel = 0
*relxpert: + ai ='330680-82290*age_ssflag+190320*age_ffflag'  ecrit0 =18333
ecritg  =0
```


© 2003-2014 All Rights Reserved.

```
simulator lang = spectre
...
```

**Note:** You need to include a space after the `:` (colon) in the `*relxpert:` prefix.

The Virtuoso RelXpert simulator parameters are either values or filenames. The format for entering values can be one of the following:

1. Numeric values, in decimal or engineering notation with or without key scale factor letters

   The supported key scale factors are f=1e-15, p=1e-12, n=1e-9, u=1e-6, m=1e-3, k=1e3, M=1e6, G=1e9.

2. Algebraic expressions enclosed in single quotation (`'`) marks

   The arithmetic operations and functions supported are +, -, *, /, sqrt(x), abs(x), sin(x), cos(x), tan(x), atan(x), sinh(x), cosh(x), tanh(x), log(x), log10(x), pow(x,y), exp(x), min(x,y), max(x,y), db(x). Parameters defined in `.param` statements are also allowed in algebraic expressions.

In cases where the filename is required, specify it relative to the current directory unless the full path is required. The filename is typed without quotation marks; if the path is required, enclose the filename in quotation marks. For example:

```
.model nchan nmos level=49
+ vt0=0.7 u0=450 tox=80
* Above line specifies SPICE parameters
*relxpert: + h0=5.0e4 nn0=0.4
*relxpert: + ecrit0=2.0e5 lc0=0.89
*relxpert: .ageproc files = 'models/nmos0.mod'
* Above three lines specify RELXPERT parameters
```

Many of the model parameters have default values, which are shown in the tables below. Some model parameters have no default, and you will need to specify values.


© 2003-2014 All Rights Reserved.

# MOSFET Hot-Carrier

## Entering Gate/Substrate Current Parameters

The gate and substrate parameters from “Substrate Current Model” on page 23 and “PMOSFET Gate Current Model” on page 26 are:

**Table 6-1 Model Parameters for Substrate and Gate Currents**

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $V_{dsat}$ Parameters | | | |
| $E_{crit0}$ | Ecrit0 | V/cm | $1.0 \times 10^4$ |
| $E_{critg}$ | Ecritg | 1/cm | 0.0 |
| $E_{critb}$ | Ecritb | 1/cm | 0.0 |
| $E_{critd}$ | Ecritd | 1/cm | 0.0 |
| Substrate Current Parameters | | | |
| $A_i$ | Ai | 1/cm | 200 (NMOSFET) |
| | | | 1000 (PMOSFET) |
| $B_i$ | Bi[a] | V/cm | $1.7 \times 10^6$ (NMOSFET) |
| | | | $3.7 \times 10^6$ (PMOSFET) |
| $l_{c0}$ | lc0 | $\mu m^{1/2}$ | 1.0 |
| $l_{c1}$ | lc1 | $\mu m^{1/2}$/V | 0.0 |
| $l_{c2}$ | lc2 | $\mu m^{1/2}$/V | 0.0 |
| Gate Current Parameters | | | |
| $A_g^0$ | Ag0 | 1/cm | 3.2 |
| $B_g^i$ | Bgi | V/cm | $6.7 \times 10^6$ |
| $l_c^g{}_0$ | lcg0 | $\mu m^{1/2}$ | 0.01 |
| $l_c^g{}_1$ | lcg1 | $\mu m^{1/2}$/V | 0.0 |
| Other parameters | | | |
| | hcilevel[b] | | |


© 2003-2014 All Rights Reserved.

[a] Values for $B_i$ cannot be changed.
[b] hcilevel = 1: $V_t$ is dependent on $V_{ds}$; hcilevel = 2: $V_t$ value is fixed at the value calculated from $V_{ds}$ = 0.

Enter the parameter names (in the second column) and values in the transistor `.model` card.

## Entering MOSFET Hot-Carrier Lifetime Parameters

The MOSFET Hot-Carrier lifetime parameters from “Hot-Carrier Lifetime Model” on page 30 are

**Table 6-2 Lifetime Parameters for MOSFET Hot-Carrier Degradation**

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $H_0$ [a] | H0 | $C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{gd}$ [1] | Hgd | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{gd1}$~$H_{gd9}$ [b] | Hgd1~Hgd9 | | 0.0 |
| $m_0$ | m0 | | 3.0 |
| $m_{gd}$ | mgd | $V^{-1}$ | 0.0 |
| $m_{gd1}$~$m_{gd9}$ [2] | mgd1~mgd9 | | 0.0 |
| $n_0$ | nn0 | | 0.3 |
| $n_{gd}$ | nngd | $V^{-1}$ | 0.0 |
| $n_{gd1}$~$n_{gd9}$ [2] | ngd1~ngd9 | | 0.0 |
| $H_{g0}$ [1] | Hg0 | $(A/m)^{m_g} \cdot s \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ggd}$ [1] | Hggd | $V^{-1} \cdot (A/m)^{m_g} \cdot s \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ggd1}$~$H_{ggd9}$ [2] | Hggd1~Hggd9 | | 0.0 |
| $m_{g0}$ | mg0 | | 1.5 |
| $m_{ggd}$ | mggd | $V^{-1}$ | 0.0 |
| $m_{ggd1}$~$m_{ggd9}$ [2] | mggd1~mggd9 | | 0.0 |
| $W_g$ | wg | | 1.0 |
| $H_{gs}$ | Hgs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{gs1}$~$H_{gs9}$ | Hgs1~Hgs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $H_{ds}$ | Hds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |


© 2003-2014 All Rights Reserved.

**Table 6-2 Lifetime Parameters for MOSFET Hot-Carrier Degradation**

| Parameters | Name Used in RelXpert | Units | Default Values |
|---|---|---|---|
| $H_{ds1}$~$H_{ds9}$ | Hds1~Hds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{gs}$ | mgs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{gs1}$~$m_{gs9}$ | mgs1~gs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{ds}$ | mds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_{ds1}$~$m_{ds9}$ | mds1~mds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{gs}$ | Hggs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{gs1}$~$Hg_{gs9}$ | Hggs1~Hggs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{ds}$ | Hgds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $Hg_{ds1}$~$Hg_{ds9}$ | Hgds1~Hgds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{gs}$ | mggs | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{gs1}$~$mg_{gs9}$ | mggs1~mggs9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{ds}$ | mgds | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $mg_{ds1}$~$mg_{ds9}$ | mgds1~mgds9 | $V^{-1} \cdot C \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |

[a] The units depend on the quantity chosen for $\Delta D$.
[b] Applicable only in (Equation 2-31 on page 33).

Enter the parameter name (in the second column) and values in the transistor `.model` card.

## Setting Up Aged Model Files for MOS Transistors

The aged model files are files that contain extracted transistor model parameters at different stages of Hot-Carrier stress. See "Hot-Carrier Degradation Model for MOS Transistors" on page 23 for details on how the aged model files are used in simulation.

**Note:** Each aged model file contains only one `.model` card.

The format is exactly the same as the fresh or un-degraded transistor except that the parameters are extracted from the degraded transistor in the aged model file. Each aged model file contains the parameter $Age$, which is calculated using Equation 2-37 on page 36 for NMOSFET or Equation 2-38 on page 37 for PMOSFET. $Age$ for the fresh model file has to


© 2003-2014
All Rights Reserved.

be set to a small and non-zero number to avoid underflow if log-log or linear-log interpolation/regression is selected for aging simulation (see ".agemethod" on page 98).

In the aged model file, you can choose to enter only the model parameters that have changed compared to the fresh model. However, any parameter that appears in one aged model file must also appear in the others, although the parameter may be unchanged in the other files. For example, if the SPICE parameter `u0` is changed in `nmos2.mod` but is unchanged in `nmos1.mod`, you need to specify `u0` in `nmos1.mod` and all other aged model files.

Parameters for substrate/gate current and Hot-Carrier degradation is specified in the fresh model file only. These Virtuoso RelXpert simulator parameters are ignored if found in the aged model files.

For each transistor model reference name, a set of aged model files is prepared. The control statement to specify aged model files in Virtuoso RelXpert simulation is the `.ageproc` statement (see ".ageproc" on page 100).

For example, file `nmos0.mod` contains the following:

```
.model nchan nmos level=49
* This file contains the fresh transistor model parameters
+ vth0=0.7 u0=450 tox=80
* Above line specifies SPICE parameters
*relxpert: +h0=5.0e4 nn0=0.4
*relxpert: +ecrit0=2.0e5 lc0=0.89
*relxpert: +age=0.001
* Above lines specify RELXPERT parameters
```

and file `nmos1.mod` contains the following:

```
.model nchan nmos level=49
* This file contains the transistor model parameters after 20 min stress
+ vth0=0.65 u0=430
* Above line specifies changed SPICE parameters
*relxpert: +age=0.5
```

In the SPICE netlist file, the `.ageproc` statement is

```
*relxpert: .ageproc nchan files=nmos0.mod nmos1.mod
```

## Model-in-Model Support

By default, `relxpert_pre -age` (`prebert2`) creates an aged model card for each degraded device. As a complete model card is generated for each degraded device, the size of the aged netlist file can increase significantly.

Spectre now supports a feature that allows a global model to be shared and overridden by local models. You can define a global model in the `.model` card that contains the shared model parameters. You can then make the local models to inherit the parameters defined in


© 2003-2014 All Rights Reserved.

the global model by setting the local model master name as the name of the global model. Model parameters that override the global parameter values only need to be defined in the local models. For example:

```
.model nch_global nmos level=value0
+ toxe=value1 vth0=value1 ...
.model  nch_a  nch_global  vth0=value2
.model  nch_b  nch_global  vth0=value3
```

In the above example, `nch_global` is a global model, and `nch_a` and `nch_b` are local models whose master names have been set to `nch_global`. `nch_a` and `nch_b` models will inherit all model parameter values in the global model except the `vth0` values.

RelXpert now supports the output of model-in-model format for the aged netlist. As only model parameters changed need to be included in the local models, the size of aged netlist file is reduced considerably.

The model-in-model feature is activated when the .compact_aged_netlist simulator control statement has been set to `mod` or `all`.

## Analog Model

By default, `relxpert_pre -age` (prebert2) creates an aged model card for each degraded device. If the model belongs to a macromodel (using subckt to encapsulate the model), RelXpert generates a new degraded macromodel for each degraded device. As a complete macromodel is generated for each degraded device, the size of the aged netlist file can increase significantly for large macromodel file.

`analogmodel` is a reserved word and is feature in Spectre that allows you to bind an instance to different masters based on the value of a special instance parameter called `modelname`. An instance of `analogmodel` must have a parameter named `modelname` whose string value will be the name of the master this instance will be bound to. The value of `modelname` can be passed into subcircuits. By using the `analogmodel` feature, aged netlist generated with macromodels can be greatly reduced.

Following is an example of using the `analogmodel` feature:

**Fresh netlist**:

```
subckt fresh_sub d g s b
parameters aaa=bbb
M1 d g s b freshmodel1 w=w l=l
M2 d g s b freshmodel2 w=w l=l
End fresh_sub
X1 D1 G1 S1 B1 fresh_sub
X2 D2 G2 S2 B2 fresh_sub
```


© 2003-2014 All Rights Reserved.

Without `analogmodel`, RelXpert generates the aged netlist in the following format:

```
subckt fresh_sub d g s b
parameters aaa=bbb
M1 d g s b freshmodel1 w=w l=l
M2 d g s b freshmodel2 w=w l=l
End fresh_sub

subckt age_sub1 parameters aaa=bbb
M1 d g s b agemodel1_a1 w=w l=l
M2 d g s b agemodel2_a1 w=w l=l
End age_sub1

subckt age_sub2 parameters aaa=bbb
M1 d g s b agemodel1_a2 w=w l=l
M2 d g s b agemodel2_a2 w=w l=l
End age_sub2

X1 D1 G1 S1 B1 age_sub1
X2 D2 G2 S2 B2 age_sub2
```

With the `analogmodel` feature, RelXpert generates the aged netlist in the following format:

```
subckt fresh_sub d g s b
parameters aaa=bbb age_model_name1="aaa" age_model_name2="bbb"
M1 d g s b analogmodel modelname=age_model_name1 w=w l=l
M2 d g s b analogmodel modelname=age_model_name2 w=w l=l
End fresh_sub

X1 D1 G1 S1 B1 fresh_sub age_model_name1=agemodel1_a1 age_model_name2=agemodel2_a1
X2 D2 G2 S2 B2 fresh_sub age_model_name1=agemodel1_a2 age_model_name2=agemodel2_a2
```

Here, the original macromodels are modified to accept the analogmodel, and parameter value changes are passed using the `analogmodel` feature. As no new degraded macromodels are generated, the aged netlist file size is greatly reduced.

The analogmodel feature is activated when the .compact_aged_netlist simulator control statement has been set to `subckt` or `all`.


© 2003-2014 All Rights Reserved.

## Parameters for the Agemos Model

You are not required to provide agemos parameters for the agemos method. The agemos parameters (see “Agemos Model” on page 38, “NBTI Agemos Method” on page 57, and “PBTI Agemos Method” on page 67) are specified in the fresh transistor model.

Enter the parameter names (in the second column) and values in the transistor `.model` card.

For example, within the `.model` card,

```
.model nchan nmos level=54
+ vth0=0.7 u0=450 tox=80
* Above line specifies SPICE parameters
*relxpert: +h0=5.0e4 nn0=0.4
*relxpert: +ecrit0=2.0e5 lc0=0.89
*relxpert: +hd1_vth0 = 4.5 hd2_vth0 = 0 hn1_vth0 = 0.3 hn2_vth0 = 0.36488
*relxpert: +hs_vth0 = 1.2777
*relxpert: +hd1_ua = 0.11812 hd2_ua = 13.12 hn1_ua = 0.2684 hn2_ua = 0.50428
*relxpert: +hs_ua = 3
*relxpert: +hd1_ub = 372.6 hd2_ub = 1 hn1_ub = 0.44 hn2_ub = 1 hs_ub = 1
*relxpert: +hd1_a0 = 0.40162 hd2_a0 = 0 hn1_a0 = 0.08392 hn2_a0 = 1 hs_a0 = 1
*relxpert: +age=1e-14
* Above line specifies RELXPERT HCI lifetime and agemos parameters
```

And within the `.model` card,

```
.model pmos pmos level=54 …
…
* Above line specifies SPICE parameters
*relxpert: +nba=0.004162166 nbeta0=1 nbeta1=0.0 nbeta2=0.0 nbea=0.15
*relxpert: +nbgamma=1.6616 nbn=0.25 nbemod=2
*relxpert: +nd1_vth0 = 50.001 nd2_vth0 = 1.0044 nn1_vth0 = 0.62797
*relxpert: +nn2_vth0 = 0.36488 ns_vth0 = 1.2777
*relxpert: +nd1_ua = 0.11812 nd2_ua = 13.12 nn1_ua = 0.2684 nn2_ua = 0.50428
*relxpert: +ns_ua = 3
*relxpert: +nd1_ub = 372.6 nd2_ub = 1 nn1_ub = 0.44 nn2_ub = 1 ns_ub = 1
*relxpert: +age=1e-12
* Above lines specify RELXPERT NBTI lifetime and agemos parameters
```

```
.model nch nmos level=54
+ vth0=0.7 u0=450 tox=80
* Above lines specify SPICE parameters
*relxpert: +pbn=0.20225 pba=1 pbea=0.055226 pbgamma=1.6
*relxpert: +pbgammad=0.58759 pbalpha=1 pbbeta0=0.0015588 pbbeta1=20.621
*relxpert: +pbbeta2=-1 pbemod=2
*relxpert: +pd1_vth0=2 pd2_vth0=0 pn1_vth0=0.15 pn2_vth0=1
*relxpert: +ps_vth0 = 1 psign_vth0 = 1
*relxpert: +age=1e-12
* Above lines specify RELXPERT PBTI lifetime and agemos parameters
```

For Virtuoso RelXpert reliability simulator version 6.0.1 and higher, the reliability simulator accepts flexible agemos parameters and formats for Virtuoso Spectre. You can specify any SPICE parameter to be ageable, in addition to the SPICE parameters listed in ““Ageable” Parameters” on page 38. However, you are recommended to use this format to input agemos parameters.


© 2003-2014 All Rights Reserved.

For example

```
*relxpert: .agemodel pmos agelevel=0
*relxpert: +h0=5.0e4 nn0=0.4
*relxpert: +vth0=[0.1 0.2 0.3 0.5 3 1]
*relxpert: .agemodel pmos agelevel=1
*relxpert: +nba=0.004162166 nbeta0=1 nbeta1=0.0 nbeta2=0.0 nbea=0.15
*relxpert: +vth0=[0.2 0.3 0.4 0.5 3 1]


*relxpert: .agemodel nmos agelevel=2
*relxpert: +pbn=0.20225 pba=1 pbea=0.055226 pbgamma=1.6
*relxpert: +pbgammad=0.58759 pbalpha=1 pbbeta0=0.0015588 pbbeta1=20.621
*relxpert: +pbbeta2=-1 pbemod=2
*relxpert: +vth0=[2 0 0.15 1 1 1]
```

Use the format shown in the example above to simulate flexible agemos parameters.

- `agelevel=0` is used for Hot-Carrier injection (HCI) parameters
- `agelevel=1` is used for negative bias temperature instability (NBTI) parameters
- `agelevel=2` is used for positive bias temperature instability (PBTI) parameters

If there are HCI and NBTI parameters or HCI and PBTI parameters for the same SPICE model, put the parameters into a separate `.agemodel` statement.

**Note:** In the `.agemodel` statement, `pmos` is the model name but not the model master or the instance name. In the example below, the model name is `pch`, model master is `pmos`, and the instance name is `pfet`.

```
pfet (d1 g1 s1 b1) pch w=...
.model pch pmos level=49...
*relxpert: .agemodel pmos agelevel=0
```

In the ageMOS model, the following ageable parameters can be specified using a matrix format: `A0`, `CGDO`, `CGSO`, `CJ`, `CJSW`,`CJSWG`, `DLC`, `DSUB`, `DWC`, `ETA0`, `ETAB`, `K1`, `K2`, `NFACTOR`, `PCLM`, `RDSW`, `RDW`, `RSW`, `U0`, `UA`, `UB`, `UC`, `VOFF`, `VSAT`, and `VTH0`.

**Note:** Starting in the 7.1.1 release, any ageable parameters can be specified using the matrix format.

The matrix can contain a maximum of six columns arranged in the following order: `d1`, `d2`, `n1`, `n2`, `s`, and `sign`. If fewer than six columns are used, the same order must be maintained (that is, the parameters in the middle columns cannot be skipped).

For example, the parameter is `vth0` and `vth0=[val1 val2 val3 val4 val5 val6].`

HCI method:

```
hd1_vth0=val1 hd2_vth0=val2 hn1_vth0=val3 hn2_vth0=val4 hs_vth0=val5
hsign_vth0=val6
```


© 2003-2014 All Rights Reserved.

NBTI method:

```
nd1_vth0=val1 nd2_vth0=val2 nn1_vth0=val3 nn2_vth0=val4 ns_vth0=val5
nsign_vth0=val6
```

PBTI method:

```
pd1_vth0=val1 pd2_vth0=val2 pn1_vth0=val3 pn2_vth0=val4 ps_vth0=val5
psign_vth0=val6
```

**Note:** The `h`, `n`, and `p` prefixes indicate whether the HCI, NBTI, or PBTI method is being used in the ageMOS model.

In the binning model, the Reliability model card is removed from the fresh model definition and specified using `*relxpert: .agemodel`. The binning model syntax supports Virtuoso Spectre and SPICE conventions. The Virtuoso Spectre agemodel is included in the `simulator lang=spectre` block and the SPICE agemodel is included in the `simulator lang=spice` block.

All of the binning agemodel components can be specified. For binning agemodels that are not specified, the Virtuoso RelXpert reliability simulator uses the specified global agemodel.

For example, assuming the binning model is `nch`, the agemodel can also be called `nch` or `nch.#`. The `nch.#` designation has a higher priority for the reliability model definition.

```
*relxpert: .agemodel nch agelevel = 0
*relxpert: +ai=...
*relxpert: .agemodel nch.2 agelevel = 0
*relxpert: +ai=...
```

If `.agemodel nch.1` is not defined, the reliability model for `nch.1` will use the global agemodel defined by `.agemodel nch`. An error message is issued if a global agemodel is not found.

For example,

Virtuoso Spectre convention:

```
    Simulator lang=spectre
    model nch bsim3v3 {
    1: type=n ....
    2: type=n.....
    ....
    9: type=n...
    }
....
    *relxpert: agemodel nch bsim3v3 {
    *relxpert:  1: ai=...
    *relxpert:  2: ai=...
    ....
    *relxpert:  9: ai=...
    *relxpert:  }
```


© 2003-2014 All Rights Reserved.

SPICE convention:

```
    simulator lang=spice
    .model nch nmos level=49 ....
    .model nch.1 nmos level=49 ....
    .model nch.2 nmos level=49 ....
....
    *relxpert: .agemodel nch agelevel=0
    *relxpert: +ai=...
    *relxpert: .agemodel nch.2 agelevel=0
    *relxpert: +ai=...
    *relxpert: .agemodel nch.6 agelevel=0
    *relxpert: +ai=...
    *relxpert: .agemodel nch.9 agelevel=0
    *relxpert: +ai=...
```

## Parameters to Check Device Degradation

You can use the following RelXpert warning/error model parameters to check for device degradation:

| Warning Parameters | Error Parameters | Action |
|---|---|---|
| `warn_deltad` | `error_deltad` | Issue a warning or error if `bo0` degradation is greater than model value |
| `warn_dvth` | `error_dvth` | Issue a warning or error if `bt0 d_Vth` degradation is greater than model value |
| `warn_dgm` | `error_dgm` | Issue a warning or error if `bt0 d_Gm` degradation is greater than model value |
| `warn_didlin` | `error_didlin` | Issue a warning or error if `bt0 d_Idlin` degradation is greater than model value |
| `warn_didsat` | `error_didsat` | Issue a warning or error if `bt0 d_Idsat` degradation is greater than model value |
| `warn_did` | `error_did` | Issue a warning or error if `bt0 d_Idlin` or `d_Idsat` degradation is greater than model value (for spectre native usage) |
| `warn_dgds` | `error_dgds` | Issue a warning or error if `bt0 d_Gds` degradation is greater than model value |

**Note:** The RelXpert reliability simulator first checks for the error parameters and then the warning parameters.


© 2003-2014 All Rights Reserved.

- The warn_*/error_* model parameters are specified for each model (age level). If there are separate age level outputs (for example in bo0, HCI or NBTI table), each age level check is supported by only warn_deltad and error_deltad parameters.
- You must specify an absolute value for the model parameters warn_dvth, error_dvth, warn_deltad, and error_deltad. The values for all other parameters should be specified as percentages.
- For total degradation (for example in bo0, combined HCI/NBTI degradation table, bt0, vth), the RelXpert reliability simulator considers the largest values for the warn_*/ error_* parameters as the checking criteria.
- For the warn_deltad and error_deltad parameters, the total degradation check is performed only when the .combine_deg control statement is set to on, otherwise, RelXpert only checks the warn_deltad and error_deltad for each age level.
- Only the warn_didlin and warn_didsat parameters have the default value as 30%. If idlin and idsat degradation is greater than the default value, a warning is reported.

### Example

```
*relxpert: agemodel nmos agelevel=101
*relxpert: +m0=...
*relxpert: +warn_deltad=0.25  error_deltad=0.5
*relxpert: +warn_dvth=0.3  error_dvth=0.7
```

In the above example, the RelXpert reliability simulator will issue a warning if the device's total deltad is greater than 0.25 or dvth degradation is greater than 0.3, and error out if deltad is greater than 0.5 or dvth is greater than 0.7.

You can also use the .degration_check control statement to check the device for degradation. However, if warning/error model parameters are also specified along with the .degradation_check control statement, the .degradation_check control statement takes precedence over the model parameters. For more information, refer to .degradation_check on page 108.

## Obtaining Gate/Substrate Terminal Currents for Selected Models

You can use the igatemethod/isubmethod control statements to obtain the gate/substrate terminal current values for all reliability models. For more information, refer to .igatemethod on page 119 and .isubmethod on page 120. To obtain the gate/substrate terminal current values for a specific reliability model, you can use the igatemode and isubmode reliability model parameters.


© 2003-2014 All Rights Reserved.

The `igatemode` model parameter specifies the method to use for obtaining the gate terminal current for the specified reliability model. The parameter accepts 0 and 1 as values. If you specify 1, then the gate terminal current value is calculated using the internal Isub model. If you specify 0, then the gate terminal current value is calculated using the built-in SPICE model.

Similarly, the `isubmode` model parameter specifies the method to use for obtaining the substrate terminal current for the specified reliability model. The `isubmode` parameter accepts 0 and 1 as values. If you specify 1, then the substrate terminal current value is calculated using the internal Isub model. If you specify 0, then the substrate terminal current value is calculated using the built-in SPICE model.

The default value for both `igatemode` and `isubmode` model parameters is `1` (calc).

### Example

```
.model nch …

*relxpert: .agemodel nch agelevel = 0
…...igatemode =0(spice) isubmode=1(calc)…..
```

In the above example, the substrate terminal current is calculated using the internal Isub model and the gate terminal current value is calculated using the built-in SPICE model for the model `nch`.


© 2003-2014 All Rights Reserved.

# 7

# Simulator Control Statements

This chapter describes the Virtuoso® RelXpert reliability simulator control statements. These statements are used to request analyses, select model, output control, or pass other relevant information to the simulator.

## General Rules

The Virtuoso RelXpert simulator control statements are placed in the SPICE netlist file between the `.title` statement and `.end` statement. The order of the statements in the file is arbitrary. However, if the same control statement appears more than once, the statement that appears last will overwrite all previous statements except for the `.deltad` statement. Control statements can be up to 256 characters. Continuation lines can be used by beginning the line with a plus (+) sign.

The Virtuoso RelXpert simulator uses the default values if some of the control statements are not specified.

**Note:**

1. All the control statements should begin with the prefix "`*relxpert:`".
2. If the RelXpert simulator control statements are placed inside SPICE syntax block, control statements need to precede with . (dot command). For example,

```
simulator lang=spice
*relxpert: .age 10y
```

3. If the RelXpert simulator control statements are placed inside Spectre syntax block, RelXpert statements can be specified without the '.'. For example,

```
simulator lang=spectre

*relxpert: age 10y
```


© 2003-2014 All Rights Reserved.

# Summary of Control Statements

Below is a list of all Virtuoso RelXpert simulator control statements. For usage and explanation of each statement, see the description of each statement in the next few sections.

**Table 7-1 MOSFET Hot-Carrier Analysis Control Statements**

| Control Statement | Default |
|---|---|
| .accuracy | 1 |
| .age | - |
| .agemethod | interp loglog |
| .agemodellimit | - |
| .ageproc | - |
| .deltad | - |
| .hci_only | - |
| .idmethod | - |
| .igatemethod | calc |
| .maskdev | - |
| .maxdeg | - |
| .minage | 0.0 |
| .mindeg | - |
| .preset | - |
| .vmax | - |

**Table 7-2 MOSFET Hot-Carrier Input/Output Control Statements**

| Control Statement | Default |
|---|---|
| .append_device_age | - |
| .degsort | - |
| .dumpagemodel | - |
| .lifetime_method | calc |


© 2003-2014
All Rights Reserved.

**Table 7-2 MOSFET Hot-Carrier Input/Output Control Statements,** *continued*

| Control Statement | Default |
|---|---|
| .output_device_degrad | - |
| .printigate | - |
| .output_method | single |
| .printigatetotal | - |
| .printisub | - |
| .printisubtotal | - |
| .printisub_idsmax | - |
| .report_model_param_changes | |
| .vthmethod | calc |

**Table 7-3 PMOSFET NBTI Analysis Control Statements**

| Control Statement | Default |
|---|---|
| .age | - |
| .agemethod | interp loglog |
| .deltad | - |
| .maskdev | - |
| .maxdeg | - |
| .mindeg | - |
| .nbti_only | - |
| .nbtiageproc | - |
| .preset | - |
| .vmax | - |


© 2003-2014 All Rights Reserved.

**Table 7-4 PMOSFET NBTI Input/Output Control Statements**

| Control Statement | Default |
|---|---|
| .append_device_age | - |
| .degsort | - |
| .dumpagemodel | - |
| .minage | 0.0 |
| .output_device_degrad | - |
| .output_method | single |
| .report_model_param_changes | |
| .vthmethod | calc |

**Table 7-5 NMOSFET PBTI Analysis Control Statements**

| Control Statement | Default |
|---|---|
| .age | - |
| .agemethod | interp loglog |
| .deltad | - |
| .maskdev | - |
| .maxdeg | - |
| .mindeg | - |
| .pbti_only | - |
| .pbtiageproc | - |
| .preset | - |
| .vmax | - |

**Table 7-6 NMOSFET PBTI Input/Output Control Statements**

| Control Statement | Default |
|---|---|
| .append_device_age | - |


© 2003-2014 All Rights Reserved.

**Table 7-6 NMOSFET PBTI Input/Output Control Statements**

| Control Statement | Default |
|---|---|
| .degsort | - |
| .dumpagemodel | - |
| .minage | 0.0 |
| .output_device_degrad | - |
| .output_method | single |
| .report_model_param_changes | |
| .vthmethod | calc |


© 2003-2014 All Rights Reserved.

## .accuracy

```
*relxpert: .accuracy n
```

### Description

Specifies methods used in the Virtuoso RelXpert simulator when performing integration and substrate current calculation.

### Arguments

| | |
|---|---|
| *n* | When set to `1`, uses backward Euler integration and sets Isub=0 when Vgs < Vth. When set to `2`, uses trapezoidal integration and calculates Isub when Vgs < Vth. Setting `.accuracy` to `2` is more accurate, but increases simulation time when compared to `.accuracy 1`.<br>Default: `1` |

### Example

```
*relxpert: .accuracy 2
```

Specifies trapezoidal integration when performing integrations and calculates Isub for Vgs < Vth.


© 2003-2014 All Rights Reserved.

## .age

```
*relxpert: .age time
```

### Description

Specifies the time at which the transistor degradation and degraded SPICE model parameters are calculated. The degraded SPICE model parameters are used in aged circuit simulation.

**Note:** This statement must be specified to invoke MOSFET Hot-Carrier simulation.

The calculated transistor degradation can be transconductance $(\Delta g_m / g_m)$ , linear or saturation drain current $(\Delta I_d / I_d)$ , degradation or threshold voltage shift $(\Delta V_t)$ or any other degradation monitor, depending on the definition of the lifetime parameter $H$ in Equation 3-7 on page 50 or Equation 3-9 on page 51. You can also set up to eight age values.

### Arguments

| | |
|---|---|
| *time* | The number of seconds in the future at which the transistor degradation and degraded SPICE model parameters are to be calculated. Attach the suffix `y` (year), `h` (horror `m` (minute). There should no be space between the number and suffix. For example, `10m`, `1e-5sec`. |

### Example

```
*relxpert: .age 10y
*relxpert: .age 1y 2y 5y 8y 10y
```


© 2003-2014
All Rights Reserved.

## .agelevel_only

```
.agelevel_only value=[(level_value model_name) (level_value model_name) ...]
```

### Description

Specifies the age levels and the list of devices (model names) at each age level for which to perform HCI, NBTI, or PBTI reliability analysis. If list of model names is not specified, reliability analysis is performed for all the devices as per the specified age levels.

**Note:** This option provides similar functionality that is provided by the `.hci_only`, `.nbti_only` and `.pbti_only` options. If this option and either of the three `.hci_only`, `.nbti_only` and `.pbti_only` options are specified together, the `.agelevel_only` option is given higher priority and the other option(s) are ignored.

### Arguments

| | |
|---|---|
| `level_`*`value`* | Specifies the age level by using the following values:<br>■ `0`: Performs Hot-Carrier Injection (HCI) reliability analysis for the specified models.<br>■ `1`: Performs Negative Bias Temperature Instability (NBTI) reliability analysis for the specified models.<br>■ `2`: Performs Positive Bias Temperature Instability (PBTI) reliability analysis for the specified models. |
| *`model_name`* | Specifies the model names for each age level. |

### Example

```
*relxpert: .agelevel_only value=[(101 pmos1 pmos2) (112 pmos1 pmos2)]
```

Performs reliability analysis for `pmos1` and `pmos2` models with age levels `101` and `112`.

**Note:** The syntax for `.agelevel_only` has been made consistent with Spectre (`agelevel_only`). However, the following old format is still supported:

```
*relxpert: agelevel_only [level=value, mod1, mod2, …] [level=value, mod1, mod2, …]
     [level=value, mod1, mod2, …]…
```


© 2003-2014 All Rights Reserved.

## .age_macro_model

```
*relxpert: .age_macro_model <subckt_name> agelevel=<value> p1=v1 [p2=v2....]
```

### Description

Includes the RelXpert aging parameters for the macro device in a similar way (but not exactly in the same syntax) as model cards have for regular devices.

**Note:** Macro devices are implemented as subcircuits and therefore cannot have a native model card file associated with them. Instead, each macro device specified in the .macrodevice RelXpert control statement must have a new RelXpert control statement `.age_macro_model`.

It is recommended that the macro device subcircuits are immediately followed by their matching `.age_macro_model` statement.

In addition, the `.age_macro_model` statement parameters should not contain SPICE parameters (such parameters are declared in the subcircuit definition).

### Arguments

| | |
|---|---|
| `subckt_name` | Name of the subcircuits implemented as macro devices. |
| `agelevel` | Specifies the age level for the aged macro device. |
| `p1 p2` | RelXpert aging parameters. |

### Example

```
*relxpert: .age_macro_model mysubckt agelevel=1
```


© 2003-2014 All Rights Reserved.

## .agemethod

```
*relxpert: .agemethod
     { agemos |
     { interp | regres } [ method ]
     [ noeff ]
     }
```

### Description

Specifies the method for calculating degraded SPICE model parameters for aged circuit simulation (see "Aged Model Files" on page 37).

If there is a sign change in the parameter in aged model files, linear-linear interpolation is suggested.

### Arguments

| | |
|---|---|
| `agemos` | Specifies the agemos method. |
| `interp` *`method`* | Specifies the method of interpolation from aged model files.<br>Valid Values: `linlin` (linear-linear), `linlog` (linear-log), `loglog` (log-log)<br>Default: `loglog` |
| `regres` *`method`* | Specifies the method of regression from aged model files.<br>Valid Values: `linlin` (linear-linear), `linlog` (linear-log), `loglog` (log-log)<br>Default: `loglog` |
| `noeff` | By default, RelXpert performs aged parameter scaling using the effective parameters. Use this option to enable RelXpert perform scaling of aged parameters without using the effective parameters. |

### Example

```
*relxpert: .agemethod interp loglog
```


© 2003-2014 All Rights Reserved.

## .agemodellimit

```
*relxpert: .agemodellimit
     [ maxModels ]
     [ age | degradation ]
```

### Description

Specifies the number of aged models, based on age or degradation values.

This statement is useful if the number of degraded devices is large: grouping devices with similar degradation reduces the size of degraded netlist file.

### Arguments

| | |
|---|---|
| *maxModels* | The maximum number of models (not including the fresh model) to be generated. If the device degraded more than the last aged model, additional models will be generated using extrapolation from the last two aged model.<br><br>If *maxModels* is large, `age` and `degradation` produce similar results; if *maxModels* is small, the results are dissimilar. |
| `age` | Interpolates the models based on age values in log domain. This is the default. |
| `degradation` | Interpolates the models based on degradation values using linear scale. |

### Example

```
*relxpert: .agemodellimit 10
```

Generates a maximum of 10 aged models for each model in the netlist file, based on age values.

```
*relxpert: .agemodellimit 30 degradation
```

Generates a maximum of 30 aged models, based on degradation results.


© 2003-2014 All Rights Reserved.

# .ageproc

```
*relxpert: .ageproc
     mName
     files=fileName1 [ fileName2 … ]
```

### Description

Specifies aged SPICE model files (obtained experimentally) for generating HCI degraded SPICE model (see "Setting Up Aged Model Files for MOS Transistors" on page 79) using interpolation or regression method (selected through `.agemethod`).

### Arguments

| | |
|---|---|
| *mName* | The transistor model name for which the aged SPICE models are applicable. It must be the same model name used in SPICE `.model` card.<br><br>Any *mname* without a corresponding `.ageproc` statement will not be aged (that is, no degraded models will be generated). |
| *fileName1* | The model file containing the fresh model. Must contain only one `.model` card. *fileName* must be entered relative to the directory from which the Virtuoso RelXpert simulator is invoked unless the full pathname is specified. |
| *fileName2* … | The model files containing aged SPICE models. The orders of the aged SPICE model files should correspond with increasing age values (*filename1* is the fresh model file and *filenameN* is the aged model file with the highest age value). Must contain only one `.model` card. *fileName* must be entered relative to the directory from which the Virtuoso RelXpert simulator is invoked unless the full pathname is specified. |

### Example

```
*relxpert: .ageproc nmos files=model/nmos0.mod model/nmos1.mod model/nmos2.mod
```


© 2003-2014 All Rights Reserved.

## .append_device_age

```
*relxpert: .append_device_age
```

### Description

This option can be used to configure relxpert_pre -age (or prebert2) to output a different aged netlist file so that aged model cards are not output and to append device degradation age values to the MOSFET instance definition line. All devices continue to use the fresh model card in the degradation netlist file. The relxpert_pre -age (or prebert2) simulator only adds the "Age" model parameter for the degradation model.

### Arguments

None.

### Example

In this example

- Without the `.append_device_age` option in the netlist file, the degradation netlist file contains the following:

```
subckt inv_1 A Y
M1 Y A vdd! vdd! pmos_a1 W=5u L=1u
M2 Y A 0 0 nmos_a1 W=2u L=1u
ends inv_1
*Devices info: I1.M1 pmos W=5.000000e-06 L=1.000000e-06 Age=2.829232e-03
model pmos_a1 bsim4 type=p
+
*Devices info: I1.M2 nmos W=2.000000e-06 L=1.000000e-06 Age=1.239872e-02
model nmos_a1 bsim4 type=n
+
```

- The `.append_device_age` option is set in the netlist file and degradation netlist files contain the following information:

```
subckt inv_1 A Y
M1 Y A vdd! vdd! pmos W=5u L=1u Age=2.829232e-03
M2 Y A 0 0 nmos W=2u L=1u Age=1.239872e-02
ends inv_1
```


© 2003-2014 All Rights Reserved.

# .bti_recovery

```
*relxpert: .bti_recovery [yes | no] {[level=number, mod1 mod2 …] [level=number,
     mod1 mod2 …] …}
```

*Important*

Currently, RelXpert supports NBTI recovery only. Therefore, `level` can only be set to `1`.

### Description

Specifies whether to enable or disable NBTI recovery for the specified age level and the models at that age level.

**Note:** For more information on NBTI recovery models, see the *Negative Bias Temperature Instability Degradation Model for PMOS Transistor* chapter.

### Arguments

| | |
|---|---|
| `yes` | Enables NBTI recovery for the specified age level and its respective models (Default). |
| `no` | Disables NBTI recovery for the specified age level and its respective models. |
| `level` | Specifies the age level for enabling or disabling NBTI recovery. For NBTI recovery, `level` should be set to `1`. |
| `model` | Lists the models for NBTI recovery. |

**Note:** If the `yes` or `no` argument is not set, the software assumes the default value `yes`.

### Examples

```
*relxpert: .bti_recovery no
```

All the NBTI recovery models are ineffective.

```
*relxpert: .bti_recovery yes [level=1, pmos1, pmos2]
```

Enables NBTI recovery for models `pmos1` and `pmos2`.


© 2003-2014
All Rights Reserved.

## .check_neg_aging

```
*relxpert: .check_neg_aging type={warn|error} clamp={yes|no}
```

### Description

Reports the negative degradation values for a model.

### Syntax

| | |
|---|---|
| `type` | Specifies the type of message to be generated when negative aging occurs. Possible values are `error` and `warn`. The default value is `error`. |
| `clamp` | Specifies whether or not to clamp the degradation values for negative aging. Default is `no`. If set to `yes`, RelXpert clamps the degradation values to be the same as fresh values and generates a warning message for negative aging. |

### Example

```
*relxpert: .check_neg_aging type=warn clamp=yes
```


© 2003-2014
All Rights Reserved.

## .combine_deg

```
*relxpert: combine_deg [off|on] [file=file_name]
```

### Description

Determines whether to combine the degradation for all age levels in one SPICE model. The result is printed in `bo0` files.

### Arguments

| | |
|---|---|
| `off` | Does not combine the degradation for all age levels in one SPICE model (Default). |
| `on` | Combines the degradation for all age levels in one SPICE model. |
| `file_name` | Enables you to specify an output file name for degradation results. |

### Example

```
*relxpert: .combine_deg off
```

Combines the degradation for all age levels in one SPICE model.


© 2003-2014
All Rights Reserved.

## .compact_aged_netlist

```
*relxpert: .compact_aged_netlist type=[mod|subckt|all|none]
```

### Description

In RelXpert (`relxpert_pre -age`), a subcircuit is expanded if a model name is changed to to an aged model name. If the subcircuit definition is large, it causes the netlist size to become huge after expansion and it becomes extremely difficult to open the netlist. The `.compact_aged_netlist` statement uses the `analogmodel` feature of Spectre in the aged netlist to change the model name. Therefore, the subcircuit is not expanded, which reduces the netlist size considerably.

RelXpert also uses the model-in-model feature of Spectre to output an aged netlist file in which the shared model parameters are listed only once for devices sharing the same model in the netlist. This also reduces the size of the netlist to a large extent.

### Arguments

| | |
|---|---|
| `mod` | RelXpert uses the model-in-model feature of Spectre to output an aged netlist file.<br><br>See Model-in-Model Support on page 80 for more information. |
| `subckt` | RelXpert uses the analogmodel feature of Spectre to change the model name in the netlist.<br><br>See Analog Model on page 81 for more information. |
| `all` | RelXpert uses both the Spectre features to reduce the netlist size. |
| `none` | RelXpert does not use any of the Spectre features to reduce the size of the netlist file. This is the default. |

**Note:** The model-in-model and analogmodel features are supported only when the Spectre simulator is used with RelXpert.


© 2003-2014 All Rights Reserved.

## .deg_ratio

```
*relxpert: .deg_ratio { hci = hci_ratio_value nbti = nbti_ratio_value
     pbti = pbti_ratio_value [include | exclude] devices }
```

### Description

Specifies the HCI, NBTI, and PBTI degradation weighting ratio for devices in a Virtuoso Relxpert simulation. The HCI, NBTI, and PBTI ratio values are specified by using the *hci_ratio_value*, *nbti_ratio_value*, and *pbti_ratio_value*, respectively. The new degradation values for the specified devices is obtained by multiplying the original degradation value.

**Note:** If more than one `.deg_ratio` statement is specified, the latter one will take precedence.

### Arguments

| | |
|---|---|
| *hci_ratio_value* | Specifies the HCI degradation weighting ratio in total degradation. The specified value should be in the range 0.0~1.0.<br>*Default*: 1.0 |
| *nbti_ratio_value* | Specifies the NBTI degradation weighting ratio in total degradation. The specified value should be in the range 0.0~1.0.<br>*Default*: 1.0 |
| *pbti_ratio_value* | Specifies the PBTI degradation weighting ratio in total degradation. The specified value should be in the range 0.0~1.0.<br>*Default*: 1.0 |
| `include` *devices* | Applies the degradation weighting ratio for only the listed devices. |
| `exclude` *devices* | Applies the degradation weighting ratio for all devices except the ones listed. |

**Note:** The `include` and `exclude` values are mutually exclusive. In addition, if both include and exclude options are not specified, the software applies the degradation ratio on all devices.


© 2003-2014 All Rights Reserved.

### Examples

```
*relxpert: .deg_ratio [ hci = 0.5 nbti = 0.3 pbti = 0.2 include x1 x4.m1]
```

Considers the specified degradation weighting ratio for all devices in `x1` and `m1` in `x4`.

```
*relxpert .deg_ratio [hci = 0.5 nbti = 0.3 pbti = 0.2 exclude x3 x4.m1]
```

Considers the specified degradation weighting ratio for all devices except devices in `x3` and `m1` in `x4`.

```
*relxpert .deg_ratio [hci = 0.5 nbti = 0.3 pbti = 0.2]
```

Considers the specified degradation weighting ratio for all devices.


© 2003-2014 All Rights Reserved.

## .degradation_check

```
*relxpert: .degradation_check type=[warn | error] parameter=[deltad | dvth | didlin
| didsat | did | dgm | dgds] value=<degradation_value> {agelevel=agelevel_number}
{ sub = [sub1 sub2 sub3 …] | mod = [mod1 mod2 mod3…] | dev = [inst1 inst2 inst3…] }
```

### Description

The `.degradation_check` statement checks for device degradation and issues a warning or error if the device's degradation value (in `bo0` output) or `vth`, `idlin`, `idsat`, `gm`, and/or `gds` degradation (in `bt0` output) is greater than the value specified by the `value` argument.

**Note:** You can also use the RelXpert warning/error model parameters to check for device degradation. However, if warning/error model parameters are also specified along with the `.degradation_check` control statement, the `.degradation_check` control statement takes precedence over the model parameters. For more information on warning/error model parameters, refer to Parameters to Check Device Degradation on page 86.

### Arguments

| | |
|---|---|
| `type` | Specifies the message type. Possible values are `error` or `warning`. For `type=warning`, only the first five warning messages are reported. For `type=error`, the RelXpert reliability simulator terminates with the error message. |
| `parameter` | Specifies the model parameters for the device. Possible values are `deltad`, `dvth`, `ddlin`, `didsat`, `did`, `dgm`, or `dgds`.<br><br>`deltad` refers to the degradation value in the `bo0` output. `dvth`, `didlin`, `didsat`, `did`, `dgm`, `dgds` refer to $V_{th}$, $Id_{lin}$, $Id_{sat}$, either $Id_{lin}$ or $Ids_{at}$, $G_m$, or $G_{ds}$ degradation in the `bt0` output. |
| `value` | Specifies the degradation value for the model. An error or warning is issued if the degradation for the specified parameter is greater than the degradation value. |
| `agelevel` | Specifies the age level for the model. |
| `sub` | Specifies the name of the subcircuit. |
| `mod` | Specifies the list of models. |
| `dev` | Specifies the name of the instances. |


© 2003-2014 All Rights Reserved.

- You can specify only one of the `sub`, `mod`, or `dev` arguments in the control statement. A combination is not allowed. If none of these arguments is specified, then all devices are checked for degradation.
- The `agelevel` argument can be specified only for the `deltad` parameter. In addition, if agelevel is specified for the `deltad` parameter, then degradation check is performed only at that agelevel.
- Only one `degradation_check` control statement is allowed. If multiple statements are specified, only the last statement is considered by the simulator.

#### Example

```
*relxpert: .degradation_check type=warn parameter=deltad value=0.25
```

Issues a warning if any device's total `deltad` in `bo0` is greater than 0.25.

```
*relxpert: .degradation_check type=error parameter=dvth value=0.7 mod=[ nmos ]
```

Issues an error if any device with model `nmos` has `d_Vth` value greater than 0.7 in `bt0`.

```
*relxpert: .degradation_check type=warn parameter=dvth value=0.5 sub=[ x1 ]
```

Issues a warning for devices in subcircuit `x1` that have `d_Vth` value greater than 0.5 in `bt0`.


© 2003-2014 All Rights Reserved.

# .degsort

```
*relxpert: .degsort
     [ -threshold value | -number number ]
```

or

```
*relxpert: .degsort
     [ number_agelevel = [number1 agelevel1, number2 agelevel2...] |
     threshold_agelevel = [ threshold1 agelevel1, threshold2 agelevel2....] ]
```

### Description

Requests the output of all MOS transistors, sorted in the order of descending degradation. If neither `-threshold` nor `-number` is specified, all transistors will be output.

You can also use `.degsort` to output the MOS transistors for a specific age level by using the `number_agelevel` and `threshold_agelevel` arguments.

**Note:** This statement must be specified to print transistor degradation results and will overwrite any other `.degsort` statement**.**

### Arguments

| | |
|---|---|
| `-threshold` *`value`* | Prints transistors having degradation values greater than *`value`*. *`value`* can be in decimal notation (xx.xx) or in engineering notation (x.xxe+xx). |
| `-number` *`number`* | Prints the first *`number`* transistors having the highest degradations. |
| `number_agelevel` [*`number agelevel...`*] | Prints the first *`number`* transistors that have the highest degradation for each *`agelevel`*.<br><br>**Note:** *`number`* and *`agelevel`* must be specified in pairs. In addition, if `agelevel` is not specified, then all transistors are printed. |
| `threshold_agelevel` [*`threshold agelevel...`*] | Prints transistors having degradation values greater than *`threshold`* for age level *`agelevel`*.<br><br>**Note:** *`threshold`* and *`agelevel`* must be specified in pairs. In addition, if `agelevel` is not provided, then all transistors are printed. |


© 2003-2014 All Rights Reserved.

### Example

```
*relxpert: .degsort -threshold 0.1
```

Prints only the transistors having degradation greater the `0.1`.

```
*relxpert: .degsort threshold_agelevel=[0.01 0]
```

Prints only the transistors having degradation greater than `0.01` for age level `0`. For other age levels there is no limitation.

```
*relxpert: .degsort -number 100
```

Prints the first 100 transistors in decreasing order of degradation.

```
*relxpert: .degsort number_agelevel=[6 0, 7 1]
```

Prints the first 6 transistors for age level `0` and first 7 transistors for age level `1` in descending order.

### See Also

"MOSFET Hot-Carrier Simulation" on page 233.


© 2003-2014 All Rights Reserved.

## .deltad

```
*relxpert: .deltad
     value
     [ modelName ]
```

### Description

Requests the calculation of lifetime for each transistor under the circuit operating conditions.

You can use multiple `.deltad` commands for different types of transistors.

### Arguments

| | |
|---|---|
| *value* | The degradation value can be transconductance $(\Delta g_m / g_m)$, linear or saturation drain current degradation $(\Delta I_d / I_d)$, threshold voltage shift $(\Delta V_t)$, or any other degradation monitor, depending on the definitions of the lifetime parameters $H$ and $m$ in Equation 3-7 on page 50. *value* can be in decimal notation (xx.xx) or in engineering notation (x.xxe+xx). |
| *modelName* | Name of a specific model whose lifetime is calculated; must be the same as in the `.model` card. |

### Example

```
*relxpert: .deltad 0.1
```

Requests lifetime calculation for 10% transconductance change for all devices.

```
*relxpert: .deltad 0.1 nmos
*relxpert: .deltad 1e-9 pmos
```

Requests lifetime calculations for 10% saturation drain current change for nmos transistors and 1e-9A linear drain current degradation for PMOS transistors.

```
*relxpert: .deltad 0.1 nmos
*relxpert: .deltad 0.15 nmos.3
```

Includes binning models nmos(nmos.1~nmos.9). These options request lifetime calculations for 15% saturation drain current change for `nmos.3` and 10% for the other binning models.


© 2003-2014
All Rights Reserved.

## .dumpagemodel

`*relxpert: .dumpagemodel` *`filename`*

### Description

The `.dumpagemodel` option tells the Virtuoso RelXpert reliability simulator to output the degraded model into a *`filename`* file. If the netlist file contains an `.alter` statement, the output files are named `filename_0`, `filename_1` ... `filename_n`.

### Arguments

| | |
|---|---|
| *`filename`* | Degraded model card filename |


© 2003-2014 All Rights Reserved.

## .enable_bjt_device

```
*relxpert: .enable_bjt_device value=[yes|no]
```

### Description

The `.enable_bjt_device` option allows the Virtuoso RelXpert reliability simulator to enable or disable the BJT reliability model.

The BJT reliability model is implemented using URI. Refer to the *Virtuoso Unified Reliability Interface* manual for more information.

**Note:** Currently, RelXpert supports BJT devices only for the VBIC model. It ignores other BJT devices.

### Arguments

| | |
|---|---|
| `value` | If set to `yes`, the BJT reliability model is enabled. Default is `no`. |


© 2003-2014 All Rights Reserved.

## .enable_compress_waveform

```
*relxpert: .enable_compress_waveform value= [yes|no]
```

### Description

The `.enable_compress_waveform` option enables or disables the generation of compressed waveform by Spectre and allows `relxpert_post` to load the compressed waveform.

If you do not specify the `.enable_compress_waveform` statement in the netlist and the netlist contains the Spectre option `compression`, RelXpert considers the Spectre option to enable or disable compression. In addition, if `.enable_compress_waveform value=yes` and `compression=no` are specified in the netlist, the `.enable_compress_waveform value=yes` takes higher precedence over the `compression` option.

**Note:** This option is available only when Spectre is used as the simulator.

### Arguments

| | |
|---|---|
| `value` | If set to `yes`, `relxpert_post` loads the compressed waveform. Default is `no`. |


© 2003-2014 All Rights Reserved.

## .gradual_aging_pass_param

```
*relxpert: .gradual_aging_pass_param value=[yes|no]
```

### Description

The `.gradual_aging_pass_param` option, if specified in the gradual aging flow, passes the data structure of a URI instance state from a previous aging step to the next aging step. The data structure is copied to a temporary file before being passed to the next aging step.

### Arguments

| | |
|---|---|
| `yes` | Pass the data structure of a URI instance from a previous step to the next step |
| `no` | Do not pass the data structure of a URI instance from a previous step to the next step. This is the default. |


© 2003-2014 All Rights Reserved.

## .hci_only

```
*relxpert: .hci_only <mos1> <mos2...>
```

### Description

If `modelname` is not specified, the Virtuoso RelXpert reliability simulator performs only Hot-Carrier injection (HCI) on all of the devices. If specified, the simulator calculates HCI age values for devices included in the `.hci_only` statement. For models not included in the statement, the simulator calculates both HCI and NBTI degradations.

**Note:** The `.hci_only`, `.lifetime_method`, and `.nbti_only` options only work with the Cadence ageMOS model (agelevel=0, 1, 2).

### Arguments

| | |
|---|---|
| *`modelname`* | Specifies model name list (can only be a calculated HCI age value) |

### Examples

The first example

```
*relxpert: .hci_only
```

tells the Virtuoso RelXpert reliability simulator to calculate HCI for all devices.

In the next example

```
*relxpert: .hci_only pmos1
```

tells the simulator to calculate HCI for `pmos1` model devices. For all other devices, both HCI and NBTI degradation is calculated (if HCI and NBTI models are available).


© 2003-2014 All Rights Reserved.

## .idmethod

```
*relxpert: .idmethod
     { ids | idrain | idstatic}
```

### Description

Specifies how the Virtuoso RelXpert simulator obtains the drain current (Id) from the SPICE simulator for the Virtuoso RelXpert simulator calculations.

In the Virtuoso RelXpert simulator, drain currents of transistors are obtained from SPICE simulations. Two types of drain current are available from SPICE:

- Dynamic drain current (also referred to as AC drain current): this is the current flow-in to the drain node
- Static drain current (also referred to as channel drain current, DC drain current, or Ids)

The Virtuoso RelXpert simulator supports the use of either type of drain current.

### Arguments

| | |
|---|---|
| `ids` | Instructs the Virtuoso RelXpert simulator to use Ids static current. This is the default. |
| `idrain` | Instructs the Virtuoso RelXpert simulator to use dynamic drain current. |
| `idstatic` | Instructs the Virtuoso RelXpert simulator to use static drain terminal current. |


© 2003-2014
All Rights Reserved.

## .igatemethod

```
*relxpert: .igatemethod
     { calc | spice }
```

### Description

Specifies the method used for obtaining the gate terminal current of a MOSFET.

During MOSFET HCI simulation, the gate terminal current is required for calculating the degradation value. The Virtuoso RelXpert simulator can either calculate this value using internal Igate model or obtain it from the built-in SPICE model such as BSIM4 or PSP Igate model.

If this command is not used, the Virtuoso RelXpert simulator calculates the gate terminal current using internal Igate model.

### Arguments

| | |
|---|---|
| `calc` | Calculates the gate terminal current using the internal Igate model.<br><br>This is the default value. |
| `spice` | Obtains the gate terminal current value using built-in SPICE model. |

### Example

```
*relxpert: .igatemethod spice
```


© 2003-2014 All Rights Reserved.

## .isubmethod

```
*relxpert: .isubmethod { calc | spice }
```

### Description

Specifies the method used for obtaining substrate terminal current of a MOSFET.

During MOSFET HCI simulation, the substrate terminal current is required for calculating the degradation value. The Virtuoso Relxpert simulator can either calculate this value using internal Isub model or obtain it from the built-in SPICE model such as BSIM4 or PSP Isub model.

If this command is not used, the Virtuoso RelXpert simulator calculates the substrate terminal current using the internal Isub model.

### Arguments

| | |
|---|---|
| `calc` | Calculates the substrate terminal current using the internal Isub model. This is the default value. |
| `spice` | Obtains the substrate terminal current value using built-in SPICE model. |

### Example

```
*relxpert: .isubmethod spice
```

Specifies that the substrate terminal current value should be obtained from the built-in SPICE model.


© 2003-2014 All Rights Reserved.

## .lifetime_method

```
*relxpert: .lifetime_method
     [ calc | table file=fileName ]
```

### Description

Specifies the method for obtaining lifetime parameters for MOSFET Hot-Carrier simulation.

If this statement is not specified, the Virtuoso RelXpert reliability simulator calculates lifetime parameter values using lifetime model equations as described in “Hot-Carrier Lifetime Model” on page 30.

**Note:** The `.hci_only`, `.lifetime_method`, and `.nbti_only` options only work with the Cadence ageMOS model (agelevel=0, 1, 2).

### Arguments

| | |
|---|---|
| `calc` | Instructs the Virtuoso RelXpert reliability simulator to calculate these parameters (default). You will need to provide parameter values for the calculations. |
| `table file=`*`fileName`* | Instructs the reliability simulator to look up lifetime parameters and interpolate the results from *`fileName`*. The filename should contain values for lifetime parameters (that is, h, m, n, hg, and mg under different vgd, vbs, vds, vg/vd, vb/vs, or vd/vs bias conditions). Only one bias condition can be specified in the table file.<br><br>*`fileName`* must be entered relative to the directory from which the reliability simulator is invoked unless the full pathname is specified. |

### Example

```
*relxpert: .lifetime_method calc
```

The Virtuoso RelXpert simulator supports both types of equation implementations for calculating H, m, n, Hg, and mg as described in “NMOSFET DC Lifetime Model” on page 31 and “PMOSFET DC Lifetime Model” on page 31. If `.lifetime_method` is not defined or `calc` is selected, the default is to use the first implementation (Equation 2-30 on page 33)[1]. However, if any of the hgd1... hgd9, mgd1 ... mgd9, ngd1 ... ngd9, hggd1 ... hggd9, mggd1 ...


© 2003-2014
All Rights Reserved.

mggd9 parameters is used, the respective equation for the second implementation is used (Equation 2-31 on page 33)[2].

```
*relxpert: .lifetime_method table file=lookup_table
```

Instructs the Virtuoso RelXpert simulator to look up parameter values from the look-up file and use piecewise linear fitting to obtain the parameter value. Using this method, if given vgd, vds, vg/vd, or vd/vs bias, the reliability simulator obtains the h, m, n, hg, and mg lifetime parameter values which are interpolated from the table. If given vbs or vb/vs bias in the table file, the reliability simulator adds the h, m, n, hg, and mg values interpolated from the table along with the h, m, n, hg, and mg model card values.

| **Bias** | **Parameter Values** |
|---|---|
| vgd, vds, vg/vd, or vd/vs | h, m, n, hg, mg (final) = h, m, n, hg, mg (table) |
| vbs or vb/vs | h, m, n, hg, mg (final) = h, m, n, hg, mg (model) + h, m, n, hg, mg (table) |

The look-up file should list n, h, m, hg, mg values under different vgd, vbs, vds, vd/vs, vb/vs or vg/vd bias conditions. The look-up file format is as follows:

```
.model modelname
parameter1 parameter2 parameter3 ..
. . . . . {data values}
.end
```

Restrictions:

- Lines starting with `*` or `#` are treated as comments.
- Continuation characters (`+`) are not supported.
- An optional `.debug` statement can be inserted at the beginning of the file to output all interpolated results. The results are output to the `lifetime_table.lst` file.
- The keywords for specifying models are `vgd`, `vg/vd`, `vbs`, `vb/vs`, `vds`, `vd/vs`, `h`, `m`, `n`, `hg`, and `mg`.
- Only one vgd, vg/vd, vbs, vb/vs, vds, or vd/vs table can be specified with one model.
- vgd, vg/vd, vbs, vb/vs, vds, or vd/vs can be listed in ascending or descending order.
- If the model name for a device is not found in the look-up file, the device lifetime and degradation calculations are skipped.
- For PMOSFET, if wg=1, h and m values can be omitted.
- Vbs sign cannot be positive for both NMOSFET and PMOSFET.


© 2003-2014 All Rights Reserved.

For example:

```
* this line is a comment
* if a .debug is included, all interpolated results will be outputted to
"lifetime_table.lst"
.debug

.model nmos
vgd h m n
-5 8.1e6 3 0.3
-4 9.1e6 3.1 0.3
-3 1.2e7 3.4 0.3
0.5 1.3e7 3.4 0.3
1 1.35e7 3.4 0.3
3 1.4e7 3.4 0.3
5 1.5e7 3.4 0.3
.end

.model nmos1
vbs h m n
-1.1 -1.1e8 0 0
-0.9 -9.1e7 0 0
-0.7 -8.2e7 0 0
-0.5 -2.3e7 0 0
-0.3 -4.35e6 0 0
0 0 0 0
.end

.model pmos
vbs h m n
*vbs sign cannot be positive
-1.5 1e6 0 0
-1.0 2.1e6 0 0
-0.5 3.0e6 0 0
0 0 0 0
.end

.model pmos1
vgd hg mg n
* wg =1, so omit h, m values
-5 1.5e-1 1.6 0.5
-3 1.4e-1 1.6 0.5
0.4 1.4e-1 1.6 0.5
1 1.3e-1 1.5 0.5
5 1.2e-1 1.5 0.5
.end

.model pmos2
vgd h m hg mg n
* wg!=1, need to specify h,m as well
-5 1e6 3.0 1.5e-1 1.6 0.5
-3 1e6 3.1 1.4e-1 1.6 0.5
0.4 1e7 3.1 1.4e-1 1.6 0.5
1 1e7 3.1 1.3e-1 1.5 0.5
5 1e7 3.2 1.2e-1 1.5 0.5
.end

.model nmos2
vg/vd h m n
```


© 2003-2014 All Rights Reserved.

```
0 1.0e6 3.0 5e-1
1 1.2e6 3.1 5e-1
2 1.3e6 3.1 5e-1
3 1.5e6 3.2 5e-1
.end
```

The Virtuoso RelXpert reliability simulator also supports flexible vbs, vgd, and vg/vd table models format. The parameters can be set in the model card using flexible model parameter syntax. The valid parameter names are `vbs_v`, `vbs_h`, `vbs_m`, `vbs_n`, `vbs_hg`, `vbs_mg`, `vgd_v`, `vgd_h`, `vgd_m`, `vgd_n`, `vgd_hg`, `vgd_mg`, `vg_vd_v`, `vg_vd_h`, `vg_vd_m`, `vg_vd_n`, `vg_vd_hg`, and `vg_vd_mg`.

| Flexible Table Models | Parameters | Values |
|---|---|---|
| vbs | vbs_v | vbs voltages |
| | vbs_h | h |
| | vbs_m | m |
| | vbs_n | n |
| | vbs_hg | hg |
| | vbs_mg | mg |
| vgd | vgd_v | vgd voltages |
| | vgd_h | h |
| | vgd_m | m |
| | vgd_n | n |
| | vgd_hg | hg |
| | vgd_mg | mg |
| vg/vd | vg_vd_v | vg/vd voltage ratios |
| | vg_vd_h | h |
| | vg_vd_m | m |
| | vg_vd_n | n |
| | vg_vd_hg | hg |
| | vg_vd_mg | mg |

For example:


© 2003-2014 All Rights Reserved.

The following table model file:

```
.model nmos1
vbs h m n
-1.1 -1.1e8 00
-0.9 -9.1e7 0 0
-0.7 -8.2e7 0 0
-0.5 -2.3e7 0 0
-0.3 -4.35e6 0 0
0 0 0 0
.end
```

can also be written as:

```
*relxpert: + vbs_v = [ -1.1 -0.9 -0.7 -0.5 -0.3 0 ] // vbs bias
*relxpert: + vbs_h = [ -1.1e8 -9.1e7 -8.2e7 -2.3e7 -4.35e6 0 ] // h parameters
*relxpert: + vbs_m = [ 0 0 0 0 0 0] // m parameters
*relxpert: + vbs_n = [ 0 0 0 0 0 0] // n parameters
```

**Note:** If flexible table model parameters and a table file are provided at the same time, the Virtuoso RelXpert reliability simulator uses the flexible table model parameters.

### See Also

"NMOSFET DC Lifetime Model" on page 31 and "PMOSFET DC Lifetime Model" on page 31.


© 2003-2014 All Rights Reserved.

## .load_waveform_method

```
*relxpert: .load_waveform_method type=[single | partition partition_num=<value>]]
```

### Description

Specifies the method to load the waveform. The `type=partition` argument enables `relxpert_post` to split and load a large waveform and perform age simulation in parts.

### Arguments

| | |
|---|---|
| `single` | Loads the waveform in a single partition. |
| `partition` | (default). Splits and loads the waveform in parts. |
| `partition_num=<value>` | Specifies the partition numbers in which to split the waveform. If `partition_num` is not specified, `relxpert_post` automatically creates the partitions based on the waveform and system memory size. |

### Example

```
*relxpert: .load_waveform_method type=partition partition_num=5
```

Specifies that `relxpert_post` will split the waveform into five parts and perform age simulation.

**Note:** The `.load_waveform_method` command supports the waveform data only in Spectre format.


© 2003-2014
All Rights Reserved.

## .lpoly_method

```
*relxpert: .lpoly_method {spice | calc}
```

### Description

Sets the method for getting the model parameter lpoly.

### Arguments

| | |
|---|---|
| `spice` | Obtains the lpoly value from the SPICE output waveform (Default). |
| `calc` | Calculates the lpoly value internally. |

### Example

```
*relxpert: .lpoly_method spice
```

Specifies that the lpoly value will be obtained from the SPICE output waveform.


© 2003-2014 All Rights Reserved.

## .macrodevice

```
*relxpert: .macrodevice <subckt 1> [<subckt2>......]
```

### Description

Identifies the subcircuits in the netlist that are actually macro devices and therefore require special handling. You must specify at least one subcircuit.

Multiple `.macrodevice` statements can be included in a netlist, each one specifiying one or more subcircuits.

**Note:** A URI implementation of the aging equations must be properly set up for each macro device that is declared in the `.macrodevice` statement.

### Arguments

| | |
|---|---|
| `subckt` | Name of the subcircuit. |

### Example

```
*relxpert: .macrodevice mysubckt
```


© 2003-2014 All Rights Reserved.

## .maskdev

```
*relxpert: .maskdev type={include | exclude} { sub = [sub1 sub2 sub3 …] mod = [mod1
     mod2 mod3…] dev = [inst1 inst2 inst3 …] }
```

### Description

Includes or excludes:

- models that belong to subcircuits listed in the subckt list.
- devices that belong to models listed in the model list.
- devices that are listed in the instance list.

### Arguments

| | |
|---|---|
| `type=include` | Performs reliability simulation on the specified devices, or the models that belong to the listed subcircuit, or devices that belong to the listed model only. |
| `type=exclude` | Excludes the listed devices, or the models that belong to the listed subcircuit, or the devices that belong to the specified model during reliability simulation. |
| `sub` | Includes or excludes models that belong to the subcircuits listed in the `subckt` list. |
| `mod` | Includes or excludes devices that belong to the models listed in the `model` list. |
| `dev` | Specifies the instances to be included or excluded during reliability simulation. |

### Notes

- The `sub`, `mod`, and `dev` lists can be specified together.
- Multiple `.maskdev` statements can be used simultaneously.

### Examples

```
*relxpert: .maskdev type=include sub = [inv] mod=[nmos pmos] dev=[I1 I2 I3 I4]
```

Simulates all models that belong to the `inv` subcircuit and the `pmos` and `nmos` models. In


© 2003-2014 All Rights Reserved.

addition, it includes the `l1`, `l2`, `l3`, and `l4` devices.

```
*relxpert: .maskdev type=exclude sub = [inv] mod=[nmos]
```

Simulates all devices except devices that belong to the `inv` subcircuit and the `nmos` model.

```
*relxpert: .maskdev type=include dev=[I1 I2]
*relxpert: .maskdev type=include dev=[I3 I4]
```

Simulates the `I1`, `I2`, `I3`, and `I4` instances. This is equivalent to specifying `*relxpert: .maskdev type=include dev=[I1 I2 I3 I4]`.

```
*relxpert: .maskdev type=include dev=[I1 I2 I3 I4]
*relxpert: .maskdev type=exclude dev=[I3 I4]
```

Simulates the `I1` and `I2` instances. The above two statements are identical to the `*relxpert: type=include dev = [I1 I2]` statement.

**Note:** The syntax for `.maskdev` has been made consistent with Spectre (`maskdev`). However, the following old format is still supported:

```
*relxpert: maskdev { include | exclude} { subckt = [sub1 sub2 sub3 …] model = [mod1
     mod2 mod3 …] instance = [inst1 inst2 inst3 …] }
```


© 2003-2014 All Rights Reserved.

## .maxdeg

```
*relxpert: .maxdeg {value}
```

### Description

Sets the maximum degradation for reliability simulation. If the degradation of a device is higher than the maximum degradation value, the maximum degradation value is applied to the device.

### Arguments

| | |
|---|---|
| *value* | Sets the maximum degradation value. |

### Example

```
*relxpert: .maxdeg 0.55
```


© 2003-2014 All Rights Reserved.

## .minage

`*relxpert: .minage` *`value`*

### Description

Set the smallest *Age* value for which degraded SPICE model parameters are calculated. This statement speeds up aging calculation by using fresh SPICE model parameters if the transistor *Age* value is smaller than *`value`*.

### Arguments

| | |
|---|---|
| *`value`* | The smallest *Age* value that degraded SPICE model parameters calculates. *`value`* can be in decimal notation (xx.xx) or in engineering notation (x.xxe+xx). |

### Example

```
*relxpert: .minage 0.01
```

Generates a degraded SPICE model for transistors whose *Age* value is greater than `0.01`.


© 2003-2014 All Rights Reserved.

## .mindeg

`*relxpert: .mindeg {value}`

### Description

Sets the minimum degradation for reliability simulation. If the degradation of a device is lower than the minimum degradation value, minimum degradation value is set to `0` and a degraded model is not generated for the device.

### Arguments

| | |
|---|---|
| *value* | Sets the minimum degradation value. |

### Example

```
*relxpert: .mindeg 0.01
```


© 2003-2014 All Rights Reserved.

## .missing_signal_check

```
*relxpert: .missing_signal_check {type=error | ignore}
```

### Description

Specifies whether to generate an error or continue with the simulation, if a signal is missing in the input waveform.

### Arguments

| | |
|---|---|
| `error` | Generate an error and exit the simulation if a signal is missing in the input waveform. This is the default. |
| `ignore` | Generate an error but continue with the simulation if a signal is missing in the input waveform. |

### Example

```
*relxpert: .missing_signal_check type=ignore
```


© 2003-2014 All Rights Reserved.

## .nbtiageproc

```
*relxpert: .nbtiageproc
     mName
     files=fileName1 [ fileName2 … ]
```

### Description

Specifies aged SPICE model files (obtained experimentally) for generating NBTI degraded SPICE model using interpolation or regression method (selected through `.agemethod`).

### Arguments

| | |
|---|---|
| *mname* | The transistor model name for which the aged SPICE models are applicable. It must be the same model name used in SPICE `.model` card. Must contain only one `.model` card. Any *mname* without a corresponding `.nbtiageproc` statement will not be aged (that is, no degraded models will be generated). |
| *fileName1* | The model file containing the fresh model. *fileName* must be entered relative to the directory from which the Virtuoso RelXpert simulator is invoked unless the full pathname is specified. |
| *fileName2 …* | The model files containing the aged SPICE models. The orders of the aged SPICE model files should correspond with increasing age values. This means that *fileName1* is the fresh model file and *fileNamen* is the aged model file with the highest age value. *fileName* must be entered relative to the directory from which the Virtuoso RelXpert simulator is invoked unless the full pathname is specified. |

### Example

```
*relxpert: .nbtiageproc nmos files=model/nmos0.mod model/nmos1.mod model/nmos2.mod
```

### See Also

“.agemethod” on page 98 on how to set up aged model files.


© 2003-2014 All Rights Reserved.

## .nbti_only

```
*relxpert: .nbti_only <mos1> <mos2...>
```

### Description

If `modelname` is not specified, the Virtuoso RelXpert reliability simulator performs only negative bias temperature instability (NBTI) on all of the devices. If specified, the simulator calculates NBTI age values for devices included in the `.nbti_only` statement. For models not included in the statement, the simulator calculates both HCI and NBTI degradations (if HCI and NBTI models are available).

**Note:** The `.hci_only`, `.lifetime_method`, and `.nbti_only` options only work with the Cadence ageMOS model (agelevel=0, 1, 2).

### Arguments

| | |
|---|---|
| *modelname* | Specifies model name list (can only be a calculated NBTI age value) |

### Examples

The first example

```
*relxpert: .nbti_only
```

tells the Virtuoso RelXpert reliability simulator to calculate NBTI for all devices.

In the next example

```
*relxpert: .nbti_only pmos1
```

tells the simulator to calculate NBTI for `pmos1` model devices. For all other devices, both HCI and NBTI degradation is calculated.


© 2003-2014 All Rights Reserved.

## .opmethod

```
*relxpert: .opmethod {calc | spice}
```

### Description

Enables the `.vthmethod`, `.igatemethod`, and `.isubmethod` statements together.

### Arguments

| | |
|---|---|
| `calc` | Calculates the Vth, gate terminal, and substrate terminal current using the model parameters. This is the default value. |
| `spice` | Obtains the Vth, gate terminal, and substrate terminal current value from the SPICE output rawfile.<br><br>**Note:** The size of the SPICE output rawfile can increase considerably if you choose the `spice` argument. |

### Example

```
*relxpert: .opmethod spice
```

Specifies that the Vth, gate terminal, and substrate terminal current values should be obtained from the SPICE output rawfile.


© 2003-2014 All Rights Reserved.

## .output_append_deg

```
*relxpert: .output_append_deg spicepath=<path_to_the_spectre_simulator>
```

### Description

Enables RelXpert to output the degradation data in the append age flow (`uri_mode=appendage`). RelXpert outputs the degradation data in a text file with the name *basename*`_degdata.txt` after running `relxpert_pre -age` (`prebert2`).

### Arguments

| | |
|---|---|
| `spicepath` | Path to the Spectre simulator. |


© 2003-2014 All Rights Reserved.

## .output_device_degrad

```
*relxpert: .output_device_degrad
     vdd=vddValue1 [modelNameList...] {vdd=vddValue2 [modelNameList...]} ...
     {vdconst=vdconstvalue idconst=idconstvalue}
     [vgsat=vgsatvalue [modelNameList...]] [vglin=vglinvalue [modelNameList...]]
     [vdlin=vdlinvalue [modelNameList...]] [instParams=instparValue]
     [idconstwl=idconstwlValue] [vgstep=vgstepValue] [vsconst=vconstValue]
     [vbconst_n=vbconst_nValue] [vbconst_p=vbconst_pValue]
     [vgsweep_start_n=vgsweep_start_nValue]
     [vgsweep_start_p=vgsweep_start_pValue] [vgsweep_end_n=vgsweep_end_nValue]
     [vgsweep_end_p=vgsweep_end_pValue] [vth_use_id=vth_use_idVaue]
     [vth_use_is=vth_use_isValue] [vth_const_curr_sp=vth_const_curr_spValue]
     [ spice_path=path | -remote ]
```

### Description

Requests the Virtuoso RelXpert simulator output device degradation (gds, gm, Idlin, Idsat, Vth degradation) in relxpert_pre (or prebert) pass 2 for the specified age time period in your `.age` statement. The output filename is of file type `.bt#`, where `#` corresponds to the age (time) numbers.

For example, if you set three age values in your `.age` statement, relxpert_pre (or prebert) pass 2 outputs three degradation files: `.bt0`, `.bt1`, and `.bt2`.

The device degradation calculation is based on following fixed bias conditions.

**Note:** This option does not work when `.agemethod` is `agemos` and `uri_mode=appendage`. In addition, this option does not support macro devices.

#### *NMOSFET*

- Idsat: Vds=Vdd, Vgs=Vgsat if Vgsat is specified for the target model, otherwise Vgs=Vdd.
- Idlin: Vds=Vdlin if Vdlin is specified for the target model, otherwise Vds=0.05V; Vgs =Vglin if Vglin is specified for the target model, otherwise Vgs=Vdd.
- gm: Vds=Vdlin if Vdlin is specified for the target model, otherwise Vds=0.05V; Vgs =Vglin if Vglin is specified for the target model, otherwise Vgs=Vdd.
- Vth: Vds=Vdlin if Vdlin is specified for the target model, otherwise Vds=0.05V; Vgs =Vglin if Vglin is specified for the target model, otherwise Vgs=Vdd.


© 2003-2014 All Rights Reserved.

### *PMOSFET*

- Idsat: Vds=Vdd, Vgs=Vgsat if Vgsat is specified for the target model, otherwise Vgs=Vdd.
- Idlin: Vds=Vdlin if Vdlin is specified for the target model, otherwise Vds=-0.05V; Vgs =Vglin if Vglin is specified for the target model, otherwise Vgs=Vdd.
- gm: Vds=Vdlin if Vdlin is specified for the target model, otherwise Vds=-0.05V; Vgs =Vglin if Vglin is specified for the target model, otherwise Vgs=Vdd.
- Vth: Vds=Vdlin if Vdlin is specified for the target model, otherwise Vds=-0.05V; Vgs =Vglin if Vglin is specified for the target model, otherwise Vgs=Vdd.

You can also use `.output_device_degrad` to

- Set multiple vdd values for different models. If a model does not follow vdd, the vdd value should be used for models that have not been set.
- Change a measurement condition by setting the vgsat (default is vdd), vglin (default is vdd), or vdlin (default is 0.05 v) values. The order of the voltages is arbitrary.

**Note:** Wildcards (`*`) are supported and require quotation marks. For example,

```
.output_device_degrad vdd=3 "nmos*" -remote
```

### Arguments

| | |
|---|---|
| *vddValue* | The operation voltage of the circuit or target model. |
| *path* | The full path of your simulator. |
| *modelNameList* | The model name list. |
| *vgsatvalue* | Vgs value for Idsat measurement. |
| *vglinvalue* | Vgs value for Idlin/Vt/Gm measurement. |
| *vdlinvalue* | Vds value for Idlin/Vt/Gm measurement. |
| *vdconstvalue* | The constant drain voltage when calculating the threshold voltage using the `Vt_const_current` method. |
| *idconstvalue* | The constant drain current when calculating the threshold voltage using the `Vt_const_current` method.<br><br>**Note:** Set the `vdconstvalue` and `idconstvalue` arguments as a pair. |


© 2003-2014 All Rights Reserved.

| | |
|---|---|
| *instparValue* | The instance parameter type that is saved in the aged netlist file. The values for this argument are `WL` or `all` (default is `all`). |
| `-remote` | Calls a remote simulator. This statement only supports TI Spice3. |
| *idconstwlvalue* | Specifies that the constant drain current is related to transistor width(W) and length(L). idconstwl = idconst * (W / L); |
| *vgstepValue* | Specifies the gate voltage (Vg) sweep step size while calculating the device degradation (Vth, ids, idling, gds, and gm degradation). |
| *vsconstValue* | Specifies the constant source terminal voltage when calculating the threshold voltage degradation. |
| *vbconst_nValue* | Specifies the NMOSFET's constant substrate terminal voltage when calculating the threshold voltage degradation. |
| *vbconst_pValue* | Specifies the PMOSFET's constant substrate terminal voltage when calculating the threshold voltage degradation. |
| *vgsweep_start_nValue* | Specifies the start point voltage for the sweep voltage of the NMOSFET's gate terminal voltage when calculating the threshold voltage degradation. |
| *vgsweep_start_pValue* | Specifies the start point voltage for the sweep voltage of the PMOSFET's gate terminal voltage when calculating the threshold voltage degradation. |
| *vgsweep_end_nValue* | Specifies the stop point voltage for the sweep voltage of the NMOSFET's gate terminal voltage when calculating the threshold voltage degradation. |
| *vgsweep_end_pValue* | Specifies the stop point voltage for the sweep voltage of the PMOSFET's gate terminal voltage when calculating the threshold voltage degradation. |
| *vth_use_idVaue* | Specifies that drain current will be used to calculate the degraded threshold voltage with `vt_const_current` method. |
| *vth_use_isValue* | Specifies that source current will be used to calculate the degraded threshold voltage with `vt_const_current` method. |


© 2003-2014 All Rights Reserved.

*vth_const_curr_spValue*

Specifies that the `vt_const_current` method will be used to calculate the threshold degradation value.

### Examples

```
*relxpert: .output_device_degrad vdd=3.3 spice_path=/bin/tispice
*relxpert: .output_device_degrad vdd=3.3 -remote
```

In the following example

```
*relxpert: .output_device_degrad vdd=3.3 nmos nmos1 vdd=3 pmos vdd=3 -remote
```

tells the Virtuoso RelXpert reliability simulator the `nmos` and `nmos1` models are set to 3.3 v for vdd and the `pmos` model is set to 3 v for vdd (for all other models, the vdd value is 3 v).

The next example

```
*relxpert: .output_device_degrad vdd=3 nmos *2 vdd=1.5
spice_path=/bin/tispice
```

tells the simulator the vdd value for the `nmos`, `nmos2`, and `pmos2` models is set to 3 v, and the remaining models to 1.5 v.

The next example

```
*relxpert: .output_device_degrad vdd=3 nmos *2 vgsat=3.3 nmos vdd=1.5 -remote
```

tells the simulator the vdd value for the `nmos`, `nmos2`, and `pmos2` models is set to 3 v, and for the remaining models set to 1.5 v. For vgsat, the value for the `nmos` model is 3.3 v, `nmos2` and `pmos2` models is 3 v (default), and the remaining models is 1.5 v (default).


© 2003-2014 All Rights Reserved.

## .output_inst_param

```
*relxpert: .output_inst_param
     [list=[param1 param2...]]
```

### Description

Adds the instance state parameters to the `bo0` table in the new table format. The instance state parameters are obtained from the URI shared library by using the `getInstParam()` function.

**Note:** `.output_inst_param` takes higher precedence over .output_table_format. Therefore, `.output_inst_param` generates the data in the new table format even if `.output_table_format 0` is specified.

### Arguments

| | |
|---|---|
| `list` | Name of the instance state parameters that should be added to the `bo0` table.<br>**Note:** The instance state parameters must be defined in the `getInstParam()` function, otherwise, RelXpert generates a warning. |

### Example

```
*relxpert: .output_inst_param list=[dtemp leff]
```

The above statement will obtain the instance state parameters `dtemp` and `leff` from the URI shared library using the `getInstParam()` function, and add them to the `bo0` table.


© 2003-2014 All Rights Reserved.

## .output_method

```
*relxpert: .output_method
     {single|integ}
```

### Description

Specifies the degradation calculation method that will be used to calculate the degradation results for a MOSFET. The degradation results are generated in the bt# file.

RelXpert supports three reliability models, namely HCI, NBTI, and PBTI. Therefore, the bt# file contains device degradation results based on these three models. The HCI and PBTI reliability models are used for calculating degradation results for NMOSFET, and the HCI and NBTI reliability models are used for calculating degradation results for PMOSFET.

### Arguments

| | |
|---|---|
| `single` | Uses the aged model card to calculate the degradation results. The results are generated in the bt# file.<br><br>This is the default value. |
| `integ` | Calculates the HCI, NBTI, and PBTI reliability model's degradation results separately.<br><br>The HCI and NBTI degradation results are added to calculate the degradation results for PMOSFET, and the HCI and PBTI degradation results are added to calculate the degradation results for NMOSFET. The results are generated in the bt# file. |

### Examples

```
*relxpert: .output_method integ
```


© 2003-2014
All Rights Reserved.

## .output_table_format

```
*relxpert: .output_table_format {0|1} [file=filename]
```

### Description

This statement determines whether to use the old or new table format. In addition, it allows you to specify a different name for the output `bo0` file.

### Arguments

| | |
|---|---|
| `0|1` | Determines whether to use the new and improved table format or the old table format. When set to `1`, RelXpert generates the output in an improved table format. By default, the output is generated in the old table format. |
| `file=`*`filename`* | Enables you to specify a `bo0` file name of your choice.<br>Default: *`basename`*`.bo0` (same as old format) |

### Example

```
*relxpert: .output_table_format 1
```

Generates the output in the improved table format as shown:

```
elem name    total age    degradation    lifetime    agelevels    model
---------    ---------    ----------     --------    --------    -------
I3.M1        0.0181418    0.581095        197195       0,1        nfet
I12.M1       0.0196955    0.575193        181806       0,1        pfet
```


© 2003-2014 All Rights Reserved.

## .plot_agerate

```
*relxpert: .plot_agerate {devices list | all}
```

### Description

Prints waveform for agerate at each time step.

### Arguments

| | |
|---|---|
| *list* | Specifies the list of devices for which the agerate needs to be printed in the waveform at each time step. |
| all | Specifies that the agerate needs to be printed in the waveform for all devices at each time step. |

### Example

```
*relxpert: .plot_agerate I0.M0 I0.M1 I1.M0
```

Prints the agerate at each time step in the waveform for devices `I0.M0`, `I0.M1`, and `I1.M0`.

```
*relxpert: .plot_agerate all
```

Prints the agerate at each time step in the waveform for all the devices.


© 2003-2014
All Rights Reserved.

## .plot_isub

```
*relxpert: .plot_isub {devices list | all}
```

### Description

Prints waveform for substrate terminal current (Isub) of specified devices at each time step.

### Arguments

| | |
|---|---|
| *list* | Specifies the list of devices for which the substrate terminal current (Isub) needs to be printed in the waveform at each time step. |
| `all` | Specifies that the substrate terminal current (Isub) needs to be printed in the waveform for all devices at each time step. |

### Example

```
*relxpert: .plot_isub I0.M0 I0.M1 I1.M0
```

Prints Isub at each time step in the waveform for devices `I0.M0`, `I0.M1`, and `I1.M0`.

```
*relxpert: .plot_isub all
```

Prints Isub at each time step in the waveform for all the devices.


© 2003-2014 All Rights Reserved.

## .plot_igate

```
*relxpert: .plot_igate {devices list | all}
```

### Description

Prints waveform for transient gate current (Igate) at each time step.

### Arguments

| | |
|---|---|
| *list* | Specifies the list of devices for which the transient gate current (Igate) needs to be printed in the waveform at each time step. |
| `all` | Specifies that the transient gate current (Igate) needs to be printed in the waveform for all devices at each time step. |

### Example

```
*relxpert: .plot_igate I0.M0 I0.M1 I1.M0
```

Prints Igate at each time step in the waveform for devices `I0.M0`, `I0.M1`, and `I1.M0`.

```
*relxpert: .plot_igate all
```

Prints Igate at each time step in the waveform for all the devices.


© 2003-2014 All Rights Reserved.

## .plot_urimosvar

```
*relxpert: .plot_urimosvar(.ploturimosvar) all | device_list
```

### Description

This statement generates the `uri_MosVar` values in a PSF file (default) and saves it in the `relxpert_plot.raw` directory. The PSF file can then be viewed in ViVA waveform viewer.

Alternatively, you can use the `PLOT_WAVEFORM_FORMAT` environment variable to generate the `uri_MosVar` values in an SST2 file as follows:

```
setenv PLOT_WAVEFORM_FORMAT sst2
```

### Arguments

| | |
|---|---|
| `all` | Generates the `uri_MosVar` values for all devices. |
| *`device_list`* | Specifies the list of devices for which the `uri_MosVar` values should be generated in PSF or SST2 format. |

### Example

```
*relxpert: .plot_urimosvar all
```

Prints the `uri_MosVar` values for all devices in a PSF file. The following values in `uri_MosVar` struct will be plotted:

```
typedef struct uri_MosVar
{
   ...
        double vgs;     /* Vgs */
        double vds;     /* Vds */
        double vbs;     /* Vbs */
        double von;     /* thresold voltage */
        double vdsat;   /* saturation voltage */
        double vfbeff;  /* effective Vfb */
        double phi;     /* surface potential */
        double ids;     /* channel drain current */
        double isub;    /* substrate current */
        ..
        double leff;    /* bias dependent */
```


© 2003-2014 All Rights Reserved.

```
        double weff;     /* bias dependent */
        double tox;      /* bias dependent */
        ...
        double igate;    /* gate current*/
        double vs;       /*voltage of source area*/
        double lpoly;    /* lpoly value. Only BSIM4 and PSP model support this
    paramter.*/
} URI_MosVar;
```


© 2003-2014 All Rights Reserved.

# .preset

```
*relxpert: .preset age|deg|lifetime=double [agelevel=integer] device|model|block
```

### Description

This statement is used to preset the total `age`, `deg` (degradation), or `lifetime` values of devices in the netlist file. If `.preset` is set, the Virtuoso RelXpert reliability simulator uses the values directly, but does not calculate them. The `.preset` statement is useful if you already know the `age`|`degradation`|`lifetime` of blocks or devices, and want to preset the values in order to speed up the calculation.

If the preset is `age`, the simulator does not perform an age calculation for the specified devices, models, or blocks. Lifetime or degradation calculations are based on the preset `age` value. If the preset is `lifetime` or `deg`, the simulator derives age from the preset lifetime or degradation values to calculate degradation or lifetime for the specified devices, models, or blocks.

**Note:** Wildcards (`*`) are supported.

### Arguments

| | |
|---|---|
| `age` | The age value of the device, model, or block. |
| `deg` | The degradation value of the device, model, or block. |
| `lifetime` | The lifetime value of the device, model, or block. |
| `agelevel` | Defined for Hot-Carrier injection (HCI), negative bias temperature instability (NBTI), or positive bias temperature instability (PBTI) modeling. The following agelevel values are assigned for different modeling types:<br>■ HCI: `agelevel=0`<br>■ NBTI: `agelevel=1`<br>■ PBTI: `agelevel=2`<br>The default behavior is to calculate both HCI and NBTI, or both HCI and PBTI. |


© 2003-2014
All Rights Reserved.

### Examples

In the following example

```
*relxpert: .preset age=1.0001e-5 agelevel=1 x1.mp1
```

tells the Virtuoso RelXpert reliability simulator to set the device `x1.mp1` NBTI age value to 1.001e-5.

In the next example

```
*relxpert: .preset deg=1.01e-02 PMOS
```

tells the simulator to set the degradation value to 1.01e-02 for all devices which use the `PMOS` model.

In the next example

```
*relxpert: .preset lifetime=10y agelevel=0 X1 X2*mp1
```

tells the simulator to set the `lifetime` value to 10 years for all devices in the `X1` block or for devices with hierarchical names that begin with the `X2` prefix and end with the `mp1` suffix (for example, `X2.mp1`, `X21.mp1`, and `X2.X22.mp1`).


© 2003-2014 All Rights Reserved.

## .print_agerate

```
*relxpert: .print_agerate {devices list | all}
```

### Description

Prints transient agerate for the listed devices (transistors) or all the devices. agerate is the degradation rate as a function of device degradation mode. Under DC stress:

```
agerate * time_period = Age
```

The `.print_agerate` statement allows you to print NMOSFET and PMOSFET agerate.

### Arguments

| | |
|---|---|
| *list* | Specifies the list of devices for which the transient agerate needs to be printed in the waveform at each time step. |
| `all` | Specifies that the transient agerate needs to be printed in the waveform for all devices at each time step. |

### Examples

```
*relxpert: .print_agerate I0.M0 I0.M1 I1.M0
```

Prints the agerate at each time step in bo0 file for devices `I0.M0`, `I0.M1`, and `I1.M0`.

```
*relxpert: .print_agerate all
```

Prints the agerate at each time step for all the devices.


© 2003-2014 All Rights Reserved.

## .printigate

```
*relxpert: .printigate
     { all | trnNames }
```

### Description

Requests printout of transient gate current for all or the listed transistors.

### Arguments

| | |
|---|---|
| *trnNames* | The transistors whose transient gate current is to be printed. |

### Example

```
*relxpert: .printigate xxdecoder.mp1 xydecoder.mp2 mpout
```

Prints the transient gate current for the transistors `xxdecoder.mp1`, `xydecoder.mp2`, and `mpout`.

### See Also

"MOSFET Hot-Carrier Simulation" on page 233 on the printed output.

"relxpert_post (or postbert)" on page 218 for using the `-m` option in the relxpert_post (or postbert) command line and for the plot file option.


© 2003-2014 All Rights Reserved.

## .printigatetotal

```
*relxpert: .printigatetotal
```

### Description

Requests a printout of the total transient gate current in the circuit. Only PMOSFET gate current is printed.

### Arguments

None.

### See Also

"MOSFET Hot-Carrier Simulation" on page 233.

"Batch Mode" on page 214 for using the `-m` option in the relxpert_post (or postbert) command line.


© 2003-2014 All Rights Reserved.

# .printisub

```
*relxpert: .printisub
     { all | trnNames }
```

### Description

Requests a printout of transient substrate current for all or the listed transistors. NMOSFET and PMOSFET substrate currents can be printed.

### Arguments

| | |
|---|---|
| *trnNames* | The transistors whose transient substrate current is to be printed. |

### Example

```
*relxpert: .printisub xxdecoder.mn1 xydecoder.mp2
```

Prints transient substrate current for the transistors `xxdecoder.mn1` and `xydecoder.mp2`.

### See Also

"MOSFET Hot-Carrier Simulation" on page 233.

"Batch Mode" on page 214 for using the `-m` option in the relxpert_post (or postbert) command line.


© 2003-2014
All Rights Reserved.

## .printisubtotal

```
*relxpert: .printisubtotal
```

### Description

Requests a printout of the total transient substrate current in the circuit. NMOSFET and PMOSFET substrate currents are printed separately.

### Arguments

None.

### See Also

"MOSFET Hot-Carrier Simulation" on page 233 on the printed output.

"Batch Mode" on page 214 for using the `-m` option in the relxpert_post (or postbert) command line.


© 2003-2014 All Rights Reserved.

## .printisub_idsmax

```
*relxpert: .printisub_idsmax
```

### Description

This command requests a printout of $\left(\frac{I_{sub}}{I_{ds}}\right)_{max}$ values for all or listed transistors.

### Arguments

None.

**Note:** If `.printisub_idsmax` is specified, average gate current (ageIg) will not be printed; it will be replaced by $\left(\frac{I_{sub}}{I_{ds}}\right)_{max}$


© 2003-2014
All Rights Reserved.

## .relx_tran

`*relxpert: .relx_tran` *`start_time`* `[`*`stop_time`*`]`

### Description

Specifies the start and stop time (in nanoseconds) for reliability simulation during transient simulation. The start and stop time defines the transient simulation range. If this option is not specified, Relxpert transient simulation range is decided by the SPICE transient simulation option or the Spectre transient simulation option.

### Arguments

| | |
|---|---|
| *start_time* | Specifies the start time of reliability analysis during transient simulation. |
| *stop_time* | Specifies the stop time of reliability analysis during transient simulation.<br><br>Default: If *stop_time* is not specified, the stop time of transient analysis in the netlist is available. |

### Example

```
*relxpert: .relx_tran 1n 10n.
```

Performs reliability analysis from 1n to 10n during transient simulation.


© 2003-2014 All Rights Reserved.

## .report_model_param_changes

```
*relxpert: .report_model_param_changes
```

### Description

This option is used to track the aged parameters. If this option is used in the netlist, the fresh and aged parameters are outputted to the .bm# file.

### Arguments

None


© 2003-2014
All Rights Reserved.

## .show_unaged_devices

```
*relxpert: .show_unaged_devices value=[all|none|limit]
```

### Description

This option is used to output all unaged devices.

### Arguments

| | |
|---|---|
| `all` | Output all unaged devices |
| `none` | Do not output unaged devices |
| `limit` | Output only 20 unaged devices. This is the default. |


© 2003-2014 All Rights Reserved.

## .uri_lib

```
*relxpert: .uri_lib file = { "uri_lib_name" } uri_mode= [agemos | appendage] debug
     = [ 0 | 1 ]
```

### Description

Loads the URI library for reliability calculation.

For more information on using URI, see *Virtuoso Unified Reliability Interface Reference*.

### Arguments

| | |
|---|---|
| *file=uri_lib*_name | URI library filename. |
| *uri_mode* | Age calculation method used for uri_mode (value can either be `appendage` or the default value `agemos`).<br><br>**Note:** If both `uri_lb` with `uri_mode` = `appendage` and `.output_device_degrad` control statements are specified, the `.output_device_degrad` control statement is ignored. |
| *debug* | Specifies the debug mode for URI library. The value can be `0` or `1`. When this argument is specified, a flag is added to the URI library indicating whether the debug information should be printed. `debugMode=1` prints debug messages.<br><br>Default value is 0. |

### Example

```
*relxpert: .uri_lib file= “./Uri_lib.so” uri_mode=appendage debug=1
```

**Note:** The syntax for `.uri_lib` has been made consistent with Spectre. However, the following old format is still supported:

```
*relxpert: .uri_lib uri_lib.so [uri_mode=uri_modeValue debug=debugMode]
```


© 2003-2014 All Rights Reserved.

## .vmax

`*relxpert: .vmax {`*`value`*`} {model = [`*`model_list`*`]}`

### Description

Sets the maximum voltage for reliability simulation. If the voltage of `Vds`, `Vgs` or `Vbs` is higher than the maximum voltage, the specified maximum voltage value is applied.

### Arguments

| | |
|---|---|
| *value* | Sets the maximum voltage value. |
| *model_list* | Sets the list of models for which the maximum voltage value will be applied. |

### Examples

```
*relxpert: .vmax 2 model = [nmos_0 nmos_1]
```


© 2003-2014 All Rights Reserved.

## .vthmethod

```
*relxpert: .vthmethod
     { calc | spice }
```

### Description

Specifies the method used for obtaining vth values.

In MOSFET HCI simulation, substrate and gate current models require vth values (see "MOSFET Substrate and Gate Current Models" on page 23). The Virtuoso RelXpert simulator can either calculate this value or obtain it from the SPICE output rawfile if the SPICE simulator provides such an option.

If this command is not specified, the Virtuoso RelXpert simulator calculates vth values from model parameters.

### Arguments

| | |
|---|---|
| `calc` | Calculates vth values from the model parameters. This is the default. |
| `spice` | Obtains vth values from the SPICE rawfile. The size of the SPICE output rawfile will be increased to include all vth values. |

### See Also

"MOSFET Substrate and Gate Current Models" on page 23.

"Extracting MOSFET Substrate and Gate Current Parameters" on page 47.


© 2003-2014 All Rights Reserved.

# 8

# Virtuoso Spectre Interface

This chapter describes the Virtuoso® Spectre® direct interface in the Virtuoso RelXpert reliability simulator: Input statements with the Virtuoso Spectre input files that are recognized by the Virtuoso RelXpert simulator. Details of Virtuoso Spectre command statements can be found in the *Virtuoso Spectre Circuit Simulator User Guide*.

For Virtuoso RelXpert simulator command options compatible with Virtuoso Spectre, refer to "Batch Mode" on page 214.

**Table 8-1 Virtuoso Spectre Input Control Statements**

| |
|---|
| define |
| global |
| include |
| options |

**Table 8-2 Virtuoso Spectre Simulation Control Statements**

| |
|---|
| tran |
| dc |

**Table 8-3 Virtuoso Spectre Model and Element Statements**

| |
|---|
| Iyyy |
| model |
| subckt |


© 2003-2014 All Rights Reserved.

## dc

Use dc analysis to calculate substrate and gate currents from a HCI simulation.

For example

```
dc dc dev=V0 start=0 stop=5 step=1
```

**Note:** The Virtuoso RelXpert reliability simulator currently only supports simple dc sweeps.


© 2003-2014 All Rights Reserved.

## define

Set the values of parameters.

The *save* option in *saveOptions* must be *save=all* or *save=allpub* to save all voltage results of the circuit.


© 2003-2014 All Rights Reserved.

## global

Set the common node name referenced by subcircuits and the top-level circuit.


© 2003-2014 All Rights Reserved.

## include

The name of the file in the `include` statement is read and included in the input netlist file.


© 2003-2014 All Rights Reserved.

## Iyyy

Subcircuit call is read and expanded in the input netlist file by the Virtuoso RelXpert simulator.


© 2003-2014 All Rights Reserved.

## model

The Virtuoso RelXpert simulator reads model parameters for MOSFET devices. Parameters relevant to reliability simulation are also set in the `.model` card (see Chapter 6, “Setting Up Models for Simulation”).

The Virtuoso Spectre MOSFET models supported by the Virtuoso RelXpert simulator for Hot-Carrier injection (HCI), negative bias temperature instability (NBTI), and positive bias temperature instability (PBTI) simulations are:

- UC Berkeley series models

  BSIM3, BSIM3V3, BSIM4, BSIMSOI, and BSIMCMG

- Philips MOSFET models

  MOS9, MOS11, and PSP102.0

- TI Spice3 MOSFET models

  BSIM1, BSIM3, BSIM3v3, BSIM4, and BSIMSOI

- HVMOS (Spectre syntax only, does not support SPICE model level=66 syntax), HISIM_HV, and HISIM2 models.


© 2003-2014 All Rights Reserved.

## options

The following options in the control statement are used by the Virtuoso RelXpert simulator:

- `rawfmt`

  Set `rawfmt=psfbin` or `rawfmt=psfascii` in *simulatorOptions* to output waveform results in PSF format.

- save

  Set `save=all` or `save=allpub` in *saveOptions* to output all voltage waveforms in the circuit.

  **Note:** The two options above must be set in order to generate the output file for the Virtuoso RelXpert simulator.

- `scale=x`

  Set the scale factor in *simulatorOptions* for transistor dimensions to `x`.

- `scalem=x`

  Set the scale factor in *simulatorOptions* for transistor model parameters to `x`.

- `tnom=x`

  Set the nominal temperature in *simulatorOptions* SPICE model parameters are referenced.

- `temp=x`

  Set the temperature for Virtuoso RelXpert simulation in *simulatorOptions*.


© 2003-2014 All Rights Reserved.

## subckt

Subcircuit definition is read by the Virtuoso RelXpert simulator.


© 2003-2014
All Rights Reserved.

## tran

Transient analysis is used for Hot-Carrier, NBTI, and PBTI simulations. Set the time step and time interval for Virtuoso RelXpert simulation.

**Note:** This statement must be set for Virtuoso RelXpert simulator Hot-Carrier degradation, NBTI, PBTI, and circuit aging simulations.


© 2003-2014 All Rights Reserved.

## SPICE Circuit Elements

The following table lists the circuit elements read and analyzed by the Virtuoso RelXpert simulator:

**Table 8-4 Default Parameters for Substrate Current**

| Reliability Simulation | SPICE Element |
|---|---|
| MOSFET Hot-Carrier, NBTI, and PBTI | myy |


© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 9

# TI Spice3 Interface

This chapter describes the TI Spice3 interface in the Virtuoso® RelXpert reliability simulator: Input statements with the TI Spice3 input files that are recognized by the Virtuoso RelXpert simulator. Details of TI Spice3 command statements can be found in the TI Spice3 manual.

For Virtuoso RelXpert simulator command options compatible with TI Spice3, refer to "Batch Mode" on page 214.

**Table 9-1 TI Spice3 Input Control Statements**

| |
|---|
| .global |
| .include |
| .lib |
| .option |
| .param |

**Table 9-2 TI Spice3 Simulation Control Statements**

| |
|---|
| .dc |
| .mselect |
| .punch |
| .scale |
| .temp |
| .tran |

**Table 9-3 TI Spice3 Model and Element Statements**

| |
|---|
| .macro |


© 2003-2014
All Rights Reserved.

**Table 9-3 TI Spice3 Model and Element Statements**

| |
|---|
| .model |
| .subckt |
| xyyy |


© 2003-2014 All Rights Reserved.

## .dc

DC analysis is used to calculate substrate and gate currents from Hot-Carrier simulation.


© 2003-2014
All Rights Reserved.

## .global

Set the common node name referenced by subcircuits and the top-level circuit.


© 2003-2014 All Rights Reserved.

## .include

The name of the file in the `.include` statement is read and included in the input netlist file.


© 2003-2014 All Rights Reserved.

## .lib

The `.lib` statement is similar in function to the `.include` statement.


© 2003-2014 All Rights Reserved.

## .macro

Subcircuit definition is read by the Virtuoso RelXpert simulator.


© 2003-2014
All Rights Reserved.

## .model

The Virtuoso RelXpert simulator reads model parameters for MOSFET devices. Parameters relevant to reliability simulation are set in the `.model` card (see Chapter 6, "Setting Up Models for Simulation,").


© 2003-2014
All Rights Reserved.

## .mselect

The Virtuoso RelXpert simulator only supports model selection for MOSFET. Each `.mselect` statement can contain multiple models names.


© 2003-2014 All Rights Reserved.

## .options

The following options in the control statement are used by the Virtuoso RelXpert simulator:

- `tnom=x`

  Set the nominal temperature SPICE model parameters are referenced.

- `defl=x`

  Set the default MOSFET channel length for Berkeley models.

- `defw=x`

  Set the default MOSFET channel width for Berkeley models.


© 2003-2014 All Rights Reserved.

## .param

The values of the parameters are set.

In TI Spice3, p*=X has the same meaning as the `.param` statement.


© 2003-2014 All Rights Reserved.

## .punch

This statement is used to output voltage values of all of the nodes and drain current values for Virtuoso RelXpert simulation.


© 2003-2014 All Rights Reserved.

## .scale

The Virtuoso RelXpert simulator only supports scale for MOSFET devices. In addition, only the `W` and `L` parameters can be scaled.


© 2003-2014 All Rights Reserved.

## .subckt

Subcircuit definition is read by the Virtuoso RelXpert simulator.


© 2003-2014
All Rights Reserved.

## .temp

Set the temperature for Virtuoso RelXpert simulation.

**Note:** If multiple temperatures are specified, the Virtuoso RelXpert simulator uses the first temperature value.


© 2003-2014
All Rights Reserved.

## .tran

Transient analysis is used for Hot-Carrier, NBTI, and PBTI simulations. Set the time step and time interval for Virtuoso RelXpert simulation.

**Note:** This statement must be set for Virtuoso RelXpert simulator Hot-Carrier degradation, NBTI, PBTI, and circuit aging simulations.


© 2003-2014
All Rights Reserved.

## xyyy

Subcircuit call is read and expanded in the input netlist file by the Virtuoso RelXpert simulator.


© 2003-2014
All Rights Reserved.

# SPICE Circuit Elements

The following table lists the circuit elements read and analyzed by the Virtuoso RelXpert simulator:

**Table 9-4 SPICE Circuit Elements Used by the Virtuoso RelXpert Simulator**

| Reliability Simulation | SPICE Element |
|---|---|
| MOSFET Hot-Carrier, NBTI, and PBTI | myy |


© 2003-2014 All Rights Reserved.

# 10

# TITAN Interface

This chapter describes the TITAN interface in the Virtuoso® RelXpert reliability simulator.

## Running the Virtuoso Reliability Simulator

The Virtuoso RelXpert reliability simulator consists of pre- and post-processors linked by a SPICE simulator (that is, `titan_ip` and `titan`). The reliability simulator can be run in batch or interactive mode.

### Interactive Mode

To run the Virtuoso RelXpert reliability simulator in interactive mode:

1. Run relxpert_pre (or prebert).
2. Run the SPICE simulation using TITAN.

   Running a SPICE simulation with TITAN is a two step process using the `titan_ip` and `titan` commands.

   A more common approach is to use a wrapper to run the SPICE simulation. Cadence has released the `cds_titan_run` wrapper, which can be modified with the `$RELXPERT_TITAN` environment variable and used with the Virtuoso RelXpert reliability simulator.

   Use one of the following statements to run the TITAN wrapper,

   ```
   cds_titan_run circuit_name
   ```

   or

   ```
   $RELXPERT_TITAN circuit_name
   ```

3. Run relxpert_post (or postbert).

   **Note:** For a degraded netlist file simulation (aging), run relxpert_pre (or prebert) and SPICE again in a second simulation.


© 2003-2014 All Rights Reserved.

### Example

If the `cmos.inc` netlist file is in TITAN format and the filename in the .save statement is `cmos`, the simulation flow is as follows:

1. `relxpert_pre -si -p1 cmos.inc cmos.p1`
2. `cds_titan_run cmos.p1`

   (or `$RELXPERT_TITAN cmos.p1`)
3. `relxpert_post -r cmos/psf/cmos.tr.psf cmos1.out`
4. `relxpert_pre -si -p2 cmos.ba0 cmos.inc cmos.p2`
5. `cds_titan_run cmos.p2`

   (or `$RELXPERT_TITAN cmos.p2`)

## Batch Mode

To run the Virtuoso RelXpert reliability simulator in batch mode, use the `RUN_RELXPERT` script provided by Cadence. For more information about the script, refer to "RUN_RELXPERT" on page 228.

### Example

If the `cmos.inc` netlist file is in TITAN format and the filename in the .save statement is `cmos`, use

```
% RUN_RELXPERT -si cmos.inc cmos
```


© 2003-2014 All Rights Reserved.

# TITAN Statements

The TITAN statements described in this section are supported by the Virtuoso RelXpert reliability simulator.

**Table 10-1 TITAN Input Control Statements**

| |
|---|
| .include |
| .incpath |
| .options |
| .param |

**Table 10-2 TITAN Simulation Control Statements**

| |
|---|
| .comopt |
| .connect |
| .geo |
| .save |
| .temp |
| .tran |

**Table 10-3 TITAN Model and Element Statements**

| |
|---|
| .model |
| .subckt |

For Virtuoso RelXpert reliability simulator command options compatible with TITAN, refer to "Batch Mode" on page 214.

### MOSFET Node Names

The Virtuoso RelXpert reliability simulator supports MOSFET node names designated by ND, NS, NB, and NG.


© 2003-2014 All Rights Reserved.

### *Example*

```
mP888 ND=WLDV NG=WLDINT NS=vpp! NB=vpp! mpfltw w=' 3.500' l='240.000m'
```

The TITAN statements that are not supported by the Virtuoso RelXpert reliability simulator include

- `.alter`
- `.modify`
- `.trlocal`
- FORTRAN-like exponent D (1D-10 and the exponentiation operator `**` )
- Lines starting with an asterisk (`*`) immediately followed by a vertical bar (`|`)

  For example, `*|`.
- Special characters in device names, such as `<` , `>`, `:`, `$`.

  For example, `M52<, Mout>, Minput$`.

**Note:** The list above may not contain all of the TITAN statements that are not supported by the reliability simulator.


© 2003-2014 All Rights Reserved.

## .comopt

| Options | Descriptions |
|---|---|
| `.comopt names` | Specify the hierarchical names |
| `.comopt ground` | Define additional nodes, other than 0, to be treated as ground nodes |


© 2003-2014 All Rights Reserved.

## .connect

Connect two nodes so that they are identified electrically.


© 2003-2014
All Rights Reserved.

## .geo

The geometrical dimensions of the MOSFET are modified by the model parameters.


© 2003-2014 All Rights Reserved.

## .include

The name of the file in the `.include` statement is read and included in the input netlist file.


© 2003-2014 All Rights Reserved.

## .incpath

The name of the file in the `.include` statement is read in this directory. The absolute or relative path is supported.


© 2003-2014
All Rights Reserved.

## .model

The Virtuoso RelXpert reliability simulator reads model parameters for MOSFET devices. Parameters relevant to reliability simulation are also set in the `.model` card (see Chapter 6, “Setting Up Models for Simulation”).

The TITAN MOSFET models supported by the reliability simulator are BSIM3v3 and BSIM4.


© 2003-2014 All Rights Reserved.

## .options

The following options in the `.options` control statement are used by the Virtuoso RelXpert reliability simulator:

- `scale=x`

  Set the scale factor for transistor dimensions to `x`.

- `scalm=x`

  Set the scale factor for transistor model parameters to `x`.

- `tnom=x`

  Set the nominal temperature for SPICE model parameters to `x`.


© 2003-2014 All Rights Reserved.

## .param

The values of parameters are set.


© 2003-2014
All Rights Reserved.

## .save

```
.save xxx FORMAT=psf
```

Save transient analysis results.

**Note:** Only parameter storage format (PSF) output is supported by the Virtuoso RelXpert reliability simulator.


© 2003-2014 All Rights Reserved.

## .subckt

Subcircuit definition is read by the Virtuoso RelXpert reliability simulator.


© 2003-2014
All Rights Reserved.

## .temp

Set the temperature for Virtuoso RelXpert reliability simulation.

**Note:** If multiple temperatures are specified, the reliability simulator uses the first temperature value (by default).


© 2003-2014 All Rights Reserved.

## .tran

Transient analysis is used for Hot-Carrier injection and NBTI simulation. Set the time step and interval for the Virtuoso RelXpert reliability simulation.

**Note:** This statement must be set for Virtuoso RelXpert simulator Hot-Carrier degradation, NBTI, and circuit aging simulations.


© 2003-2014 All Rights Reserved.

# SPICE Circuit Elements

The following table lists the circuit elements read and analyzed by the Virtuoso RelXpert reliability simulator:

**Table 10-4 Default Parameters for Substrate Current**

| Reliability Simulation | SPICE Element |
|---|---|
| MOSFET Hot-Carrier | myy |
| MOSFET NBTI | myy |
| MOSFET PBTI | myy |


© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 11

# Running the Simulator

The Virtuoso® RelXpert reliability simulator can be run in interactive or batch mode using the Virtuoso RelXpert simulator/analog design environment (ADE) interface. Interactive mode lets you change the Virtuoso RelXpert simulator command lines, run the simulation, and view the results interactively. Batch mode runs simulations in the background.

**Note:** Starting in the 7.1.1 release, the binary name prebert was replaced with relxpert_pre, postbert was replaced with relxpert_post, and prebert2 was replaced with relxpert_pre -age. However, to maintain backward compatibility, the RelXpert software will continue to support the prebert, postbert, and prebert2 binary names.

## Overview of Running the Reliability Simulator

The Virtuoso RelXpert simulator consists of pre- and postprocessors linked by the SPICE simulator.

The preprocessor, relxpert_pre (or prebert),

1. Filters Virtuoso RelXpert simulator command lines and stores the information relevant to reliability simulation in a temporary file to be read by the postprocessor
2. Stores device elements in a temporary file to be read by the postprocessor
3. In second-pass Hot-Carrier simulations, creates degraded transistor models for circuit aging simulation

The postprocessor, relxpert_post (postbert),

1. Reads model parameters, device information, and simulation options from files stored by the preprocessor, relxpert_pre (or prebert)
2. Calculates reliability by reading the voltage and current waveforms from SPICE output

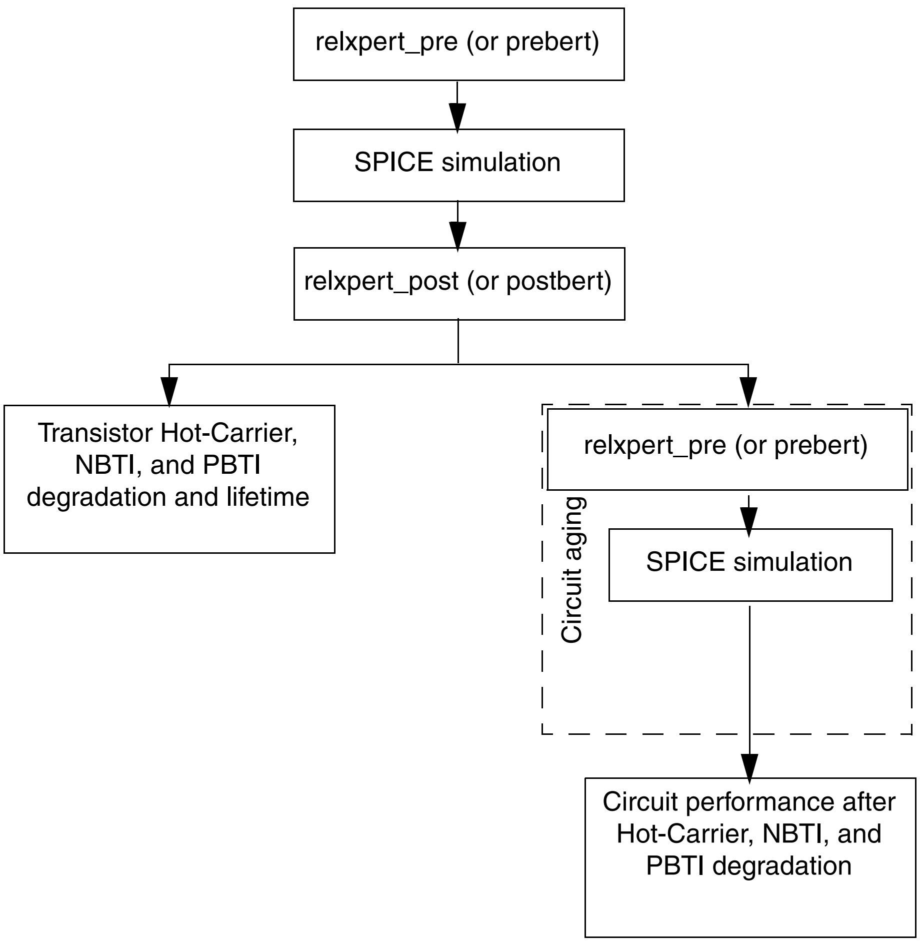
To use the Virtuoso RelXpert simulator,

1. Run relxpert_pre (or prebert).


© 2003-2014 All Rights Reserved.

2. Run the SPICE simulation.
3. Run relxpert_post (or postbert).
4. For Hot-Carrier circuit aging simulation, run a second-pass simulation by running relxpert_pre -age (or prebert2) again, and then SPICE.

**Figure 11-1 Sequence of Virtuoso RelXpert Simulations**




# Batch Mode

In batch mode, Virtuoso RelXpert reliability simulation is performed by running relxpert_pre (or prebert) and relxpert_post (or postbert) from the UNIX command line.


© 2003-2014 All Rights Reserved.

## relxpert_pre (or prebert/prebert2)

`relxpert_pre` *`options inputFile outputFile`*

### Arguments

| | | |
|---|---|---|
| *`options`* | Can be any of the following: | |
| | `-64` | Activates the 64-bit binary of RelXpert to perform reliability simulation. |
| | `-p1` | Forces pass 1 Hot-Carrier simulation. If `-p1` is not specified, relxpert_pre (or prebert) searches for the file *`basename`*`.ba0` in the current directory where *`basename`* is the input file without the suffix. If this file exists, relxpert_pre (or prebert) runs a pass 2 simulation using the age information in *`basename`*`.ba0`. If this file is not found, pass 1 simulation is performed. It is good practice to always use `-p1` for first-pass Hot-Carrier simulation. |
| | `-age` (or `-p2`) *`fileName`* | Forces pass 2 Hot-Carrier simulation. The age information is read from *`fileName`*. Usually *`fileName`* has the extension `.ba0`.<br>Running `prebert2` is equivalent to running `prebert -p2`. |
| | `-si` | Selects the TITAN interface. You must use this option if you are using TITAN as your circuit simulator. |
| | `-sp` | Selects the Virtuoso Spectre interface. You must use this option if you are using Virtuoso Spectre as your circuit simulator. |
| | `-st` | Selects the TI Spice3 interface. You must use this option if you are using TI Spice3 as your circuit simulator. |
| | `-I includePath` | Include path for the include file in the netlist. |


© 2003-2014 All Rights Reserved.

| | |
|---|---|
| `-k` | In pass 2 aging simulation, deletes `rwamdx` files (degraded transistor model parameter file) after it is done. This option is used to keep the `rwamdx` files for future use. |
| `+lqtimeout` (or `+lqt`) *`value`* | Specfies the duration (in seconds) for which the simulator should wait for a license to be available. If 0 is specified, the simulator will wait until the license is available. The default value is 900 seconds. |
| `+log` *`file`* | Dump the log messages to *`file`*. You may use `+l` as an abbreviation for `+log`. |

`+mt=N | +multithread=N`

Enables multithreading and specifies the number of threads to be used. The maximum number of threads that can be specified is 16.

**Note:** Multithreading is enabled by default.

`+mt | +multithread`

Enables multithreading. When the number of threads is not specified, the Virtuoso RelXpert Reliability simulator automatically detects the number of processors and selects the appropriate number of threads to use.

**Note:** Multithreading is enabled by default.

`-mt | -multithread`

Disables multithreading capability. Since multithreading is enabled by default, use this option if you do not want to use multithreading.

| | |
|---|---|
| `-dc` | Runs the DC degradation calculation case. |
| `-V` | Displays the version of the program. |
| `-W` | Displays the sub-version of the program. |


© 2003-2014 All Rights Reserved.

| | | |
|---|---|---|
| | `-uricheck <libName>.extension` | Enables URI library file checking. If this option is specified, `relexpert_pre` ignores all other options and just performs URI library checking. You can specify multiple URI libraries within quotation marks, for example, "`urilib1.so urilib2.so`". |
| | `-help` | Prints relxpert_pre (prebert) help information. |
| *`inputFile`* | The file that contains SPICE netlist file, Virtuoso RelXpert simulator commands, and models.<br><br>**Note:** You can read in an encrypted model while running relxpert_pre. | |
| *`outputFile`* | The input file for SPICE. | |

Files created by relxpert_pre (or prebert) are:

- A file with the suffix `.bp`, which contains the filtered Virtuoso RelXpert simulator commands

  This file is read by relxpert_post (or postbert).

- A file with the suffix `.bd`, which contains the device information

  This file is read by relxpert_post (or postbert).

- `rwmd`*`xxx`* files, which contain model parameters

  These files are read and deleted by relxpert_post (or postbert).

- `rwx`*`yyy`* directories, which include the temporary files which store subcircuit definitions

  These files and directories are erased by relxpert_pre (or prebert).

Files needed by relxpert_pre (or prebert) in second-pass simulation that are created by relxpert_post (or postbert) are

- Files with the suffix `.ba0`, which contain the transistor degradation results for Hot-Carrier aging simulation

  relxpert_pre (or prebert) searches for the file *`baseName`*`.ba#` where *`baseName`* is the input file without the suffix.


© 2003-2014 All Rights Reserved.

## relxpert_post (or postbert)

`relxpert_post` *`options inputFile`*

### Arguments

*`options`* Can be any of the following:

`-64` Activates the 64-bit binary of RelXpert to perform reliability simulation.

`-r` Normally relxpert_post (or postbert) reads the waveform data from the file with the extension `.xp` (for ADM) or `.raw` (for Berkeley SPICE). The basename is the input file without the suffix. If you type

```
relxpert_post cmos1.out
```

relxpert_post (or postbert) reads the waveform data file with extension depending on the type of SPICE simulator selected. (For example, if ADM is the SPICE simulator, the input waveform is `cmos1.xp`)

If you are using the Virtuoso Spectre simulator, you need to specify the waveform data file using the `-r` option because Virtuoso Spectre output does not have a default extension. Usually, the waveform data file is stored (by the Virtuoso Spectre simulator) in the file `timeSweep.tran` or `tran.tran` under the `.raw` directory. For example, if your netlist file name is `cmos1`, after running relxpert_pre (or prebert) and bertspectre, your command line for relxpert_post (or postbert) is

```
relxpert_post -r cmos1.raw/timeSweep.tran
cmos1.out
```

You may also use this option to include an input waveform filename that is different from the above default extension.


© 2003-2014 All Rights Reserved.

`-rawfmt sst2|psfbin|psfascii|psfxl`

Specifies the waveform format. The supported formats are `sst2`, `psfbin`, `psfascii`, and `psfxl`.

The waveform format can also be set by using the `rawfmt` option in the netlist file.

**Note:** The command line option has higher priority than the option set in the netlist.

`-k`

Normally relxpert_post (or postbert) deletes `rwmdx` files (transistor model parameter files created by relxpert_pre) after it is done. This option keeps the `rwmdx` files.

`-I inclulePath`

Specify the include path for the include statements in netlist.

`+lqtimeout` (or `+lqt`) *`value`*

Specfies the duration (in seconds) for which the simulator should wait for a license to be available. If 0 is specified, the simulator will wait until the license is available. The default value is 900 seconds.

`+mt=`*`N`* `|` `+multithread=`*`N`*

Enables multithreading and specifies the number of threads to be used. The maximum number of threads that can be specified is 16.

**Note:** Multithreading is enabled by default.

`+mt | +multithread`

Enables multithreading. When the number of threads is not specified, the Virtuoso RelXpert Reliability simulator automatically detects the number of processors and selects the appropriate number of threads to use.

**Note:** Multithreading is enabled by default.

`-mt | -multithread`

Disables multithreading capability. Since multithreading is enabled by default, use this option if you do not want to use multithreading.


© 2003-2014 All Rights Reserved.

| | | |
|---|---|---|
| | `-V` | Displays the version of the program. |
| | `-W` | Displays the sub-version of the program. |
| | `-help` | Prints relxpert_post (or postbert) help information. |
| *`inputFile`* | The output list file from SPICE simulation. relxpert_post (or postbert) reads transient waveforms from *`baseName`*`.raw` (if Berkeley SPICE is used), where *`baseName`* is the input file without the suffix.<br><br>**Note:** If Virtuoso Spectre or TI Spice3/4 is used, you need to use the `-r` option to specify input waveform. | |

The output file from relxpert_post (or postbert) is *`baseName`*`.bo0`.

Files needed by relxpert_post (or postbert) in the second pass that are created by relxpert_pre (or prebert) are

- Files with the suffix `.bp0`, which contain the Virtuoso RelXpert simulator commands

  relxpert_post (or postbert) searches for the file *`baseName`*`.bp` where *`baseName`* is the input file without the suffix.

- Files with the suffix `.bd`, which contain the list of circuit elements for reliability calculation

  relxpert_post (or postbert) searches for the file *`baseName`*`.bd` where *`baseName`* is the input file without the suffix.

Files created by relxpert_post (or postbert) are

- A file with the suffix `.bo0`, which contains the simulation results
- A file with the suffix `.ba0`, which contains the age information for second-pass Hot-Carrier simulation, to be read by relxpert_pre (or prebert)


© 2003-2014 All Rights Reserved.

## rxprofile

`rxprofile [-V] [-W] [-raw dirname] [cfgfile_name]`

or

`rxprofile <-sh | -st | -sp | -se> input_netlist`

### Description

Tool for running of profiling or gradual-aging. The input can be either a configuration file or a netlist. If the input is a netlist, one of `-sh`, `-st`, `sp`, or `-se` arguments must be specified.

### Arguments

| | |
|---|---|
| `-raw` *`dirname`* | Specifies the temporary file directory. All the generated result files are stored in the directory *`dirname`* during the rxprofile simulation. If *`dirname`* is not specified, all temporary files are stored in the `.rxprofile_temp` directory. |
| `-V` | Displays the program version. |
| `-W` | Displays the program sub-version. |
| `-help` | Prints rxprofile help information. |
| *`cfgfile_name`* | Specifies the name of the configuration file in which gradual aging options are specified.<br><br>If not specified in the command line, the default configuration file name is `profile.cfg`. |
| `-sh\|-sp\|-st` | Specifies the SPICE simulator used in the simulation. It can be either hspice (`-sh`}, Spectre (`-sp`) or tispice (`-st`). |
| `input_netlist` | Specifies the netlist. The netlist should contain the gradual aging options. |

### Configuration File

You can specify the following information in the configuration file:

- Netlist File Type
- SPICE Simulator Path


© 2003-2014 All Rights Reserved.

- Reference Netlist Name
- Temporary File Name
- Gradual Aging Flow Specification

### *Netlist File Type*

`.netlisttype [spectre | tispice | hspice]`

#### Description

Specifies the SPICE simulator type for RelXpert. If not specified, the default simulator is Spectre.

#### Arguments

| | |
|---|---|
| `spectre` | Specifies that the netlist type is Spectre. |
| `tispice` | Specifies that the netlist type is Tispice. |
| `hspice` | Specifies that the netlist type is Hspice. |

#### Example

```
.netlist_type spectre
```

### *SPICE Simulator Path*

`.spice_path {`*`spice_path`*`}`

#### Description

Specifies a different SPICE simulator to perform the simulation (optional). If not specified, the SPICE path specified in the work environment is used by default.

#### Arguments

| | |
|---|---|
| *`spice_path`* | Specifies the path to the SPICE simulator binary. |

#### Example

```
.spice_path /tools/spectre/lnx86/tools.lnx86/bin/spectre
```


© 2003-2014
All Rights Reserved.

### *Reference Netlist Name*

```
.reference_name { netlist_name }
```

#### Description

Specifies a reference netlist, which is run for every stress aging step, to compare with the typical aging. The difference between a reference netlist and a typical netlist is the bias or temperature. The circuit's topology in the reference netlist should be the same as the typical aging netlist.

#### Arguments

| | |
|---|---|
| *netlist_name* | Specifies the name of the reference netlist. |

Consider the following points when using this option:

- A fresh SPICE simulation for the reference netlist will be run automatically.
- The reference aged netlist file is suffixed with .p4.
- This option can work with the `.iteration`, `.agestep`, `.agepoint`, and `.profile_names` options.

#### Example

```
.reference_name reference.ckt
```

### *Temporary File Name*

```
.save_tempfile  step=[step_num]
```

#### Description

Saves the temporary files that are generated in the specified step of `rxprofile` flow. The result files of the first step and the last step of the gradual aging flow are always saved.

The temporary files with the suffix `.ba0`, `.p1`, `.p3`, `.p4` and the SPICE waveform file in step one are saved under the directory name specified with the `-raw` option of the `rxprofile` command.

The `.save_tempfile` option can be used with `.profile_names`, `.iteration`, `.agestep`, and `.agepoint` options. If this option is not specified, all temporary files are saved in the directory name specified by the `-raw` option of `rxprofile`.


© 2003-2014 All Rights Reserved.

### Arguments

| | |
|---|---|
| *step_num* | Specifies an index list of the steps for which the temporary files need to be saved. |

Example

```
.save_tempfile step=1 3 5...
```

Specifies that the result files including `.ba0`, `.p1`, `.p3`, `.p4` and the waveform `.raw(tr0)` for step 1, 3, and 5 will be saved in the directory name specified with the `-raw` option of `rxprofile`. In addition, the result file of fresh simulation and last step will also be saved in the same directory.

**Note:** The syntax for `.save_tempfile` has been made consistent with Spectre (`gradual_aging_save`). However, the following old format is still supported:

```
.save_tempfile  { step_list }
```

### *Gradual Aging Flow Specification*

When running rxprofile simulation, you can select one of the following gradual aging flows:

**Note:** Only one of the four gradual aging options should be specified. If all the four options are specified together, the `.agepoint` option takes priority and the other options are ignored. The following list shows the option names in the order of decreasing priority.

- `.agepoint`
- `.iteration`
- `.agestep`
- `.profile_names`

➡ To specify the netlist file list, use the .profile_names option.

```
.profile_names  { netlist_files }
```

Description

This option should be specified in the rxprofile simulation. The netlist files specified with this option are identical except for the stress voltage and temperature values.

Argument

- *`netlist_name`*: Specifies the netlist name for the rxprofile simulation.


© 2003-2014 All Rights Reserved.

Example

```
.profile_names  test1.ckt  test2.ckt  test3.ckt...
```

Specifies the `test.ckt`, `test2.ckt`, and `test3.ckt` as the testcase in rxprofile simulation.

➡ To specify the iteration method for gradual aging flow, use the .iteration option.

```
.iteration  {netlist_name}  {iteration_num}
```

Description

This option is used when the same netlist is used in the rxprofile simulation. The output result files are suffixed with the iteration index.

Arguments

- *`netlist_name`*: Specifies the netlist name.
- *`iteration_num`*: Specifies the step number in rxprofile simulation.

Example

```
.iteration test.ckt 10
```

Specifies that the total number of steps in rxprofile simulation is 10 and the name of the test case is test.ckt.

➡ To specify the agestep method as the gradual aging flow, use the .agestep option:

```
.agestep type=[log|lin] start=start_time stop=stop_time
total_step=total_step_num
```

Description

This option is used to specify the rxprofile simulation from the start time to the stop time. The total steps of rxprofile are specified by *`total_step_num`*. Two age step types, linear and logarithm, can be specified.

Arguments

- `type`: Specifies the type of agestep, `linear` or `logarithm`. The default type is `linear`. When you specify linear, the time step is calculated using the following formula:

  `(stop_time-start_time)/(total_step-1)`

  When you specify `logarithm`, the time step is calculated using the following formula:

  `[log(stop_time)-log(start_time)]/(total_step-1)`


© 2003-2014
All Rights Reserved.

- `start`: The start time of rxprofile simulation. Default is 0.0 for linear type and 1.0 for logarithm.
- `stop`: The stop time of rxprofile simulation.
- `total_step`: The total step of rxprofile simulation. The value should be larger than 1.

**Note:** The time unit can be set as y(year), d(day), h(hour), m(minute) and s(second)

Example:

```
.agestep type=log start=1y stop=10y total_step=6
```

Specifies that the agestep method will be used for the gradual aging flow where the type of agestep is `logarithm`, the start and stop time for simulation is 1 year and 10 years, respectively, and the total steps for rxprofile simulation is 6.

**Note:** The syntax for `.agestep` has been made consistent with Spectre (`gradual_aging_agestep`). However, the following old format is still supported:

```
.agestep { netlist_name } [ type = [log | lin] ] [ start = start_time ] {
stop = stop_time }  { total_step = total_step_num }
```

➡ To specify the agepoint method as the gradual aging method, use the `.agepoint` option.

```
.agepoint { points= age_point_list }
```

Description

This option is used to specify the selected age points to perform rxprofile simulation. The results files for each step are suffixed with age point values.

Arguments

- *`age_point_list`*: Specifies the age point list. Note that the point's value should not exceed 0. The value in the list should be arranged in ascending order and no negative value should be specified.

**Note:** The .age command in the netlist will be ignored

Example

```
.agepoint points= 1y 3y 5y 8y ...
```

Specifies that the rxprofile simulation be run for age points 1y, 3y, 5y, and 8y.

**Note:** The syntax for `.agepoint` has been made consistent with Spectre (`gradual_aging_agepoint`). However, the following old format is still supported:

```
.agepoint {netlist_name} { points= age_point_list }
```

- Comment lines


© 2003-2014 All Rights Reserved.

All statements that begin with `*` or `#` in the configuration file are treated as comments.

### Examples

```
rxprofile -raw case_temp
```

All temporary files are stored in the `case_temp` directory (the configuration file is named `profile.cfg`).

```
rxprofile case.cfg
```

All temporary files are stored in the `.rxprofile_temp` directory and rxprofile runs the simulation according to the contents of `case.cfg`.

```
rxprofile -raw result_file file.cfg
```

All the result files including .p1 .p3 .p4 .ba0 and waveform .raw(.tr0) files will be saved in the result_file directory.

### Sample Configuration File

Here is a sample `case.cfg` configuration file:

```
.netlist_type spectre
.spice_path /tools.lnx86/bin/spectre
.agestep test.ckt total_step=6 type=log start=100d stop=5y
.reference_name agemos.ckt
.save_tempfile 1 3 5
```

rxprofile also enables you to input a netlist when `-sh|-sp|-st` option is specified. Instead of using a configuration file to specify gradual aging options, you can specify the following gradual aging options directly in the netlist,

```
*relxpert: gradual_aging_agestep type=[linear|log] start=start_time stop=stop_time
     total_step=total_step_num
```

This option is the same as the `.agestep` option in the configuration file for specifying the aging steps for gradual aging.

```
*relxpert: gradual_aging_agepoint points=[age_point_list]
```

This option is the same as the `.agepoint` option in the configuration file for specifying the aging points for gradual aging.

```
*relxpert: gradual_aging_spice_path path=spice_exec_path
```

This option is the same as `.spice_path` option in the configuration file for specifying the SPICE executable path.


© 2003-2014 All Rights Reserved.

```
*relxpert: gradual_aging_save steps=[step_list]
```

This option is the same as the `.save_tempfile` option in the configuration file for saving temporary files.

### Example of gradual aging options in netlist

```
Simulator lang=spice
*relxpert: .age 10y
*relxpert: .deltad 0.1
*relxpert: .gradual_aging_agepoint points=[1y 3y 5y 10y]
```

# Scripts

## RUN_RELXPERT

`RUN_RELXPERT` *`options inputFile`*

`RUN_RELXPERT -si` *`inputFile savename`*

### Description

The Virtuoso RelXpert run script runs all steps for reliability simulation. The input netlist file can be used with Virtuoso Spectre or TI Spice3/4 format. This script runs through relxpert_pre (or prebert) and SPICE simulation on a fresh netlist file and `relxpert_post` (or postbert), `relxpert_pre -age` (or prebert2), and SPICE simulation on a degraded netlist file for a single run.

Starting with the MMSIM 13.1 release, you can run the Virtuoso RelXpert run script with two input netlist files (stress and measure). The measure netlist file can be specified using the `-p2netlist` file option. It replaces the fresh netlist file during the `relxpert -pre -age` (`prebert2`) step.

### Arguments

| | | |
|---|---|---|
| *`options`* | Can be any of the following: | |
| | `-help` | Prints script help information. |


© 2003-2014 All Rights Reserved.

| | |
|---|---|
| `-saps` | Runs the netlist in APS flow. The *`inputFile`* needs to be in Virtuoso APS format and the simulator should be Virtuoso APS. |
| `-sp` | Runs the netlist file in the Virtuoso Spectre flow. The `inputFile` needs to be in Virtuoso Spectre format and Virtuoso Spectre is the simulator. |
| `-si` | Runs the netlist file in the TITAN flow. The `inputFile` needs to be in TITAN format and TITAN is the simulator. |
| `-st` | Runs the netlist file in the TISPICE flow. The `inputFile` needs to be in TISPICE format and TI Spice3.4 is the simulator. |
| `-st4` | Runs the netlist file in the TISPICE4 flow. The `inputFile` needs to be in TISPICE4 format. |
| `-prebertarg` | Additional arguments to pass to a `prebert` run. |
| `-postbertarg` | Additional arguments to pass to a `postbert` run. |
| `-prebert2arg` | Additional arguments to pass to a `prebert2` run. |
| `-spicearg` | Additional arguments to pass to a SPICE run. |
| `-p2netlist` *`measure_file`* | Specifies a measure netlist file that replaces the fresh netlist file during the `relxpert -pre -age` (`prebert2`) step. |
| `-32\|-64` | Runs RelXpert binary in 32 or 64 bit (default is 32 bit) |

*`inputFile`*

Top netlist filename (file needs to be located in the current directory).

*`savename`*

Directory where the simulation results are stored (defined in the original TITAN netlist file with `.save`). The argument must be stated in the command line for the TITAN flow.


© 2003-2014 All Rights Reserved.

### Example

```
RUN_RELXPERT -sp cmos1.sp
RUN_RELXPERT -si de.inc PSF_save
RUN_RELXPERT -sp -64 input.scs -spicearg '-f psfxl' -postbertarg '-f psfxl'
RUN_RELXPERT -sp input.scs -p2netlist input2.scs
```

# TMI Aging Flow

TMI is a modeling application programming interface (API) that supports extensions of standard compact models. It is an add-on to the standard models. TMI aging is an API that also supports aging simulation. RelXpert supports the TMI aging flow.

For the TMI aging flow, you only need to set the RelXpert control statements and RelXpert automatically maps them to the corresponding TMI aging parameters in the TMI shared library and runs the TMI aging flow.

The following table shows the TMI aging parameters mapped by the RelXpert control statements:

| RelXpert Statements | TMI Aging Parameters | Description |
|---|---|---|
| `.age` | `Dagetime` | Specifies the aging time |
| `.hci_only`<br>`.nbti_only`<br>`.pbti_only` | `phys_id=0|1|2` | Simulates HCI, NBTI, or PBTI |
| `.relx_tran` | `tr_start`, `tr_end` | Reliability transient start/stop time interval |
| `.deg_ratio` | `r_hci`, `r_bti` | Multiplication factor for HCI or BTI degradation |

The following TMI options are supported and can be controlled by the RelXpert TMI aging flow:

- `.option tmiflag=0|1` - enables or disables the TMI flow.
- `.option tmipath` - specifies the location where the TMI compiled shared libraries are stored.
- `.option tmiinput` - specifies the location of the TMI configuration file.
- `.option tmioutput` - specifies the location to generate the TMI model configuration file.


© 2003-2014 All Rights Reserved.

- `.option tmiage=0|1` - enables or disables the TMI aging flow.
- `.option tmisave=0|1` - enables or disables saving of age data to a file.

For more information on the TMI parameters and options, refer to the TMI documentation.


© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 12

# Simulator Output

The simulation results from the Virtuoso® RelXpert reliability simulator are stored in the output file with the suffix `.bo0`. The output from both modules is described in detail.

## MOSFET Hot-Carrier Simulation

The results from MOSFET Hot-Carrier simulation are given in the MOSFET HCI degradation tables. This is requested through the `.degprint` and `.degsort` commands.

The results for NMOSFET and PMOSFET are in separate tables. Simulation results are printed for every transistor in the circuit (if `.degsort` or `.degprint` is used) or for the specified transistors in the `.degsort` command (see ".degsort" on page 110).

The results printed are

- Maximum and average substrate current in Amperes
- Maximum and average gate current in Amperes
- If `.printisub_idsmax` (see ".printisub_idsmax" on page 158) is used, the average gate current will be replaced by $(I_{sub}/I_{ds})max$ values
- Lifetime of the transistor to reach the failure criterion specified in the `.deltad` command (see ".deltad" on page 112)

  The degradation can be transconductance ($\Delta g_m/g_m$), linear or saturation drain current ($\Delta I_d/I_d$) degradation, threshold voltage shift ($\Delta V_t$), or any other degradation monitor. The selection of this quantity depends on the values of $H$ and $m$ chosen in Equation 3-8 on page 51 or Equation 3-9 on page 51.

- The transistor degradation after the time specified in the `.age` command (see ".age" on page 95)

  The degradation can be transconductance ($\Delta g_m/g_m$), linear or saturation drain current ($\Delta I_d/I_d$) degradation, threshold voltage shift ($\Delta V_t$), or any other degradation monitor. The selection of this quantity depends on the values of $H$ and $m$ chosen in Equation 3-8 on page 51 or Equation 3-9 on page 51.


© 2003-2014
All Rights Reserved.

- The calculated $Age$ of the transistor (Equation 2-37 on page 36 and Equation 2-38 on page 37)

Transistors experiencing “bad” AC waveforms are indicated with an asterisk (`*`). The “bad” AC waveform is defined according to the criteria specified in the `.checkbadac` command.

In addition, the following plots are generated:

- The total substrate current in the circuit if requested through `.printisubtotal` (see “.printisubtotal” on page 157).

  Text output will be appended to the output file.

- The total gate current in the circuit if requested through `.printigatetotal` (see “.printigatetotal” on page 155).

  Text output will be appended to the output file.

- The substrate current for the transistor specified in `.printisub` (see “.printisub” on page 156).

  Text output will be appended to the output file.

- The gate current for the transistor specified in `.printigate` (see “.printigate” on page 154).

  Text output will be appended to the output file.

The age of all transistors is stored in a file with the suffix `.ba0`. This file is read by relxpert_pre (or prebert) during the second pass for circuit aging simulation.

The information stored in the file is

- Transistor name with subcircuit call name
- The age in the forward and reverse modes of transistor operation

  The forward mode is defined when the HCI degradation damage is found at the first (drain) node of the transistor.

- The total age of the transistor without considering forward or reverse mode operation

# PMOSFET NBTI Simulation

The results from PMOSFET NBTI simulation are given in the MOSFET NBTI degradation tables. This is requested through the `.degprint` and `.degsort` commands.


© 2003-2014
All Rights Reserved.

Simulation results are printed for every transistor in the circuit (if the `.degprint` or `.degsort` command is used) or for the specified transistors if the `.degsort` command is used (see ".degsort" on page 110).

The results printed are

- Lifetime of the transistor to reach the failure criterion specified in the `.deltad` command (see ".deltad" on page 112)

  The degradation can be transconductance ($\Delta g_m/g_m$), linear or saturation drain current ($\Delta I_d/I_d$) degradation, threshold voltage shift ($V_t$) or any other degradation monitor. The selection of this quantity depends on the criterion used in NBTI lifetime parameter extraction.

- The transistor degradation after the time specified in the `.age` command (see ".age" on page 95)

  The degradation can be transconductance ($\Delta g_m/g_m$), linear or saturation drain current ($\Delta I_d/I_d$) degradation, threshold voltage shift ($V_t$) or any other degradation monitor. The selection of this quantity depends on the criterion used in NBTI lifetime parameter extraction.

- The calculated age of the transistor

If there are both HCI and NBTI simulations for PMOSFET, there is also a table with Hot-Carrier and NBTI combination results.

The age of all transistors is stored in a file with the suffix `.ba0`. This file is read by relxpert_pre (or prebert) during the second pass for circuit aging simulation.

The information stored in the file is

- Transistor name with subcircuit call name
- For a specified PMOSFET, the first line is the HCI $Age$ value, the second line is the NBTI $Age$ value; the age in the forward and reverse modes of transistor operation

  The forward mode is defined when the NBTI degradation damage is found at the first (drain) node of the transistor.

- The total age of the transistor without considering forward or reverse mode operation

If there are both HCI and NBTI simulations for PMOSFET, both HCI $Age$ and NBTI $Age$ are printed in the file with the suffix `.ba0`.


© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# 13

# Examples of Simulations

The example in this chapter demonstrates the model setup, simulation, and output of the Virtuoso® RelXpert reliability simulator for Hot-Carrier and NBTI degradation in MOS transistors. Virtuoso Spectre is used as the SPICE simulator in this example.

## Demo Files

The model and example files discussed in this chapter are found in

`/relxpert/example/spectre`

The directory `spectre/` contains the following:

| | |
|---|---|
| `/model` | Directory containing the models used in the Virtuoso Spectre example |
| `ringos` | SPICE input deck for MOSFET Hot-Carrier and NBTI degradation simulation of a ring oscillator |
| `run.csh` | Script for running simulation in batch mode |
| `clean.csh` | Script for cleaning generated files during simulation |

The directory `spectre/model` contains the following:

| | |
|---|---|
| `mos.scs` | NMOSFET and PMOSFET fresh SPICE models with lifetime and agemos parameters. |

## MOSFET Hot-Carrier and NBTI Simulation

The relevant files for this example are `example/spectre/ringos` and `example/spectre/model/mos.scs`. Included in the `.model` statement are the Hot-CarrierHot-


© 2003-2014 All Rights Reserved.

Carrier and NBTI parameters. Agemos parameters are included in the `.model` statement for NMOSFET and PMOSFET. They are used for generating aged model parameters for aging.

## Substrate Current Parameters

Substrate current parameters are determined from measurements on fresh transistors (using the BSIMProPlus® model extractor from ProPlus Design Solutions Inc) and are given in the following table.

**Table 13-1 Parameters Needed for Substrate and Gate Current Calculations**

| Parameters | Name Used in RelXpert | Units | Values |
|---|---|---|---|
| NMOSFET substrate current parameters | | | |
| $A_i$ | Ai | 1/cm | $7.2822 \times 10^6$ |
| $B_i$ | Bi | V/cm | $1.7 \times 10^6$ |
| $E_{crit0}$ | Ecrit0 | V/cm | $7.0122 \times 10^4$ |
| $E_{critg}$ | Ecritg | 1/cm | $9.6005 \times 10^3$ |
| $E_{critb}$ | Ecritb | 1/cm | 0.0 |
| $l_{c0}$ | lc0 | $\mu m^{1/2}$ | 1.467 |
| $l_{c1}$ | lc1 | $\mu m^{1/2}/V$ | $1.0629 \times 10^{-1}$ |
| $E_a$ | Ea | eV | 0.0 |
| PMOSFET gate current parameters | | | |
| $A_i$ | Ai | 1/cm | $7.4822 \times 10^6$ |
| $B_i$ | Bi | V/cm | $3.7 \times 10^6$ |
| $E_{crit0}$ | Ecrit0 | V/cm | $1.0233 \times 10^5$ |
| $E_{critg}$ | Ecritg | 1/cm | $3.75 \times 10^4$ |
| $E_{critb}$ | Ecritb | 1/cm | 0.0 |
| $l_{c0}$ | lc0 | $\mu m^{1/2}$ | 1.0402 |
| $l_{c1}$ | lcg1 | $\mu m^{1/2}/V$ | $1.03 \times 10^{-1}$ |
| $E_a$ | Ea | eV | 0.0 |


© 2003-2014 All Rights Reserved.

## Hot-Carrier Lifetime Parameters

Hot-Carrier stress experiments were performed on 0.25 μm n- and p-channel transistors. The following lifetime parameters are extracted by the BSIMProPlus® model extractor from ProPlus Design Solutions Inc.

**Table 13-2 Parameters Needed for MOSFET Hot-Carrier Degradation Simulation**

| Parameters | Name Used in RelXpert | Units | Values |
|---|---|---|---|
| NMOSFET lifetime parameters | | | |
| $H_0$ | H0 | $(A/m) \cdot \Delta D^{-1/n}$ | 8.3413 x $10^4$ |
| $H_{gd}$ | Hgd | $V^{-1} \cdot A \cdot m^{-1} \cdot \Delta D^{-1/n}$ | 0.0 |
| $m_0$ | m0 | | 1.9603 |
| $m_{gd}$ | mgd | $V^1$ | 0.0 |
| $n_0$ | nn0 | | 0.34775 |
| $n_{gd}$ | nngd | $V^1$ | 0.0 |
| PMOSFET lifetime parameters | | | |
| $H_0$ | Hg0 | $(A/m)^{m_g} \cdot s$ | 6.238 x $10^6$ |
| $H_{gd}$ | Hggd | $V^{-1} \cdot (A/m)^{m_g} \cdot s$ | 0.0 |
| $m_0$ | mg0 | | 1.1691 |
| $m_{gd}$ | mggd | $V^1$ | 0.0 |
| $n_0$ | nn0 | | 0.1875 |
| $n_{gd}$ | nngd | $V^1$ | 0.0 |

## NBTI Lifetime Parameters

NBTI stress experiments are performed on p-channel transistors with different channel lengths. NBTI lifetime parameters are extracted by the BSIMProPlus® model extractor from ProPlus Design Solutions Inc. The following shows the NBTI lifetime parameters used in NBTI simulation.


© 2003-2014
All Rights Reserved.

nbn = 0.20225

nba = 1

nbea = 0.055226

nbgamma = 1.6

nbgammad = 0.58759

nbalpha = 1

nbeta0 = 0.0015588

nbeta1 = 20.621

nbeta2 = -1

nbemod = 2


© 2003-2014 All Rights Reserved.

## Agemos Parameters

Agemos parameters are extracted by the BSIMProPlus® model extractor from ProPlus Design Solutions Inc. The following tables show the agemos parameters for Hot-Carrier and NBTI simulations.

**Table 13-3 Agemos Parameters for NMOSFET HCI Simulation**

| Parameters | Name Used in RelXpert | Values |
|---|---|---|
| a0 | hd1_a0 | 3.2384 |
| | hd2_a0 | 2.9328 |
| | hn1_a0 | 0.1129 |
| | hn2_a0 | 0.31186 |
| | hs_a0 | 4.3238 |
| | hsign_a0 | -1 |
| ub | hd1_ub | 5.887 |
| | hd2_ub | 0.0 |
| | hn1_ub | 0.42988 |
| | hn2_ub | 0.0 |
| | hs_ub | 1.0 |
| | hsign_ub | 1.0 |
| vsat | hd1_vsat | 0.4575 |
| | hd2_vsat | 0.0 |
| | hn1_vsat | 0.33652 |
| | hn2_vsat | 1.0 |
| | hs_vsat | 1.0 |
| | hsign_vsat | -1.0 |
| vth0 | hd1_vth0 | 8.208 |
| | hd2_vth0 | 1.1738 |
| | hn1_vth0 | 0.4326 |
| | hn2_vth0 | 0.19745 |
| | hs_vth0 | 2.829 |


© 2003-2014 All Rights Reserved.

**Table 13-3 Agemos Parameters for NMOSFET HCI Simulation,** *continued*

| Parameters | Name Used in RelXpert | Values |
| --- | --- | --- |
| | hsign_vth0 | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 13-4 Agemos Parameters for PMOSFET HCI Simulation**

| Parameters | Name Used in RelXpert | Values |
|---|---|---|
| a0 | hd1_a0 | 2.4482 |
| | hd2_a0 | 0.0 |
| | hn1_a0 | 0.495 |
| | hn2_a0 | 0.0 |
| | hs_a0 | 1.0 |
| | hsign_a0 | -1 |
| ub | hd1_ub | 2.9838 |
| | hd2_ub | 0.0 |
| | hn1_ub | 0.37357 |
| | hn2_ub | 0.0 |
| | hs_ub | 1.0 |
| | hsign_ub | 1.0 |
| vsat | hd1_vsat | 2.5 |
| | hd2_vsat | 0.0 |
| | hn1_vsat | 0.19051 |
| | hn2_vsat | 0.0 |
| | hs_vsat | 1.0 |
| | hsign_vsat | -1.0 |
| vth0 | hd1_vth0 | 1.79 |
| | hd2_vth0 | 0.0 |
| | hn1_vth0 | 0.16986 |
| | hn2_vth0 | 0.0 |
| | hs_vth0 | 1.0 |
| | hsign_vth0 | 1.0 |


© 2003-2014 All Rights Reserved.

**Table 13-5 Agemos Parameters for PMOSFET NBTI Simulation**

| Parameters | Name Used in RelXpert | Values |
|---|---|---|
| ub | nd1_ub | 0.4196 |
| | nd2_ub | 0.0 |
| | nn1_ub | 0.45 |
| | nn2_ub | 0.0 |
| | ns_ub | 1.0 |
| | nsign_ub | 1.0 |
| vsat | nd1_vsat | 0.3217 |
| | nd2_vsat | 2.6701 |
| | nn1_vsat | 0.18025 |
| | nn2_vsat | 0.12551 |
| | ns_vsat | 4.25 |
| | nsign_vsat | -1.0 |
| vth0 | nd1_vth0 | 0.4 |
| | nd2_vth0 | 3.4562 |
| | nn1_vth0 | 0.083139 |
| | nn2_vth0 | 0.2537 |
| | ns_vth0 | 3.347 |
| | nsign_vth0 | 1.0 |

## Commands for Virtuoso RelXpert Simulation

The SPICE netlist file `example/spectre/ringos` contains a ring oscillator.

Examples of Virtuoso RelXpert simulator commands for MOSFET Hot-Carrier and NBTI simulations are included in the SPICE netlist file `example/spectre/ringos` as follows:

```
*relxpert: .age 10yr
*relxpert: .deltad 0.1
*relxpert: .agemethod agemos
*relxpert: .degsort
```


© 2003-2014
All Rights Reserved.

The `.age` card (".age" on page 95) requests the simulator to predict degradation after this time period of stressing and also instructs the simulator (in the second-pass aging simulation) to generate degraded transistor models after this time period of stressing.

The `.deltad` card (".deltad" on page 112) requests the simulator to predict the lifetime of the transistor under the circuit operation condition to reach a 10% degradation. The degradation monitor depends on how the parameters $H$ and $m$ are set up (Equation 3-8 on page 51 and Equation 3-9 on page 51). In this example, the degradation corresponds to 10% Idsat degradation for NMOSFET and PMOSFET.

The `.agemethod` card (".agemethod" on page 98) specifies that the agemos method be used to calculate degraded model parameters for the second-pass aging simulation.

The `.degsort` card (".degsort" on page 110) requests the simulator to print out the degradation results. The results are sorted from the worst to the least degradation.

## Simulation Results

The Virtuoso RelXpert simulation was done in the batch mode by issuing the following commands:

```
relxpert_pre -sp ringos ringos.p1
spectre ringos.p1 > ringos.out
relxpert_post -r ringos.raw/tran.tran ringos.out
```

The post-processor relxpert_post (or postbert) writes the simulation results in the file `ringos.bo0`. This file can be viewed by calling up a text editor.


© 2003-2014 All Rights Reserved.

## Table 13-6 NMOSFET Hot-Carrier Simulation Results

```
*
*RELXPERT(TM) Version: 6.0.1
*FORMAT: PostBERT Output File
*DATE: 10-18-2004 15:56:33
=====================================================================================
<NMOS HCI Degradation Table>
  NMOS TRANSISTOR DEGRADATION RESULTS
  Transistors are listed in order of decreasing degradation.

Table Legends:
 maxIb    : maximum substrate current.
 avgIb    : average substrate current.
 maxIg    : maximum gate current.
 avgIg    : average gate current.
 Degrad.  : degradation after  10.00yrs.
 Ltime    : hot carrier lifetime (in yr) to reach  1.000e-01 Degradation.
 Age      : age used in aging simulation.

________________________________________________________________________________________
|Transistor      |maxIb(A)|avgIb(A)|maxIg(A)|avgIg(A)|Ltime(yr| Degrad.|  Age   |
|----------------|--------|--------|--------|--------|--------|--------|--------|
|        I10.M0  |1.90e-07|3.22e-09|0.00e+00|0.00e+00|5.04e+00|1.27e-01|2.64e-03|
|         I8.M0  |2.21e-07|3.30e-09|0.00e+00|0.00e+00|5.13e+00|1.26e-01|2.60e-03|
|         I0.M0  |2.49e-07|2.98e-09|0.00e+00|0.00e+00|5.30e+00|1.25e-01|2.51e-03|
|         I2.M0  |1.40e-07|3.32e-09|0.00e+00|0.00e+00|5.86e+00|1.20e-01|2.27e-03|
|        IL1.M0  |1.35e-07|4.48e-09|0.00e+00|0.00e+00|6.13e+00|1.19e-01|2.17e-03|
|        IL2.M0  |1.35e-07|4.48e-09|0.00e+00|0.00e+00|6.13e+00|1.19e-01|2.17e-03|
|        IL3.M0  |1.35e-07|4.48e-09|0.00e+00|0.00e+00|6.13e+00|1.19e-01|2.17e-03|
|        I12.M0  |2.39e-07|3.08e-09|0.00e+00|0.00e+00|6.30e+00|1.17e-01|2.11e-03|
|        I11.M0  |2.22e-07|2.81e-09|0.00e+00|0.00e+00|6.42e+00|1.17e-01|2.07e-03|
|         I3.M0  |1.85e-07|2.87e-09|0.00e+00|0.00e+00|6.63e+00|1.15e-01|2.01e-03|
|         I7.M0  |1.85e-07|2.81e-09|0.00e+00|0.00e+00|6.81e+00|1.14e-01|1.96e-03|
|         I5.M0  |1.58e-07|2.68e-09|0.00e+00|0.00e+00|7.41e+00|1.11e-01|1.80e-03|
|        IL6.M0  |1.47e-07|3.37e-09|0.00e+00|0.00e+00|7.79e+00|1.09e-01|1.71e-03|
|         I6.M0  |1.82e-07|3.02e-09|0.00e+00|0.00e+00|8.00e+00|1.08e-01|1.66e-03|
|         I9.M0  |2.05e-07|2.65e-09|0.00e+00|0.00e+00|8.23e+00|1.07e-01|1.62e-03|
|        IL4.M0  |1.42e-07|3.36e-09|0.00e+00|0.00e+00|8.86e+00|1.04e-01|1.50e-03|
|        IL5.M0  |1.42e-07|3.36e-09|0.00e+00|0.00e+00|8.86e+00|1.04e-01|1.50e-03|
|         I4.M0  |1.77e-07|2.85e-09|0.00e+00|0.00e+00|9.25e+00|1.03e-01|1.44e-03|
|         I1.M0  |1.72e-07|2.65e-09|0.00e+00|0.00e+00|9.56e+00|1.02e-01|1.39e-03|
|        IL7.M0  |1.47e-07|2.99e-09|0.00e+00|0.00e+00|9.92e+00|1.00e-01|1.34e-03|
________________________________________________________________________________________

<END>
```


© 2003-2014 All Rights Reserved.

### Table 13-7 PMOSFET Hot-Carrier Simulation Results

```
<PMOS HCI Degradation Table>
  PMOS TRANSISTOR DEGRADATION RESULTS
  Transistors are listed in order of decreasing degradation.

Table Legends:
 maxIb    : maximum substrate current.
 avgIb    : average substrate current.
 maxIg    : maximum gate current.
 avgIg    : average gate current.
 Degrad.  : degradation after  10.00yrs.
 Ltime    : hot carrier lifetime (in yr) to reach  1.000e-01 Degradation.
 Age      : age used in aging simulation.

--------------------------------------------------------------------------------
|Transistor      |maxIb(A)|avgIb(A)|maxIg(A)|avgIg(A)|Ltime(yr| Degrad.|  Age   |
|----------------|--------|--------|--------|--------|--------|--------|--------|
|           I3.M1|1.41e-09|2.62e-11|7.04e-13|8.83e-15|1.96e+00|1.36e-01|2.37e-05|
|          IL1.M1|1.30e-09|2.70e-11|7.55e-13|5.78e-15|2.09e+00|1.34e-01|2.22e-05|
|          IL2.M1|1.30e-09|2.70e-11|7.55e-13|5.78e-15|2.09e+00|1.34e-01|2.22e-05|
|          IL3.M1|1.30e-09|2.70e-11|7.55e-13|5.78e-15|2.09e+00|1.34e-01|2.22e-05|
|           I2.M1|1.40e-09|2.57e-11|7.69e-13|1.05e-14|2.09e+00|1.34e-01|2.22e-05|
|           I5.M1|1.54e-09|2.32e-11|8.29e-13|9.63e-15|2.35e+00|1.31e-01|1.98e-05|
|           I0.M1|2.17e-09|2.19e-11|1.09e-12|6.83e-15|2.44e+00|1.30e-01|1.91e-05|
|           I7.M1|1.72e-09|2.23e-11|9.22e-13|9.36e-15|2.47e+00|1.30e-01|1.88e-05|
|          IL4.M1|1.34e-09|2.27e-11|7.93e-13|5.39e-15|2.53e+00|1.29e-01|1.83e-05|
|          IL5.M1|1.34e-09|2.27e-11|7.93e-13|5.39e-15|2.53e+00|1.29e-01|1.83e-05|
|           I6.M1|1.29e-09|1.99e-11|8.83e-13|7.17e-15|2.65e+00|1.28e-01|1.75e-05|
|          I11.M1|1.86e-09|1.94e-11|9.90e-13|7.45e-15|2.83e+00|1.27e-01|1.64e-05|
|          I10.M1|1.77e-09|1.86e-11|9.33e-13|7.21e-15|2.96e+00|1.26e-01|1.57e-05|
|          IL7.M1|1.38e-09|1.95e-11|8.20e-13|4.32e-15|2.99e+00|1.25e-01|1.55e-05|
|          I12.M1|2.15e-09|1.70e-11|1.04e-12|4.66e-15|3.21e+00|1.24e-01|1.45e-05|
|           I9.M1|1.86e-09|1.71e-11|9.85e-13|4.54e-15|3.28e+00|1.23e-01|1.41e-05|
|           I8.M1|1.88e-09|1.58e-11|9.90e-13|6.37e-15|3.54e+00|1.22e-01|1.31e-05|
|          IL6.M1|1.40e-09|1.62e-11|8.26e-13|2.79e-15|3.61e+00|1.21e-01|1.28e-05|
|           I4.M1|1.64e-09|1.44e-11|9.29e-13|4.57e-15|4.08e+00|1.18e-01|1.14e-05|
|           I1.M1|1.58e-09|1.26e-11|8.40e-13|1.85e-15|4.60e+00|1.16e-01|1.01e-05|
--------------------------------------------------------------------------------

<END>
```


© 2003-2014 All Rights Reserved.

### Table 13-8 PMOSFET NBTI Simulation Results

```
<PMOS NBTI Degradation Table>
  PMOS TRANSISTOR DEGRADATION RESULTS
  Transistors are listed in order of decreasing degradation.

Table Legends:
 maxVg    : maximum absolute Vgs value.
 maxVd    : maximum absolute Vds value.
 Degrad.  : degradation after  10.00yrs.
 Ltime    : nbti/pbti lifetime (in yr) to reach  1.000e-01 Degradation.
 Age      : age used in aging simulation.

______________________________________________________________
|Transistor       |maxVg(V)|maxVd(V)|Ltime(yr| Degrad.|  Age   |
|-----------------|---------|---------|---------|---------|---------|
|        I12.M1   |2.60e+00|2.63e+00|5.77e-03|4.52e-01|1.97e-02|
|         I8.M1   |2.59e+00|2.60e+00|6.13e-03|4.46e-01|1.86e-02|
|         I3.M1   |2.56e+00|2.57e+00|6.27e-03|4.44e-01|1.81e-02|
|        I10.M1   |2.61e+00|2.59e+00|6.37e-03|4.43e-01|1.78e-02|
|         I0.M1   |2.63e+00|2.63e+00|6.39e-03|4.43e-01|1.78e-02|
|        I11.M1   |2.59e+00|2.60e+00|6.47e-03|4.42e-01|1.76e-02|
|         I9.M1   |2.60e+00|2.61e+00|6.74e-03|4.38e-01|1.69e-02|
|         I7.M1   |2.57e+00|2.59e+00|6.81e-03|4.37e-01|1.67e-02|
|         I6.M1   |2.57e+00|2.57e+00|6.83e-03|4.37e-01|1.67e-02|
|         I1.M1   |2.63e+00|2.58e+00|7.23e-03|4.32e-01|1.57e-02|
|         I5.M1   |2.59e+00|2.57e+00|7.36e-03|4.30e-01|1.55e-02|
|         I4.M1   |2.57e+00|2.59e+00|7.37e-03|4.30e-01|1.54e-02|
|        IL6.M1   |2.61e+00|2.56e+00|7.46e-03|4.29e-01|1.52e-02|
|        IL7.M1   |2.57e+00|2.55e+00|8.25e-03|4.20e-01|1.38e-02|
|         I2.M1   |2.58e+00|2.56e+00|8.70e-03|4.16e-01|1.31e-02|
|        IL4.M1   |2.59e+00|2.55e+00|8.73e-03|4.16e-01|1.30e-02|
|        IL5.M1   |2.59e+00|2.55e+00|8.73e-03|4.16e-01|1.30e-02|
|        IL1.M1   |2.58e+00|2.54e+00|1.01e-02|4.03e-01|1.12e-02|
|        IL2.M1   |2.58e+00|2.54e+00|1.01e-02|4.03e-01|1.12e-02|
|        IL3.M1   |2.58e+00|2.54e+00|1.01e-02|4.03e-01|1.12e-02|
______________________________________________________________

<END>
```


© 2003-2014 All Rights Reserved.

### Table 13-9 PMOSFET Hot-Carrier and NBTI Simulation Results

```
<PMOS HCI and NBTI Degradation Table>
  PMOS TRANSISTOR DEGRADATION RESULTS
  Transistors are listed in order of decreasing degradation.

Table Legends:
 maxIb    : maximum substrate current.
 avgIb    : average substrate current.
 maxIg    : maximum gate current.
 avgIg    : average gate current.
 Degrad.  : degradation after  10.00yrs.
 Ltime    : lifetime (in yr) to reach  1.000e-01 Degradation.

________________________________________________________________________
|Transistor      |maxIb(A)|avgIb(A)|maxIg(A)|avgIg(A)|Ltime(yr| Degrad.|
|----------------|--------|--------|--------|--------|--------|--------|
|           I3.M1|1.41e-09|2.62e-11|7.04e-13|8.83e-15|6.25e-03|5.80e-01|
|          I12.M1|2.15e-09|1.70e-11|1.04e-12|4.66e-15|5.76e-03|5.76e-01|
|           I0.M1|2.17e-09|2.19e-11|1.09e-12|6.83e-15|6.37e-03|5.73e-01|
|          I10.M1|1.77e-09|1.86e-11|9.33e-13|7.21e-15|6.36e-03|5.69e-01|
|          I11.M1|1.86e-09|1.94e-11|9.90e-13|7.45e-15|6.46e-03|5.68e-01|
|           I8.M1|1.88e-09|1.58e-11|9.90e-13|6.37e-15|6.11e-03|5.68e-01|
|           I7.M1|1.72e-09|2.23e-11|9.22e-13|9.36e-15|6.80e-03|5.67e-01|
|           I6.M1|1.29e-09|1.99e-11|8.83e-13|7.17e-15|6.81e-03|5.65e-01|
|           I5.M1|1.54e-09|2.32e-11|8.29e-13|9.63e-15|7.33e-03|5.61e-01|
|           I9.M1|1.86e-09|1.71e-11|9.85e-13|4.54e-15|6.72e-03|5.61e-01|
|          IL6.M1|1.40e-09|1.62e-11|8.26e-13|2.79e-15|7.45e-03|5.50e-01|
|           I2.M1|1.40e-09|2.57e-11|7.69e-13|1.05e-14|8.67e-03|5.50e-01|
|           I4.M1|1.64e-09|1.44e-11|9.29e-13|4.57e-15|7.36e-03|5.48e-01|
|           I1.M1|1.58e-09|1.26e-11|8.40e-13|1.85e-15|7.22e-03|5.47e-01|
|          IL7.M1|1.38e-09|1.95e-11|8.20e-13|4.32e-15|8.22e-03|5.46e-01|
|          IL4.M1|1.34e-09|2.27e-11|7.93e-13|5.39e-15|8.70e-03|5.45e-01|
|          IL5.M1|1.34e-09|2.27e-11|7.93e-13|5.39e-15|8.70e-03|5.45e-01|
|          IL1.M1|1.30e-09|2.70e-11|7.55e-13|5.78e-15|1.01e-02|5.37e-01|
|          IL2.M1|1.30e-09|2.70e-11|7.55e-13|5.78e-15|1.01e-02|5.37e-01|
|          IL3.M1|1.30e-09|2.70e-11|7.55e-13|5.78e-15|1.01e-02|5.37e-01|
________________________________________________________________________

<END>
```

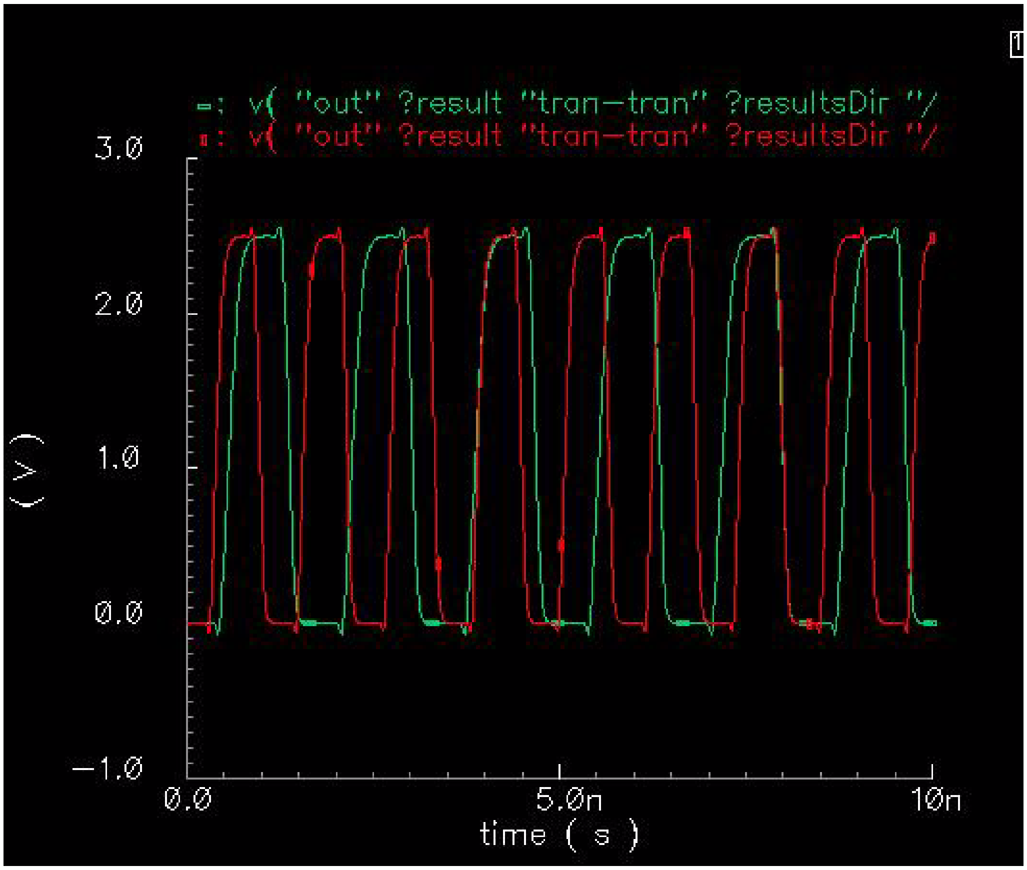
To perform the circuit aging simulation, type the following commands in batch mode:

```
relxpert_pre -age -sp ringos ringosAge.p2
spectre ringosAge.p2 > ringosAge.out
```

relxpert_pre -age (or prebert2) generates degraded model parameters for the circuit automatically because the file `ringos.ba0`, which contains the *Age* information, was created by relxpert_post (or postbert). SPICE simulation is performed on the aged circuit `ringosAge.p2`. Figure 13-1 on page 250 shows the difference between fresh and aged waveform.


© 2003-2014 All Rights Reserved.

**Figure 13-1 Oscillator Fresh and Aged Waveforms Output**





© 2003-2014 All Rights Reserved.

# Index

## Symbols



## A




© 2003-2014 All Rights Reserved.

## B



## C




© 2003-2014 All Rights Reserved.

# D



# E



# F



# G




© 2003-2014 All Rights Reserved.

## H



## I



## L



## M



## N




© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.

# T



# U



# V




© 2003-2014 All Rights Reserved.

## W



## X




© 2003-2014 All Rights Reserved.

© 2003-2014 All Rights Reserved.