The DECO framework: Reliability simulation based on a general design-environment communication approach

Sohrmann, Christoph; Lange, André; Jancke, Roland; Müller, Leif
Fraunhofer Institute for Integrated Circuits (IIS), Design Automation Division (EAS), Dresden, Germany

Abstract

Integrated circuit reliability is a major quality criterion in the semiconductor industry. Different physical mechanisms lead to transistor ageing, which depends on voltage conditions and temperature, and result in wear-out induced performance loss. To guarantee specified product life times already during the design phase, the impact of degradation needs to be analyzed in reliability simulations. First commercial tools carry out particular tasks in this subject, but they are not flexible. This paper presents our approach for an integrated reliability simulation framework. It combines ageing simulations and self-heating analyses and can easily be extended to cover further reliability concerns. Particular attention is paid to the communication between our tools and the design-environment. For this purpose, our XML-based DECO framework offers a flexible and easy-to-extend data transfer.

1 Motivation

Besides power consumption and process variability, integrated circuit reliability is a major concern in the semiconductor industry today. Early investigations of circuit ageing and variability during the design phase support the development of robust and reliable integrated circuits (IC).

This paper presents our approach for circuit reliability analyses. Considering reliability during the design phase becomes increasingly important. Especially the simultaneous impact of different effects becomes a problem for contemporary technologies, for instance interaction between self-heating and ageing. Moreover, there is much similarity in input and output data for different reliability analyses. However, most reliability tools are tailored for one particular reliability aspect. Simulation of multiple effects at the same time usually requires experience with reliability simulation and skillful handling of the design-environment (DE). We clearly see need for a more consistent approach to reliability analysis than currently available. One effort towards a standardized data interface API is the Open-Access (OA) initiative [1]. Unfortunately, this API is not entirely open. Moreover, not all DEs provide an OA data base. However, using the OA API as data source to our framework could be highly interesting as a future development.

In the following, we present an XML-based general-purpose interface we refer to as **DECO** (**DE**sign-environment **CO**mmunication) framework and how it can be used for reliability analysis. One of the main motivations behind using an abstraction layer is separation between data and logic, i.e. the tools based on the interface become independent from the data source, such that the reliability tool can be used across different DEs. There are further convincing arguments in favor of an XML layer for data exchange:

- Human-readable and editable format
- Interface is well-defined using an XML-schema
- Auto-generated parser and data-validation
- Class generators (XML-schema to object) are available for many languages, which keeps the interfaces consistent
- Logic code will be in a common language
- XML-schema straightforwardly converted into interface documentation
- Data easily accessed or converted into other formats using XSLT stylesheets
- Independent from particular DE

Besides the above mentioned advantages, we also would like to point to some drawbacks: Features of the DE may require reimplemention; the tools can only access data as provided by the interface; additional interfaces always introduce overhead, particularly when using XML. For some special application these aspects may not be acceptable. In view of a generic and more consistent approach to reliability analysis including an option to mixing different effects, the advantages outscore those drawbacks.

The remainder of this article is organized as follows: Section 2 presents the interface in detail. Our circuit analysis approach is described in Section 3. In Section 3.2, we introduce a model for NFET PBTI ageing which will be the test case of this paper. Application examples considering circuit ageing are presented in Section 4, and Section 5 concludes this paper.

2 DECO framework

2.1 Introduction

The DECO framework is an XML-based interface offering data structures for communication and data exchange between DE and an external tool. It has been designed particularly for reliability simulation. The use of XML has been motived in the introduction. Most importantly, this offers flexibility towards the tool's programming language and the choice of DE. Furthermore, the DECO frameworks aims at unifying reliability simulation throughout different domains. For instance, it allows to consider ageing and at the same time to take into account the thermal conditions due to self-heating by running a thermal simulation simultaneously.

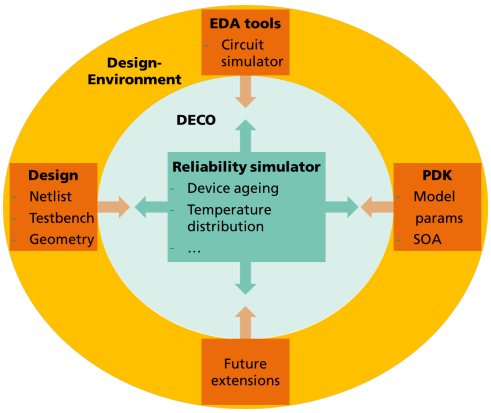

Fig. 1 The DECO framework as an abstraction layer between reliability analysis and DE.

Fig. 1 depicts how a reliability tool embeds in a DE. The DECO framework offers the interface for bidirectional communication.
In the following Sections 2.2-2.4 we present those interfaces in detail. The interfaces are shown in terms of class diagrams. They have been written in the **XSD** (**X**ML **S**chema **D**efinition) format. We recommend to use an XML editor such as Oxygen [2], which provides a graphical user interface for XSD editing. The final XSD description can be mapped into class objects using a mapping tool such as:

- CodeSynthesis XSD (C++),
- Xsd (C#),
- PyXB (Python),
- JAXB (Java) (used in this paper).

Besides the class mapping, these tools also provide parsing, validation and serialization functionality. XML files can be read and validated for interface compliance. Moreover, data stored in the class structures can be serialized into XSD-compliant XML files.
Access to the interface on the DE side is, however, less straightforward. Some DEs offer flexible scripting languages such as SKILL by Cadence [3]. If no such scripting language is available, an external tool for parsing results or writing netlists has to be created. Even with a scripting language, there is usually no XSD mapping tool and this step needs to be carried out manually. However, since this step is required only once, it could be made available publicly. In this work, we have used Cadence Virtuoso and SKILL, where we have created several functions for reading, writing, and processing XML data.

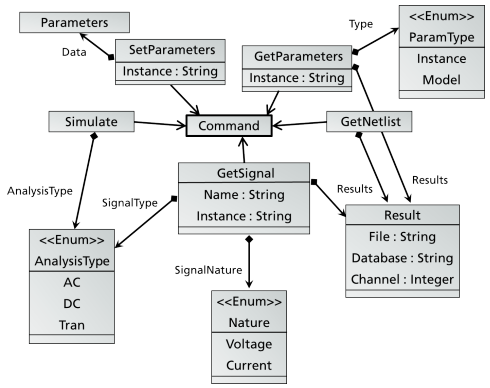

Fig. 2 UML class diagram for the **Command** object (tool → DE).

In the following, we use class diagrams to explain the interface required for ageing simulation. The diagrams are simple UML diagrams, using only the following structures: *Class*, *Generalization*, and *Composition* [4]. Fig. 2 shows the command interface for sending a command from a tool to the DE. We have implemented the commands **Simulate**, **GetSignal**, **GetNetlist**, **GetParameters**, and **SetParameters**. Corresponding attributes can be seen from the figure. Once the command has been sent, the DE will hand back the information requested as specified by the **Results** attribute, for instance as a file. Depending on the command, the answer will correspond to one of the objects explained below.

2.2 Interface to the Design

The most important interface to the design is the **GetNetlist** command. Upon execution, a netlist object will be returned as shown in Fig. 3. The structure allows a hierarchical representation of the netlist, since the class **Instance** is inherited from the class **Netlist** and thus contains terminals and instances itself.

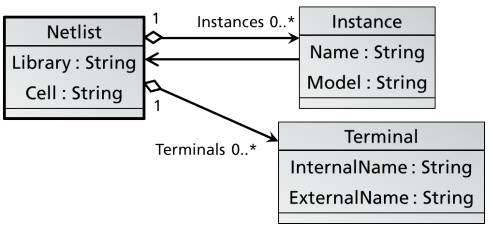

Fig. 3 UML class diagram for the **Netlist** object.

In order to access the parameters of a certain instance, the command **GetParameters** is used. It returns a **Parameters** object as shown in Fig. 4. The **Type** attribute indicates the parameter type requested and lists all parameters by name and value.

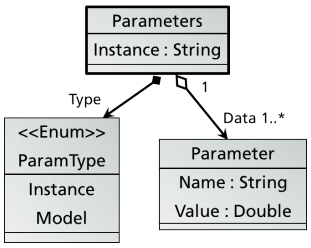

Fig. 4 UML class diagram of the **Parameters** object.

Setting a parameter is done using the command **SetParameters**. The values to be set will again be specified in the **Parameters** object and passed to the DE directly in the command request (cf. Fig. 2).
Other information which can be gathered from the design would be for instance layout information. Running a thermal simulation, layout data is essential. Working with layout data requires particular care, especially regarding mapping between netlist and layout instances. Details about this procedure exceed the scope of this work and will be given elsewhere.

2.3 Interface to the Circuit Simulator

After a simulation has been carried out (command **Simulate**), the simulator output can be accessed using the command **GetSignal**. Upon return, the **Signal** object carries information about the corresponding analysis type and data nature, as well as the data itself.

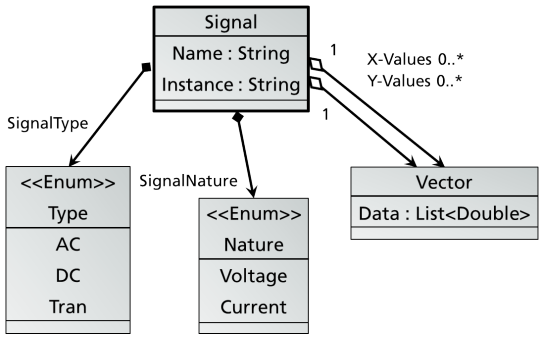

Fig. 5 UML class diagram for the **Signal** object.

The structure as shown in Fig. 5 is very basic and not suited for multidimensional signals resulting from complex parameter sweeps or Monte Carlo simulations. However, the interface is easily extended to feature such data.

2.4 Interface to the PDK

The PDK is the place to store technology-related information required for design work in circuit development. An example are device model parameters derived from hardware characterization. Besides nominal device parameters or statistical device models [5], it is likewise possible to hand over reliability related information in order to incorporate them into simulation models and reliability analysis tools. Therefore, the communication interface has been extended to cope with these technology related reliability data.
The most important reliability information are the parameters of the degradation models. A number of degradation models could be provided by the reliability simulator, but they require parameter sets provided by the PDK. For instance, the PBTI model presented in Section 3.2 has three parameters (A, m, and n) that are specific for each manufacturing process and each device type. Even geometry dependencies can be represented using either binning or additional parameters in the degradation models. Via the communication interface, the required models are selected and the respective parameters are supplied to be used in reliability analysis. Details of the interface description are beyond the scope of this work.
Other important reliability information are **SOA**s (**S**afe **O**perating **A**reas) [6]. They can originate either from measurements as carried out by a foundry or by analyzing the characterized degradation model as proposed in [7]. SOAs provide an indication to the designer about the expected lifetime of a device under different operating conditions. If such data is at all available from the foundries, it is provided in textual form within process specifications. To take into account this information when estimating circuit robustness and reliability in application environments, SOA data has to be available electronically. For this purpose, we allow the communication interface to include device-specific piecewise linear SOA borders. An example will be given in Section 4.2.

3 Reliability Simulation

3.1 Simulation Flow

Positive and Negative Bias Temperature Instability (PBTI and NBTI) as well as Hot Carrier Injection (HCI) are examples for physical effects that lead to a degradation of transistors over time and, hence, affect circuit performance. Modeling these effects has been a popular research topic. For instance, [8] proposes a two-stage model based on the statistics of hole trapping and interface-state generation to describe the permanent and recoverable parts of NBTI. This approach is, however, hard to apply in circuit simulations. To overcome this issue, electrical circuits combining diodes and resistors

[9] or distributed RC networks [10] are proposed. At least for device stress, PBTI and NBTI can be described similarly [11]. Examples for HCI models can be found in [12, 13]. It is important to note that transistor ageing strongly depends on device stress, i.e. temperature and bias conditions. This implies that the devices in a circuit degrade differently.

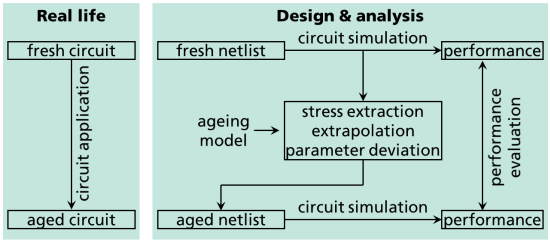

Fig. 6 Principle flow of ageing simulation

Instead of modeling physical ageing effects, the scope of this paper is on reliability analyses on circuit level. Commercial tools for ageing simulation are already available, see for instance [14]. They usually follow the flow depicted in Fig. 6 [15]: A nominal simulation with the fresh netlist determines nominal performance and device stress; device stress is evaluated and potentially extrapolated to the target age; parameters modeling the device ageing are calculated; ageing parameters are transferred back to create the aged netlist; and a circuit simulation with the aged netlist determines the circuit performance after degradation. Although such a tool fits well into the DE, it is not flexible due to fix ageing models and not transparent due to hidden degradation models. We have therefore developed an ageing simulation tool based on the DECO framework. The tool is written in Java and a screenshot is presented in Fig. 7.

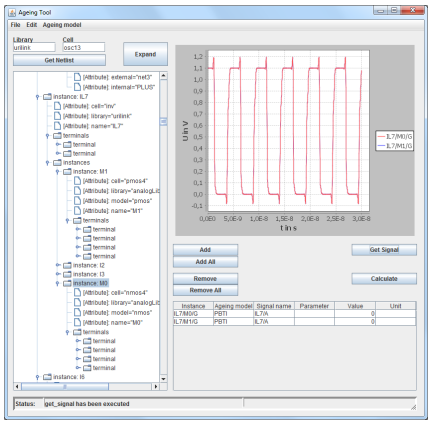

Fig. 7 Ageing simulator screenshot

The workflow starts by calling the tool directly from the DE (in this case Cadence Virtuoso). Once the tool has requested the netlist from the DE, the designer attaches ageing models to certain devices. After a nominal simulation has been run, the stress extraction is carried out by the tool's ageing models which return a set of degraded parameters. Finally, this set of parameters is written back to the DE where post-stress simulation is carried out.
The ageing simulator requires the implementation of an ageing model. An example will be given in the following section.

3.2 A PBTI Model

The ageing simulator will be demonstrated using an empirical PBTI model based on [11]. Neglecting recovery, PBTI can be modeled as a shift in threshold voltage $\mathrm{V_{th}}$ according to

$$\Delta V_{\mathrm{th}} = A \cdot t^{n} \cdot V_{\mathrm{gs}}^{m} \tag{1}$$

with the time t, the gate-source voltage $\mathrm{V_{gs}}$ of the NFET under consideration, and the model coefficients A, n, and m. The model should be understood as an example and can be replaced easily by a more physical one. We assume the units $[\mathrm{t}]=\mathrm{s}$, $[\mathrm{V_{gs}}]=\mathrm{V}$, $[\Delta\mathrm{V_{th}}]=\mathrm{mV}$, $[\mathrm{n}]=1$, $[\mathrm{m}]=1$, and $[\mathrm{A}]=\mathrm{V^{1-m}s^{-n}}$. From Fig. 2 in [11], for $\mathrm{T}=125°\mathrm{C}$ we obtain the values for the model parameters $\mathrm{n}=0.18$ and $\mathrm{m}=5.46$ as well as the data point $\mathrm{t}=1\mathrm{s}$, $\mathrm{V_{gs}}=1.4\mathrm{V}$, and $\Delta\mathrm{V_{th}}=10\mathrm{mV}$. Re-arranging (1) and inserting this data point yields $\mathrm{A}=1.6$.
Nevertheless, since (1) describes device degradation for constant stress ($\mathrm{v_{gs}(t)}=\mathrm{V_{gs}}$), it has to be adapted to circuit simulations with non-constant stress. Re-arranging (1) yields an expression for the time

$$t = A^{-1/n} \cdot \Delta V_{\mathrm{th}}^{1/n} \cdot V_{\mathrm{gs}}^{-m/n}. \tag{2}$$

Assuming constant stress, the time derivative of (1) can be determined, and (2) can be inserted,

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{dt}}\Delta V_{\mathrm{th}} &= n \cdot A \cdot V_{\mathrm{gs}}^{m} \cdot t^{n-1} \\ &= n \cdot A^{1/n} \cdot \Delta V_{\mathrm{th}}^{(n-1)/n} \cdot V_{\mathrm{gs}}^{m/n}. \end{aligned} \tag{3}$$

Separating the variables and integrating, we get

$$\Delta V_{\mathrm{th}} = A \cdot \left[\int v_{\mathrm{gs}}(t)^{m/n} \, \mathrm{dt} \right]^{n}, \tag{4}$$

allowing non-constant stress voltages $\mathrm{v_{gs}(t)}$. Threshold voltage shifts $\Delta\mathrm{V_{th}}$ can now be determined for arbitrary application scenarios using (4). However, there is a disproportion between simulation times in at most the ms regime and circuit ages of interest of days, months, or years. A solution is to use a typical circuit application case for a short time $\mathrm{t_{sim}}$ to determine device stress in a simulation and the $\mathrm{V_{th}}$ shift for this short time adding the integration boundaries 0 and $\mathrm{t_{sim}}$ to (4). The application scenario is assumed to be periodic until

the age of interest t_{age} is reached. Total device degradation can then be determined by the extrapolation

$$\Delta V_{th} = A \cdot \left[\frac{t_{age}}{t_{sim}} \cdot \int_0^{t_{sim}} v_{gs}(t)^{m/n} \, dt \right]^n . \quad (5)$$

To illustrate our ageing simulation approach, we consider PBTI stress using (5) with the determined model parameters $A = 1.6$, $n = 0.18$, and $m = 5.46$. NBTI, HCI, and recovery of PBTI are neglected in this paper for simplicity. We combine the simplified PBTI ageing model with 45 nm low-power predictive technology models [16].

4 Application Examples

4.1 7-Stage Ring Oscillator

One test vehicle for reliability simulations is the 7-stage ring oscillator depicted in Fig. 8. We assume the following typical application scenario at $Vdd = 1.1V$ and $T = 125°C$ in our analysis: a piece-wise linear voltage source drives the control signal *en*. It is at logic 0 for $0 \le t \le 10ps$ and at logic 1 for $11ps \le t \le 2ns = t_{sim}$.

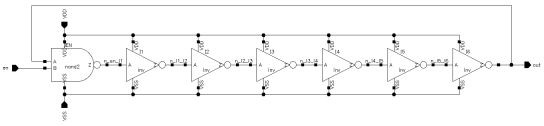

Fig. 8 Gate-level schematic of a 7-stage ring oscillator composed by a NAND2 gate and 6 inverters

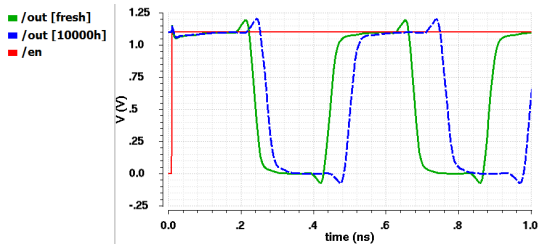

Fig. 9 Selected (truncated) oscillator signal waveforms at $t_{age} = 0$ and $t_{age} = 10000h$

In Fig. 9, the output signal *out* and the control signal *en* are shown as example waveforms of the simulation of the fresh oscillator. These signals are connected to the NAND inputs A and B and are, therefore, the stress signals of the corresponding NFETs. When we apply (5) to these waveforms, we obtain the shifts in threshold voltages for the devices as summarized in Tab. 1 for a set of target ages. Threshold voltage shifts for the inverter NFETs are also considered, but not listed here. Tab. 1 additionally shows the evolution of oscillation frequency due to device ageing. Starting at 2.278GHz, it is lowered to 96.2% after 24h, 92.6% after 1000h, and 88.8% after 10000h. This trend is illustrated by the oscillator output signal *out* for $t_{age} = 10000h$ in Fig. 9 (dotted line).

Table 1 Threshold voltage shifts for selected devices and frequency evolution over time; ND2/NA: NAND2 NFET connected to NAND2 input A; ND2/NB: NAND2 NFET connected to NAND2 input B

t_{age} [h]	ΔV_{th} [mV]		Frequency	
	ND2/NA	ND2/NB	abs. [GHz]	percentage
0	0	0	2.278	100
24	18.66	20.81	2.192	96.2
100	24.13	26.90	2.167	95.1
1000	36.52	40.72	2.110	92.6
10000	55.28	61.64	2.022	88.8

4.2 NFET Safe Operating Areas

In this paper, PBTI degradation is expressed as a shift in threshold voltage ΔV_{th} depending on stress voltage $v_{gs}(t)$ and time t. PBTI, however, is known to strongly depend on temperature T as well [8]–[10]. This accentuates the necessity to take into account device temperatures in reliability considerations and justifies our effort for developing an integrated reliability environment.
A simple approach to consider the temperature in the model (1) or (5) replaces the coefficient $A = 1.6$ by

$$A \longrightarrow A_0 \cdot \exp\left(-\frac{E_a}{kT}\right) \quad (6)$$

with the constant A_0, the activation energy E_a, and the Boltzmann constant $k = 8.617 \cdot 10^{-5} eV \cdot K^{-1}$ (cf. permanent NBTI degradation in [10]). Due to the lack of temperature-dependent measurement results in [11], we assume $E_a = 0.10eV$ following [10]. From (6) and the data in Section 3.2, we derive $A_0 \approx 29.5$.

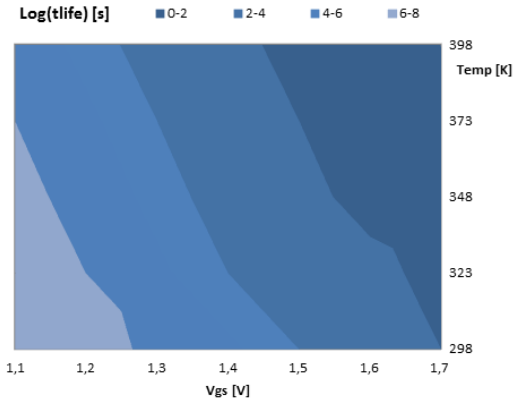

Fig. 10 SOA diagram for the used PBTI model, color-coded lifetimes for constant stress conditions

Inserting (6) into (1) and defining an end-of-life criterion, say $\Delta V_{th} = 30mV$, yields an expression that relates usable device lifetimes and operating conditions, constant stress voltages V_{gs} and temperatures T. In Fig. 10, this model is evaluated for a temperature range $25°C \le T \le 125°C$ and a voltage range $1.1V \le V_{gs} \le 1.7V$. The lifetime is given on a logarithmic scale in seconds, so 10^8s means roughly 3 years of permanent operation.

A variety of SOA-based methods for circuit analysis, optimization, and verification are possible. An example is the reliability-driven operating point selection in analog circuits. First approaches are already implemented and included in the degradation analysis environment.

5 Summary & Conclusions

The purpose of this paper was to present a flexible approach to reliability analysis. It is based on the **DECO** (**DE**sign-environment **CO**mmunication) framework, an XML-based abstraction interface which enables the separation of the data source (the design-environment) and the logic (the reliability tool). Moreover, this approach has the following advantages:

- Independence from the DE
- Well-defined interfaces using XML-schemas
- Class generation for different languages
- Auto-generated parser and data-validation

We have explained in detail the interfaces used for our in-house ageing simulator. Depending on the concrete application, the interfaces might need to be extended, which is easily carried out as described in this work. Finally, we have implemented an ageing simulator using the DECO framework and demonstrated the approach on a 7-stage ring oscillator with a PBTI model.
A unified interface for reliability simulation easily allows to incorporate multiple reliability effects into one analysis simultaneously. Currently, the coupling of ageing and thermal analysis is implemented and will be presented soon. Furthermore, we are planning to implement variability, electromigration, and substrate coupling analysis on top of the DECO framework.

6 Acknowledgement

This work has been supported by the German ministry of education and research (BMBF) within the project "Design for Reliability of SoCs for Applications like Transportation, Medical, and Industrial Automation" (RELY), 16M3091C.

7 References

[1] Si2 - Silicon Integration Initiative, URL http://www.si2.org/

[2] Oxygen XML Editor, URL http://www.oxygenxml.com/

[3] Barnes, T. J.: "SKILL: a CAD system extension language", 27th ACM/IEEE Design Automation Conference, 1990. Proceedings; pp. 266–271.

[4] Rumbaugh, J.; Jacobson, I.; Booch, G.: "The Unified Modeling Language Reference Manual", Addison-Wesley, 1998.

[5] Lange, A. et al.: "A General Approach for Multivariate Statistical MOSFET Compact Modeling Preserving Correlations", *Proceedings of the 2011 European Solid-State Device Research Conference (ESSDERC)*, pp. 163–166, 12–16 Sept. 2011.

[6] Sobe, U. et al.: "Robust Analog Design for Automotive Applications by Design Centering With Safe Operating Areas," *IEEE Transactions on Semiconductor Manufacturing*, Vol. 22, No. 2, pp. 217–224, May 2009

[7] Jancke, R.; Ellmers, C.; Gaertner, R.: "Design of reliable circuits by determination of SOA borders as part of the degradation analysis", *Semiconductor Conference SCD 2009*. Martinsried: Gerotron Communication, 2009

[8] Grasser, T. et al.: "A Two-Stage Model for Negative Bias Temperature Instability", *2009 IEEE International Reliability Physics Symposium (IRPS)*, pp. 33–44, 26–30 Apr. 2009.

[9] Reisinger, H. et al.: "The statistical analysis of individual defects constituting NBTI and its implications for modeling DC- and AC-stress", *2010 IEEE International Reliability Physics Symposium (IRPS)*, pp. 7–15, 2–6 May 2010.

[10] Maricau, E. et al.: "A Compact NBTI Model for Accurate Analog Integrated Circuit Reliability Simulation", *Proceedings of the 2011 European Solid-State Device Research Conference (ESSDERC)*, pp. 147–150, 12–16 Sept. 2011.

[11] Kerber, A.; Krishnan, S.A.; Cartier, E.A.: "Voltage Ramp Stress for Bias Temperature Instability Testing of Metal-Gate/High-k Stacks", *IEEE Electron Device Letters*, Vol. 30, No. 12, pp. 1347–1349, Dec. 2009.

[12] Hu, C. et al.: "Hot-Electron-Induced MOSFET Degradation – Model, Monitor, and Improvement", *IEEE Journal of Solid-State Circuits*, Vol. 20, No. 1, pp. 295–305, Feb. 1985.

[13] Leblebici, Y. and Kang, S.-M.: "Modeling of nMOS transistors for simulation of hot-carrier-induced device and circuit degradation", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 11, No. 2, pp. 235–246, Feb. 1992.

[14] Cadence Design Systems, Inc.: "Reliability Simulation in Integrated Circuit Design", White Paper, http://w2.cadence.com/whitepapers/5082_ReliabilitySim_FNL_WP.pdf

[15] Kole, M.: "Reliability Simulation based on Verilog-A", *2007 IEEE International Behavioral Modeling and Simulation Conference*, pp. 58–63, 20–21 Sept. 2007.

[16] Arizona State University, Nanoscale Integration and Modeling Group: "Predictive Technology Models – 45 nm PTM LP model, V2.1", URL ptm.asu.edu/modelcard/LP/45nm_LP.pm