Statistical and Numerical Approach for a computer efficient circuit yield analysis

Lucas Brusamarello[1], Roberto da Silva[1], Gilson I. Wirth[2], and Ricardo Reis[1]

[1] UFRGS - Universidade Federal do Rio Grande do Sul - Instituto de Informática
Av. Bento Gonçalves, 9500. CEP 91501-970. Porto Alegre, Brazil
[2] UFRGS - Universidade Federal do Rio Grande do Sul - Departamento de Engenharia Elétrica
Av. Osvaldo Aranha, 103. CEP : 90035-190, Porto Alegre, Brazil

Abstract. In nanometer scale CMOS parameter variations are a challenge for the design of high yield integrated circuits. Statistical Timing Analysis techniques require statistical modeling of logic blocks in the netlist in order to compute mean and standard deviate for system performance. In this work we propose an accurate and computer efficient methodology for statistical modeling of circuit blocks. Numerical error propagation techniques are applied to model within-die and die-to-die process variations at electrical level. The model handles co-variances between parameters and spatial correlation, and gives as output the statistical parameters that can be applied at higher level analysis tools, as for instance statistical timing analysis tools. Moreover, we develop a methodology to compute the quantitative contribution of each circuit random parameter to the circuit performance variance. This methodology can be employed by the designer or by an automatic tool in order to improve circuit yield.
The methodology for yield analysis proposed in this work is shown to be a solid alternative to traditional Monte Carlo analysis, reducing by orders of magnitude the number of electrical simulations required to analyze memory cells, logic gates and small combinational blocks at electrical level. As a case study, we model the yield loss of a SRAM memory due to variability in access time, considering variance in threshold voltage, channel width and length, which may present both die-to-die and within-die variations. We compare results obtained using the proposed method with statistical results obtained by Monte Carlo simulation. A speedup of $1000\times$ is achieved, with mean error of the standard deviate being 7% compared to MC.

1 Introduction

Performance and reliability of deep-sub-micron technologies are being increasingly affected by process variations and leakage current [24]. Variability in the manufacturing process imposes limitations to the design of circuits in recent technologies. Process variations are related to machinery limited precision and process methodology variations like temperature and lithography exposure time, and discreteness of the material. These variations are stochastic and the prediction of the percentage of manufactured circuits which will achieve a given performance becomes a major problem for the circuit designer. Therefore, the use of statistical methods in circuit design is of increasing relevance.

Electrical parameter variability may be decomposed into die-to-die variations (D2D) and within-die variations (WD) [27]. Within-die variations may arise from different sources, for instance the discreteness of matter and energy (dopant atoms, photo resist molecules, and photons). A well known example of a WD parameter is threshold voltage (VtMahmoodi2005Estimation-of-d. Random Dopant Fluctuations (RDF) is mainly caused by the irregular distribution of doping atoms in the channel, and this effect nowadays represents one of the greatest challenges for the industry [10]. Consider σ_{vt0} the standard deviation in threshold voltage for minimum sized transistors, then the dependence of σ_{vt} on transistor size is given by [25]:

$$\sigma_{vt} = \sigma_{vt0}\sqrt{\frac{L_{min} \times W_{min}}{L \times W}} \tag{1}$$

Die-to-die variations may arise from equipment asymmetries (like asymmetries in chamber gas flows, thermal gradients and so on) or imperfections in equipment operation and process flow. These asymmetries and imperfections affect the average value of a parameter from die to die, wafer to wafer, and lot to lot. Variations may also be originated by the pattern or layout induced deviation of a parameter from its nominal value [6]. Parameters such as oxide thickness, transistor channel length and channel width may show systematic variations [12]. In the case of a D2D parameter k, transistors close to each other are affected by the same constant fluctuation δk.

Statistical Static Timing Analysis (SSTA) gives at logic level a quantitative risk management for the design as a function of the circuit topology, the electrical parameters and the variations [26]. In order to apply a SSTA methodology, the cell libraries are characterized at electrical level, for which nowadays Monte Carlo simulation is commonly employed. Larger designs, composed by many hundreds of transistors, may be decomposed in functional blocks and treated at different levels of abstraction. A block may be a simple or complex gate, a sequential block (e.g. flip-flop) or a memory cell. At the block level the variability may be evaluated using the methodology proposed in this manuscript. The result provided by this methodology (mean, standard deviation) may then be used by higher abstraction level techniques, as for instance Statistical Static Timing Analysis (SSTA), to provide risk management at this higher abstraction levels.

In [12] cell characterization using numerical error propagation is proposed. However, their cell modeling methodology does not include D2D variation, although their proposed SSTA algorithm considers spatial correlation at gate level. Furthermore, the quantitative contribution of each random parameter to the circuit performance variance is not analyzed. As shown further in our work, this analysis may help to improve yield. Finally, in that work only first order approximation for numerical derivatives is employed, and the algorithm complexity and accuracy are not analyzed. In our work we show that the model accuracy may be very sensitive to numerical derivative approximation. Higher order approximations may lead to a better accuracy.

Yield analysis of SRAM memories using Monte Carlo has been studied in [2], [3] and [4]. Error propagation at electrical level for yield analysis of SRAM memory has been explored in [14] and [15], but Vth is the only random variable analyzed (Monte Carlo is performed to simulate D2D). Sensitivities are computed using first order numerical approximation. In these works failures in SRAM cell are statistically modeled

(access time failure, read failure, write failure and hold failure), and yield of SRAM memory is given as a function of redundant columns employed in the design. Simulations in these works show that the most significant source of failures in SRAM cell is access time failure. In [9] an electrical-level analysis of SRAM cell static noise margin is presented, which is based in the extraction of the electric parameters by an atomistic device simulator. This method is robust because the transistor cards are the most consistent and closely related to the device variations, but still a huge number of device and electric-level simulations must be run (200 runs in that case).

This manuscript presents a general methodology for analysis of circuit blocks at electrical level which is able to consider WD and D2D variations, as well as co-variances between electrical parameters. Also, we implement a method to point out the parameters which most contribute to circuit performance variance. The methodology is general because it is independent of circuit topology (SRAM cell, multiplexer block, complex gate, etc), and circuit performance parameter of interest (delay, leakage current, power, etc). It maintains the generality of the traditional Monte Carlo techniques, still largely employed in commercial electrical simulators [23].

As a case study we discuss yield analysis and optimization of a SRAM memory. SRAM memory is a good case study because memory yield is directly dependent on SRAM cell yield, and SSTA is not required to analyze critical paths and signal correlations. For this case study we develop a methodology of yield improvement based on the analysis of parameter contribution to variability. We resize the transistors that present the major contribution to the access time variance.

The paper is organized as follows. Section 2 presents a high-level introduction to the methodology. Sections 3 and 4 formally define the problem of statistical analysis of integrated circuits at electrical level and describe the mathematical foundations of the proposed methodology. Section 5 exposes a formulation for the sensitivity of the variance to the electrical parameters. Section 6 gives formulations for numerical computation of derivatives using higher order approximations. Section 7 details the proposed algorithm, and presents a study on its complexity. Section 8 focuses on the methodology applied to yield analysis and yield maximization of a SRAM memory, considering variability in the SRAM cell access time. Finally, last section presents our conclusions.

2 Methodology

This work describes a framework to compute variability in circuit electrical behavior and its dependency on the design and process parameters. The methodology is based on the computation of variance using error propagation, where derivatives are numerically computed using electrical simulations – in this work HSPICE[23] is employed.

Both circuit netlist and the set of circuit parameters which are modeled as random variables are user inputs. Each random variable has its mean and standard deviation, as well as its kind (WD or D2D).

A first script generates a set of runs for the electrical simulator – in our the case HSPICE .DATA command [23]. This library is included in the netlist file, and HSPICE is run in SWEEP mode.

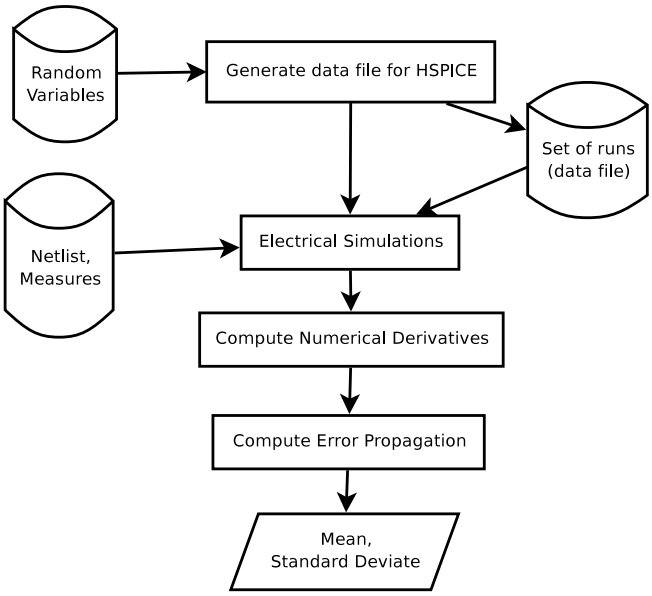

Fig. 1. High level flowchart

One value (circuit response) is computed at each run. These values are gathered in order to compute the partial derivatives for each electrical parameter. Finally, error propagation is employed to compute the variance. An approximation for the mean is obtained by simulation using nominal values.

3 Model

Consider an electric circuit denoted by ω, composed of n transistors represented as components of the vector $\overrightarrow{\tau} = (\tau_1, \ldots, \tau_n)$, interconnected according to a topology Γ. By definition, the circuit response is given by the function $F(\overrightarrow{\alpha}_1, \ldots, \overrightarrow{\alpha}_n, \overrightarrow{\beta}_1, \ldots, \overrightarrow{\beta}_n, \omega)$, where the vectors $\overrightarrow{\alpha}_i = (\alpha_i^{(1)}, ..., \alpha_i^{(p)})$ and $\overrightarrow{\beta}_i = (\beta_i^{(1)}, ..., \beta_i^{(q)})$ represent respectively the WD and D2D parameters of transistor i, p is the number of WD parameters and q the number of D2D parameters. For instance, the case $\overrightarrow{\alpha}_3 = (V_t)$ and $\overrightarrow{\beta}_3 = (T_{ox}, L, W)$ represents typical input parameters for transistor τ_3, including oxide thickness (T_{ox}), threshold voltage (Vt) and dimensions (L and W) of the transistor.

In the presence of variability in the fabrication process, electrical characteristics and physical dimensions of the circuit can be considered random variables and consequently the output is a random variable. Consider, without loss of generality, that parameters (as for instance T_{ox}, Vt, L, W) are Gaussian variables with mean (μ) and variance (σ^2), i.e, $\alpha_i^{k_1} = N\left(\mu(\alpha_i^{k_1}), \sigma^2(\alpha^{k_1})\right)$ and $\beta_i^{k_2} = N\left(\mu(\beta_i^{k_2}), \sigma^2(\beta^{k_2})\right)$, where $i = 1, ..., n$, $k_1 = 1, ..., p$ and $k_2 = 1, ..., q$.

The circuit statistical response S is a function that depends on $N = n \times (p+q)$ random variables (including WD and D2D parameters), given by the functional relation

$$S = F(\overrightarrow{\alpha}_1, \ldots, \overrightarrow{\alpha}_n, \overrightarrow{\beta}_1, \ldots, \overrightarrow{\beta}_n, \omega) \tag{2}$$

3.1 D2D and WD random variables

In order to model the impact of process variations on the electric circuit response, D2D and WD are treated differently. In the case of a D2D parameter, the same fluctuation affects transistors close to each other. Still their absolute values may be different because they may have distinct averages.

Other random variables are modeled as Gaussian random variables, which are denoted in this work as WD parameters. A WD variable assumes a random value for each transistor, although it can be subject to covariance coefficients (σ_{ij}).

Notice that both D2D and WD parameters are random variables. The difference between them is the randomness context: each instance of a WD variable assumes a different random value, while a D2D parameter has a single random fluctuation that applies to a set of devices.

D2D parameters Spatial correlation impels the D2D electrical parameter of all transistors to change in a synchronized way. For instance, if the dimension W is assumed to present D2D variations and W_1 of transistor τ_1 changes by a quantity δW, the dimension W_2 of a transistor τ_2 changes by the same quantity δW although their mean ($\mu(W_1)$ and $\mu(W_2)$) in the standard sampling process can be different. The parameter W is then defined as a variable that presents

1. exactly the same variation δW inside an single electrical block;
2. but different variation in different electrical blocks, as for instance variation δW_1 in block 1 and variation δW_2 in block 2.

The reader should notice that the position (x,y) of a device is not taken into account. Parameters that present D2D variations can be modeled as

$$\beta_i^j = \mu(\beta_i^j) + \xi_j \cdot \sigma(\beta^j)$$

where $\xi_j = N(0,1)$ is a standard normal variable which is independent of the transistor $1 \leq i \leq n$. It means that the same variable j will have the same shift of magnitude $\xi_j \cdot \sigma(\beta^j)$ independent of the transistor to which it is applied. In other words, the variables $\beta_1^j, ..., \beta_n^j$ are the same random variable except by their mean values. Looking at the contribution of this variables for error estimation, it is important to define the general variable $\beta^j = \mu(\beta^j) + \xi_j \cdot \sigma(\beta^j)$, where $\mu(\beta^j)$ is a transistor-independent constant. Then it can be written as

$$\beta_i^j(k) = \mu(\beta_i^j) + \xi_j \cdot \sigma(\beta^j) = \mu(\beta_i^j) + \beta^j - \mu(\beta^j) \tag{3}$$

Which leads to suitable simplification
$F(\overrightarrow{\alpha}_1,\ldots,\overrightarrow{\alpha}_n,\overrightarrow{\beta}_1,\ldots,\overrightarrow{\beta}_n,\omega) = F(\overrightarrow{\alpha}_1,\ldots,\overrightarrow{\alpha}_n,\beta^1,\ldots,\beta^q,\omega)$ and using the chain rule the computation of partial derivatives becomes

$$\frac{\partial F}{\partial \beta^j} = \sum_{i=1}^{n} \frac{\partial F}{\partial \beta_i^j}\frac{\partial \beta_i^j}{\partial \beta^j} \tag{4}$$

$$= \sum_{i=1}^{n} \frac{\partial F}{\partial \beta_i^j} \tag{5}$$

because according to equation 3 it is true that $\partial \beta_i^j / \partial \beta^j = 1$, for all $i \in \{1,...,n\}$.

4 Error propagation and Monte Carlo

When measuring a quantity denoted by f which depends of n variables, x_1, x_2, ..., x_n, an important point is to determine the uncertainty in f given the uncertainty in each variable. A general formula is known if we suppose that $\{x_i\}_{i=1}^n$ are random Gaussian variables, which is a widely accepted procedure [6]. In this case the uncertainty in f (this is an error estimate, including systematic and statistical sources) is given by the classical error propagation formula [21]:

$$\sigma_f^2 = \sum_{i=1}^{n} \left(\left.\frac{\partial f}{\partial x_i}\right|_{x_i=\overline{x}_i} \right)^2 \sigma_{x_i}^2 + 2\sum_{i=1}^{n}\sum_{j=i}^{n} \left(\left.\frac{\partial f}{\partial x_i}\right|_{x_i=\overline{x}_i} \left.\frac{\partial f}{\partial x_j}\right|_{x_j=\overline{x}_j} \right) \sigma_{x_i,x_j} \tag{6}$$

where $\sigma_{x_i}^2$ is the variance (error estimate of variable x_i) while σ_{x_i,x_j} is the covariance between variables x_i and x_j.

For highly non-linear parameters the methodology may lead to significant errors in the yield estimation. However, for most of the practical situations, the parameter distributions are expected not to be highly non-linear. The study of the shape of the actual distribution (and linearity) of the parameters is a topic of intense research, and general models are not yet available. It is out of the scope of this work to provide such models. What can be said is that if the parameters are not highly non linear, the proposed methodology is expected to provide an appropriate yield estimation methodology.

The general error propagation formula (equation 6) applied to the model for WD and D2D variations in an electric circuit, considering co-variances, is:

$$\sigma_S^2 = \sum_{i=1}^{n}\sum_{j=1}^{p}\left(\left.\frac{\partial F}{\partial \alpha_i^j}\right|_{\alpha_i^j=\mu(\alpha_i^j)}\right)^2 \sigma^2(\alpha^j) + \sum_{j=1}^{q}\left(\sum_{i=1}^{n}\left.\frac{\partial F}{\partial \beta_i^j}\right|_{\beta_i^j=\mu(\beta_i^j)}\right)^2 \sigma^2(\beta^j)$$
$$+ 2\sum_{i=1}^{n}\sum_{j=1}^{p}\sum_{k=j}^{p}\left(\left.\frac{\partial F}{\partial \alpha_i^j}\right|_{\alpha_i^j=\mu(\alpha_i^j)}\left.\frac{\partial F}{\partial \alpha_i^k}\right|_{\alpha_i^k=\mu(\alpha_i^k)}\right)\sigma(\alpha^j,\alpha^k)$$
$$+ 2\sum_{i=1}^{n}\sum_{j=1}^{q}\sum_{k=j}^{q}\left(\left.\frac{\partial F}{\partial \beta_i^j}\right|_{\beta_i^j=\mu(\beta_i^j)}\left.\frac{\partial F}{\partial \beta_i^k}\right|_{\beta_i^k=\mu(\beta_i^k)}\right)\sigma(\beta^j,\beta^k)$$
$$+ 2\sum_{i=1}^{n}\sum_{j=1}^{p}\sum_{k=1}^{q}\left(\left.\frac{\partial F}{\partial \alpha_i^j}\right|_{\alpha_i^j=\mu(\alpha_i^j)}\left.\frac{\partial F}{\partial \beta_i^k}\right|_{\beta_i^k=\mu(\beta_i^k)}\right)\sigma(\alpha^j,\beta^k) \quad (7)$$

The reader should notice that covariances between electrical parameters do not imply in any overhead in the number of simulations.

The non-biased sampling estimator to the standard deviation computed from a sample of n_{sample} experimental measures of S, denoted as $S_1, S_2, ..., S_{nsample}$, is calculated by the expression

$$\delta_S = \sqrt{\frac{1}{(n_{sample}-1)}\sum_{i=0}^{n_{sample}}(S_i - \langle S_i\rangle)^2}$$

must be numerically equal to σ_S for a n_{sample} sufficiently large, i.e.,

$$\delta_S \approx \sigma_S$$

Monte Carlo simulation [5] is often employed in order to obtain the probability density function (PDF) of some circuit output (delay, power consumption, leakage current, ...). Usually, a run with a large number of samples n_{sample} is generated, aiming the convergence of the standard deviation. However, the error in a Monte Carlo simulation is hardly reduced, once it is $O(1/\sqrt{n_{sample}})$.

The inputs in the error propagation formulation are 1) the partial derivatives of the circuit response to the random parameters; 2) standard deviation of the random parameters; and 3) the correlation between random parameters. Standard deviations and correlation coefficients are technology dependent and are given by the foundry. According to what will be shown in section 6, as $F(k_1,\ldots,k_N)$ is an arbitrary function that can be computed by electrical simulation, the numerical estimates for derivatives $\left.\frac{\partial F}{\partial k_i}\right|_{k_i=\overline{k}_i}$ also can be computed by electrical simulation.

5 Sensitivity of the circuit variability to the electrical parameters

When dealing with the challenges imposed by design for manufacturability, it is essential to have a methodology capable of identifying which parameters contribute most to

the circuit variability. Once error propagation decomposes the circuit response variance into its components, it can be used to point out which devices of the circuit could be re-designed in order to optimize yield.

Error propagation uncovers the quantitative contribution of each transistor to the variability in circuit performance. Revisiting equation 7, the sensitivity of the circuit response variance to a within-die parameter α^k is given by

$$K(\alpha^k) = \left(\frac{\partial F}{\partial \alpha^k}\right)^2 \sigma^2_{\alpha^k}. \tag{8}$$

For D2D components, a re-weighted function can be defined as

$$p_{ik} = \left(\frac{\left|\partial F/\partial \beta_i^k\right|}{\sum_{j=1}^{m} \left|\partial F/\partial \beta_j^k\right|} \right) \tag{9}$$

where $\sum_{i=1}^{m} p_{ik} = 1$ for m synchronized variables. For a parameter β_i^k that presents D2D variation the sensitivity is given by

$$K(\beta_i^k) = p_{ik} \times \left(\frac{\partial F}{\partial \beta^k}^2 \sigma^2_{\beta^k} \right) \tag{10}$$

6 Numerical estimate of partial derivatives

Numerical approximations of derivatives is applied in order to present a generic methodology independent of circuit topology. Linear approximations using 1, 2 and 4 points around the nominal values are exploited, aiming to obtain the sensitivity of circuit response for the random variables. The difference between these formulas is the accuracy in the numerical estimates and the number of electric simulations needed: higher order approximations require more simulations, but are more accurate.

Problem Formulation: *Consider a general function of n variables* $f = f(x_1, x_2, \ldots, x_n)$, *such that numerical values for the variables are* $x_1 = \overline{x_1}, \ldots, x_n = \overline{x_n}$. *By error propagation we have* $\sigma_f^2 = (\partial f/\partial x_1)^2_{x_1=\overline{x}_1} \sigma^2_{x_1} + \ldots + (\partial f/\partial x_n)^2_{x_n=\overline{x}_n} \sigma^2_{x_n}$. *Find a numerical approximation for* $\partial f/\partial x_i$ $(i = 1, \ldots, n)$.

6.1 1st Order Approximation

Expanding the n-dimensional Taylor series around point $f(\overline{x_1}, \ldots, \overline{x_i}, \ldots, \overline{x_n})$ up to order 2 we obtain:

$$f(\overline{x_1}, \ldots, \overline{x_i} + \varepsilon, \ldots, \overline{x_n}) = f(\overline{x_1}, \ldots, \overline{x_i}, \ldots, \overline{x_n}) + \varepsilon \frac{\partial f(\overline{x_1}, \ldots, \overline{x_i}, \ldots, \overline{x_n})}{\partial x_i} + O(\varepsilon^2) \tag{11}$$

The numerical value of $f(x_1, \ldots, x_n)$ is computed by electrical simulation. Thus, one can calculate the sensitivity at point $f(\overline{x_1}, \ldots, \overline{x_i} + \varepsilon, \ldots, \overline{x_n})$, rewriting 11 and assuming $\varepsilon \ll 1$ as follows

$$\frac{\partial f}{\partial x_i}(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) = \frac{f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n}) - f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n})}{\varepsilon} + O(\varepsilon) \tag{12}$$

Complexity of 1st order approximation: For this case 2 electrical simulations are required to compute each partial derivative: one is required to compute $f(\overline{x_1},..,\overline{x_i}+\varepsilon,...,\overline{x_n})$ and another one for $f(\overline{x_1},..,\overline{x_i},...\overline{x_n})$. However, as $f(\overline{x_1},..,\overline{x_i},...\overline{x_n})$ is the same for all partial derivatives, it needs to be computed only once. Thus, computation of all partial derivatives using first order approximation requires $n+1$ runs.

6.2 2nd Order Approximation

In order to obtain a more precise approximation, algebraic manipulations over Taylor expansion results in a formula with accuracy $O(\varepsilon^2)$. Consider Taylor expansions around the points $f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n})$ and $f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n})$, and a better approximation for $\frac{\partial}{\partial x_i} f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n})$ can be computed according to:

$$\frac{\partial}{\partial x_i} f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) = \frac{f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n}) - f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n})}{2\varepsilon} + O(\varepsilon^2) \tag{13}$$

Complexity of 2nd order approximation: this formulation requires 2 electrical simulations for each variable of interest: one to evaluate $f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n})$ and another one to evaluate $f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n})$. Therefore, to calculate n partial derivatives over all the variables – 2nd order approximation requires $2n$ runs.

6.3 4th Order Approximation

An $O(\varepsilon^4)$ approximation can be obtained for the numerical estimate of the derivative, as in (please refer to Appendix A for detailed algebraic manipulations) :

$$\begin{aligned}\frac{\partial f}{\partial x_i}(\overline{x_1},...,\overline{x_n}) = {} & \frac{1}{3}\cdot\frac{[-f(\overline{x_1},...,\overline{x_i}+2\varepsilon,...,\overline{x_n}) + f(\overline{x_1},..,\overline{x_i}-2\varepsilon,...,\overline{x_n})]}{4\varepsilon} \\ & + \frac{4}{3}\cdot\frac{f(\overline{x_1},..,\overline{x_i}+\varepsilon,...,\overline{x_n}) - f(\overline{x_1},..,\overline{x_i}-\varepsilon,...,\overline{x_n})}{2\varepsilon} + O(\varepsilon^4)\end{aligned} \tag{14}$$

Complexity of 4th order approximation: for each variable 4 electrical simulations must be run. Hence, an $O(\varepsilon^4)$ approximation requires $4n$ electrical simulations.

7 Algorithm

Algorithm 1 presents the general methodology developed along the last section. The numerical method for derivatives is the one which gives and error of $O(\varepsilon^2)$. Notice that algorithm for other approximations are very similar – in fact the unique difference would be the formula for derivative computation (l. 5-6 and 12-13).

Consider a circuit net-list ω which has a vector of transistors $\overrightarrow{\tau}$ connected according to the specified topology. The circuit response F is specified in the net-list of ω, for instance it can be a DC or a transient analysis. The vector of random variations $\overrightarrow{\alpha}$ and vector of systematic variations $\overrightarrow{\beta}$ are related to the model of variability that will be implemented. The vector of mean values $\overrightarrow{\mu}$ is in accordance to nominal transistors parameters in ω. The vector of standard deviations $\overrightarrow{\sigma}$ depends on the foundry and technology node, and the vector of steps $\varepsilon(\beta)$ are as small as possible (in this work the steps are assumed to be equal to the standard deviations). Finally, C is the matrix of co-variances between the electrical parameters.

At line 2 of the algorithm the nominal value is computed, which will be an approximation for the average.

First, for all the transistors (lines 3-24), numerical derivatives for WD parameters (lines 4-10) and D2D parameters (lines 11-23) are computed. Notice that the approach for derivative computation requires 2 HSpice runs for each parameter, since the algorithm being studied employs a $O(\varepsilon^2)$ approximation for the computation of derivatives.

For WD parameters, electrical simulations are computed (l. 5 and 6) and in the next step the derivative is calculated using these values (l. 7). Then the sensibility of variance to the parameter is computed in l. 8, and added to the circuit variance.

For D2D parameters, electrical simulations are run (l. 12 and 13) and next the derivative is computed (l. 14). As the contribution of D2D parameters is given in function of the sum of these parameters ($\zeta(\beta^j)$ at line 16) to all transistors (eq. 10), the actual contribution is computed at line 22 (at l. 15 K is employed as a temporary variable).

Correlations are added to the circuit variance in lines 25-41. For all transistors, add correlation between WD to WD parameters (l. 26 - 30), D2D to D2D (l. 31-35), and WD and D2D (36-40). The reader should notice that the number of correlation coefficients given as input does not affect the number of HSpice simulations needed. Thus, the covariances do not affect the running time of the method.

The algorithm computes the following outputs:

1. matrix of contributions K, which represents the contribution of the parameter $1 \leq j \leq (p+q)$ of the transistor $1 \leq i \leq n$;
2. variance of circuit response σ_F^2 and
3. approximation for the average value of the response μ_F.

7.1 Complexity

The proposed tool runs as a front-end for HSpice and computational complexity of each electrical simulation depends on the kind of analysis – DC, AC, transient, ... – and the

Algorithm 1 Error propagation using numerical derivatives

Require: ω, $\overrightarrow{\tau} = (\tau_1, ..., \tau_n)$, $\overrightarrow{\alpha} = (\alpha_1^1, ..., \alpha_n^p)$, $\overrightarrow{\beta} = (\beta_1^1, ..., \beta_n^q)$,
$\overrightarrow{\mu} = (\mu(\alpha_1^1), ..., \mu(\alpha_n^p), \mu(\beta_1^1), ..., \mu(\beta_n^q))$,
$\overrightarrow{\sigma} = (\sigma(\alpha_1^1), ..., \sigma(\alpha_n^p),$
$\sigma(\beta_1^1), ..., \sigma(\beta_n^q))$, $\overrightarrow{\varepsilon} = (\varepsilon(\alpha^1), ..., \varepsilon(\alpha^p), \varepsilon(\beta^1), ..., \varepsilon(\beta^q))$, C_{p+q}^{p+q}

$\sigma_F^2 \leftarrow 0; \zeta(\overrightarrow{\beta}) \leftarrow 0$
$\mu_F \leftarrow F(\alpha_1^1 = \mu(\alpha_1^1), \ldots, \alpha_n^p = \mu(\alpha_n^p), \beta_1^1 = \mu(\beta_1^1), \ldots, \beta_n^q = \mu(\beta_n^q), \omega)$
for all i such that $1 \leq i \leq n$ **do**
  for all j such that $1 \leq j \leq p$ **do**
    $s_\downarrow \leftarrow F(\alpha_1^1 = \mu(\alpha_1^1), \ldots, \alpha_i^j = \mu(\alpha_i^j) - \varepsilon(\alpha^j), \ldots, \alpha_n^p = \mu(\alpha_n^p), \beta_1^1 = \mu(\beta_1^1), \ldots, \beta_n^q = \mu(\beta_n^q), \omega)$
    $s_\uparrow \leftarrow F(\alpha_1^1 = \mu(\alpha_1^1), \ldots, \alpha_i^j = \mu(\alpha_i^j) + \varepsilon(\alpha^j), \ldots, \alpha_n^p = \mu(\alpha_n^p), \beta_1^1 = \mu(\beta_1^1), \ldots, \beta_n^q = \mu(\beta_n^q), \omega)$
    $s_i^j \leftarrow s_\uparrow - s_\downarrow$
    $K_i^j \leftarrow \left(\frac{s_i^j}{2\varepsilon(\alpha_i^j)}\right)^2 \sigma^2(\alpha^j)$
    $\sigma_F^2 \leftarrow \sigma_F^2 + K_i^j$
  end for
  for all j such that $1 \leq j \leq q$ **do**
    $s_\downarrow \leftarrow F(\alpha_1^1 = \mu(\alpha_1^1), \ldots, \alpha_n^p = \mu(\alpha_n^p), \beta_1^1 = \mu(\beta_1^1), \ldots, \beta_i^j = \mu(\beta_i^j) - \varepsilon(\beta^j), \ldots, \beta_n^q = \mu(\beta_n^q), \omega)$
    $s_\uparrow \leftarrow F(\alpha_1^1 = \mu(\alpha_1^1), \ldots, \alpha_n^p = \mu(\alpha_n^p), \beta_1^1 = \mu(\beta_1^1), \ldots, \beta_i^j = \mu(\beta_i^j) + \varepsilon(\beta^j), \ldots, \beta_n^q = \mu(\beta_n^q), \omega)$
    $s_i^{p+j} \leftarrow s_\uparrow - s_\downarrow$
    $K_i^{p+j} \leftarrow \left(\frac{s_i^{p+j}}{2\varepsilon(\beta^j)}\right)$
    $\zeta(\beta^j) \leftarrow \zeta(\beta^j) + K_i^{p+j}$
  end for
end for
for all j such that $1 \leq j \leq q$ **do**
  $\sigma_F^2 \leftarrow \sigma_F^2 + \zeta(\beta^j)^2 \sigma^2(\beta^j)$
  for all i such that $1 \leq i \leq n$ **do**
    $K_i^{p+j} \leftarrow (\frac{K_i^{p+j}}{\varepsilon(\beta_j)})(\zeta(\beta_j)^2 \sigma^2(\beta_i^j))$
  end for
end for
for all i such that $1 \leq i \leq n$ **do**
  for all j such that $1 \leq j \leq p$ **do**
    for all k such that $j \leq k \leq p$ **do**
      $\sigma_F^2 \leftarrow 2 \times s_i^{p+j} \times s_i^{p+k} \times C_j^k$
    end for
  end for
  for all j such that $1 \leq j \leq q$ **do**
    for all k such that $j \leq k \leq q$ **do**
      $\sigma_F^2 \leftarrow 2 \times s_i^j \times s_i^k \times C_{p+j}^{p+k}$
    end for
  end for
  for all j such that $1 \leq j \leq p$ **do**
    for all k such that $j \leq k \leq p$ **do**
      $\sigma_F^2 \leftarrow 2 \times s_i^j \times s_i^{p+k} \times C_j^{p+k}$
    end for
  end for
end for
Ensure: K, μ_F, σ_F^2

algorithms implemented by the simulator. For a suitable study about transient and DC analysis performed by spice refer to [17] and its references.

The number of electrical simulations required to compute the circuit variance is a function of the following inputs:

n: number of transistors
p: number of parameters that present WD variations
q: number of parameters that present D2D variations
d: numerical method employed to compute the derivatives
ω**:** spice netlist

Section 6 introduced 3 linear approximations for the computation of partial derivatives. Each one has a different accuracy order and each requires a given number of electrical simulations to compute partial derivative for a variable. Let d be the number of electrical simulations required to compute the derivative using each numerical method. Therefore, d is related to the desired accuracy as follows:

Accuracy	**d**	**Equation**
$0(\varepsilon)$	1	12
$0(\varepsilon^2)$	2	13
$0(\varepsilon^4)$	4	14

One electrical simulation is run using nominal values in order to compute the nominal response, which is an approximation for the average.

The complexity required to compute variance using numerical error propagation is exactly

$$C(n,d,p,q,\omega) = [d \times n \times (p+q)+1]C_{spice}(n,p,q,\omega)$$

where $C_{spice}(n,p,q,\omega)$ is the computational complexity of one spice run for the given netlist and the parameters n,p,q.

8 Case Study: Yield Analysis of a SRAM Memory

In the last sections the mathematical foundations of the proposed parametric yield analysis methodology were exposed. This section presents a case study, describing the application of the method to yield analysis and yield improvement of a SRAM memory based on cell access time failure [7]. The transistors nomenclature employed in this work is as shown in picture 2. Electric simulations were run in HSpice, using the 70nm node Berkeley Predictive Technology Model [8].

Variations in threshold voltage, channel width and channel length are modeled. The electrical parameters which are assumed to show variability are as follows:

Parameter	Type	nominal	3σ
Length	WD	70 nm	3.5 nm
Length	D2D	70 nm	3.5 nm
Width	WD	100 nm	7.5 nm
Width	D2D	100 nm	7.5 nm
Vt(PMOS)	WD	-0.22 V	40 mV
Vt(NMOS)	WD	0.2 V	40 mV

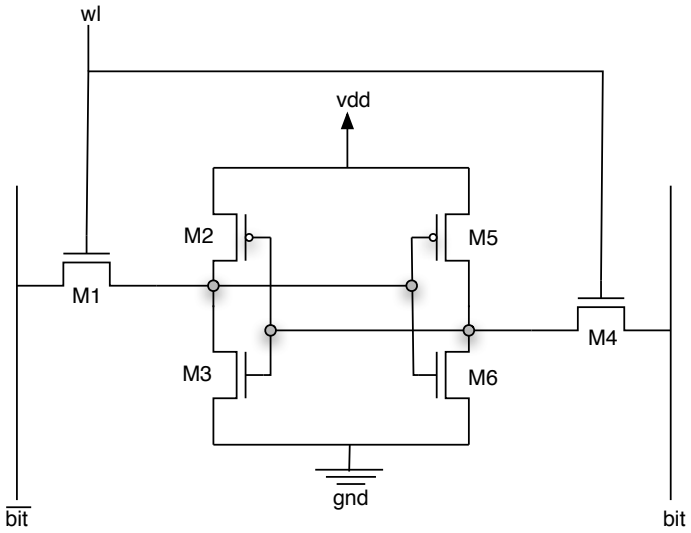

Fig. 2. 6-transistors SRAM cell

This data is in accordance to the ITRS [11] and [19]. In our case study, we assume that there is no correlation between parameters, but we consider the functional dependence given by equation 1.

Access time is the time needed to read the data stored in a cell, computed as the time needed to discharge the bit line (*bit*) or the negated bit line ($\overline{bit}$) to $0.5VDD$, if a zero or an one is stored in the cell, respectively. Access time failure is assumed to occur if the access time of a given cell is greater than the maximum value allowed for the design. For the results presented in this section, both bit line (bit) and negated bit line ($\overline{bit}$) are assumed to be pre-charged to VDD. After pre-charge, signal *wl* is set to VDD and transistors M1 and M4 are switched to on. *Bit* is maintained at VDD if an one is stored in the cell, or is discharged to *gnd* if the cell stores a zero.

The access time may be written as a function of the random variables, where considering the cell symmetry we have:

$$T_{AC} = T_{AC}(L_{M1},\ldots,L_{M3},W_{M1},\ldots,W_{M3},Vt_{M1},\ldots,Vt_{M3},)$$

Channel width and channel length will be considered to present WD and D2D variations. Thus, according to equation 4 access time can be written as
$T_{AC} = T_{AC}(L^{wd}_{M1},\ldots,L^{wd}_{M3},L^{d2d},W^{wd}_{M1},\ldots,W^{wd}_{M3},W^{d2d},Vt_{M1},\ldots,Vt_{M3})$.

Rewriting equation 7 we have:

$$\sigma^2_{T_{AC}} = \sum_{i=1}^{3} \left(\frac{\partial T_{AC}}{\partial W^{wd}_{Mi}} \bigg|^2_{W^{wd}_{Mi}=\overline{W^{wd}_{Mi}}} \right) \sigma^2_{W^{wd}} + \left(\sum_{i=1}^{3} \frac{\partial T_{AC}}{\partial W^{d2d}_{Mi}} \bigg|_{W^{d2d}_{Mi}=\overline{W^{d2d}_{Mi}}} \right)^2 \sigma^2_{W^{d2d}}$$
$$+ \sum_{i=1}^{3} \left(\frac{\partial T_{AC}}{\partial L^{wd}_{Mi}} \bigg|^2_{L^{wd}_{Mi}=\overline{L^{wd}_{Mi}}} \right) \sigma^2_{L^{wd}} + \left(\sum_{i=1}^{3} \frac{\partial T_{AC}}{\partial L^{d2d}_{Mi}} \bigg|_{L^{d2d}_{Mi}=\overline{L^{d2d}_{Mi}}} \right)^2 \sigma^2_{L^{d2d}}$$
$$+ \sum_{i=1}^{3} \left(\frac{\partial T_{AC}}{\partial Vt_{Mi}} \bigg|^2_{Vt_{Mi}=\overline{Vt_{Mi}}} \right) \sigma^2_{Vt} \tag{15}$$

Considering T_{MAX} a design constraint related to target circuit clock, then the probability p of a SRAM cell do not present access time violation failure is given by

$$p = P(T_{AC} \leq T_{MAX}) = \frac{1}{\sigma_{T_{AC}}\sqrt{2\pi}} \int_{-\infty}^{T_{MAX}} e^{-\frac{(x-\mu_{T_{AC}})^2}{2\sigma^2_{T_{AC}}}} dx \tag{16}$$

Next sections expose the algorithm 7 applied to a SRAM memory, using the specific formulations in order to provide a comprehensive explanation. Circuit partial derivatives are computed using electric simulations. Derivatives and variances are inputs for equation 15, which gives SRAM cell access time variance. The PDF is plotted by using the standard deviate obtained applying error propagation and mean value approximated by simulation using nominal values for input parameters. After, the yield of the entire memory chip is computed. In the last section of this case study, we analyze the contribution of each parameter to the circuit variability, and improve yield resizing critical transistors.

8.1 Yield of the SRAM cell

The access time variance is computed by error propagation, which has as input the numerical estimates of derivatives and the standard deviates. The derivatives can be computed using equation 12, 13 or 14, according to the desired trade-of between accuracy and run-time. Higher number of points implies in higher order accuracy and running time increases. Figure 3 shows a comparison between PDF obtained using values computed using error propagation (using 1, 2 and 4 points around mean) and histogram given by Monte Carlo. The yield of the SRAM cell considering access time failure as a function of the design constraint T_{MAX} is shown by figure 4.

Numerical error propagation using 1, 2 or 4 points requires respectively 10, 19 or 37 electrical simulations, while we run Monte Carlo using 10^4 runs. Monte Carlo simulation with 10^4 runs has a running time of $\tilde{3}4000$ seconds, while the running time for error propagation with numerical derivatives using 2 points is less than 80 seconds in a dual processor Sun Fire V240 (UltraSPARC IIIi 1 GHz).

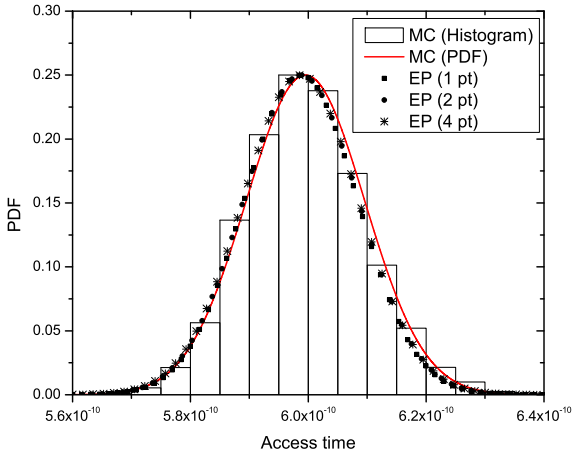

Fig. 3. Monte Carlo histogram (10^4 Spice simulations) compared to PDF obtained by error propagation using 1 point (10 Spice simulations), 2 points (19 Spice simulations) and 4 points (37 Spice simulations)

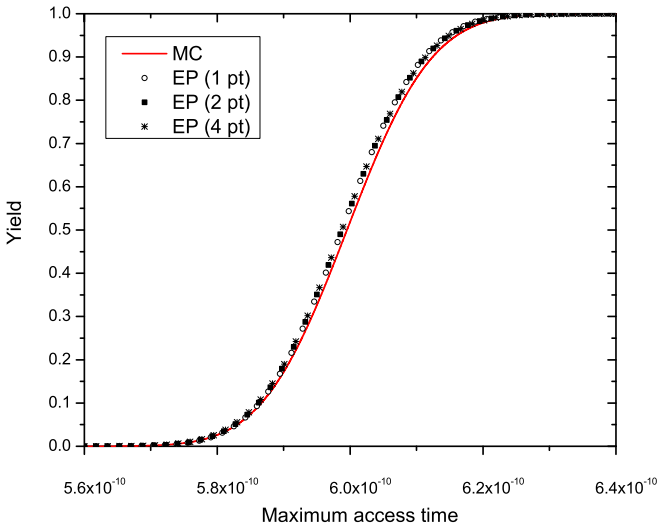

Fig. 4. Yield of the SRAM cell as a function of T_{MAX} computed using Monte Carlo (10^4 runs) compared to error propagation using 1 point (10 runs), 2 points (19 runs) and 4 points (37 runs)

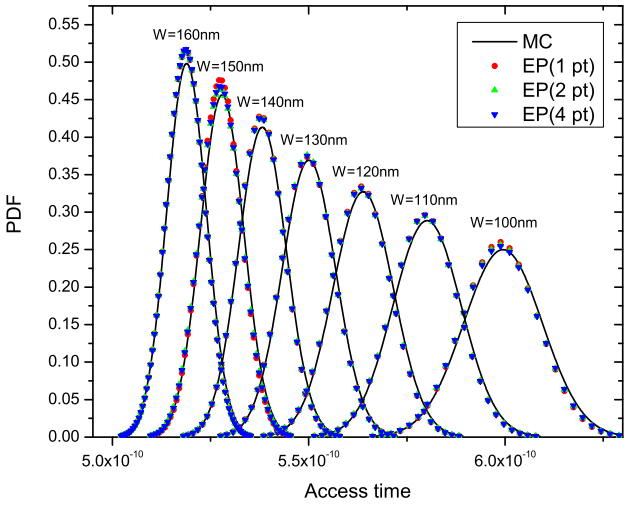

Fig. 5. Influence of Transistors Width in Access Time Variability

8.2 Resizing of the critical transistors

Once the sensitivity and contribution of each electric parameter are computed, the design can be optimized in order to diminish the effect of these parameters, decreasing the variance. Although this work presents an yield optimization based in the access time failure only, there are of other issues which need to be taken into account during yield optimization (for instance read margin, write margin and hold margin in the case of SRAM cells).

By resizing the transistors, three components of access time variance are affected. While W_{wd} and W_{d2d} are directly affected, Vt variance decreases because of the functional relation given by equation 1.

Figure 5 presents the access time PDFs for several transistor channel widths varying from 100nm to 160nm (in these experiments we assume $W_{M1} = \ldots = W_{M6} = W$). The PDFs were computed using MC (10^4 runs) and EP (9 runs for 1 point, 19 runs for 2 points and 37 runs for 4 points). SRAM cell access time variance and average are inversely proportional to transistors channel width. Thus, memory yield can be increased by properly sizing the transistors which more contribute to access time variance.

Figure 6 reports the sensitivity of the access time variance to each parameter (in percentage). The transistors are shown in pairs because of the symmetry of the SRAM cell. The transistor which contributes most to the variance is M1-M4($\tilde{6}5\%$), and the second most preponderant is M3-M6 ($\tilde{3}5\%$). The most significant parameter is Vt of transistor M1-M4: 27% of the access time variance is caused by this parameter.

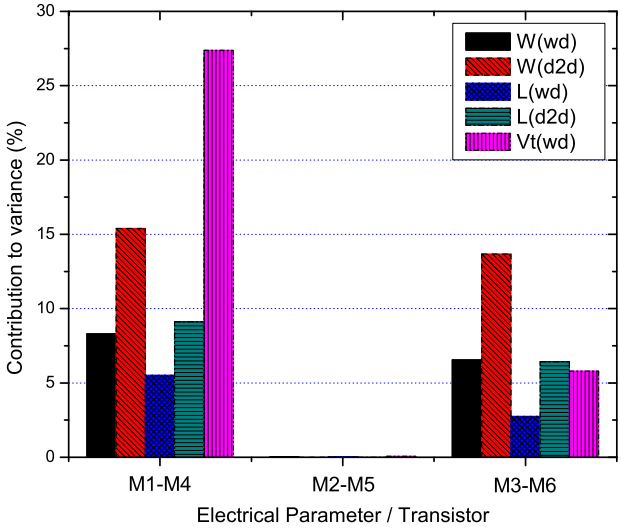

Fig. 6. Sensitivity of the access time variance to each parameter

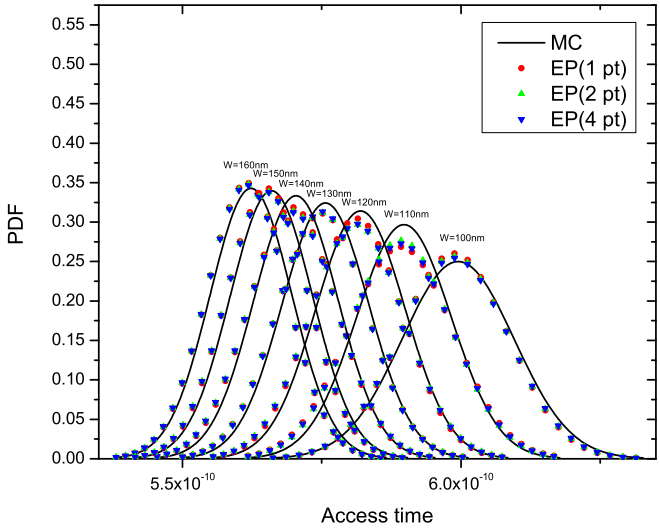

Fig. 7. Impact of transistors M1 and M4 in access time variability by Monte Carlo (10^4 runs) compared to error propagation using derivatives with 1 (10 runs), 2 (19 runs) and 4 points (37 runs)

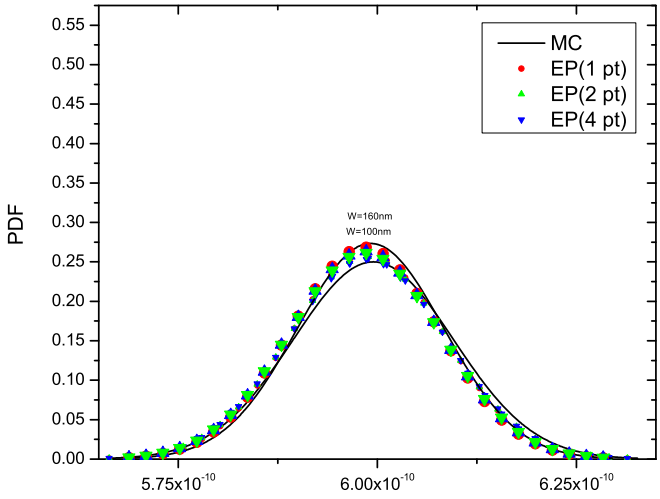

Fig. 8. Impact of transistors M2 and M5 in access time variability by Monte Carlo (10^4 runs) compared to error propagation using derivatives with 1 (10 runs), 2 (19 runs) and 4 points (37 runs)

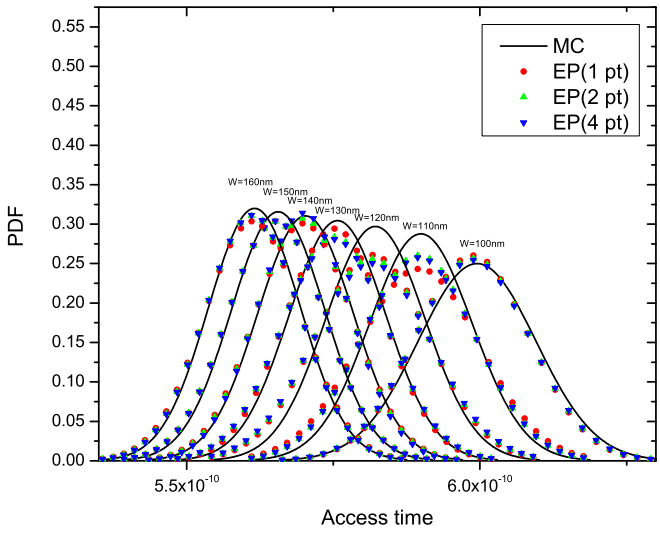

Fig. 9. Impact of transistors M3 and M6 in access time variability by Monte Carlo (10^4 runs) compared to error propagation using derivatives with 1 (10 runs), 2 (19 runs) and 4 points (37 runs)

Figure 7 reveals the access time PDFs obtained from experiments where $W_2 = W_3 = W_5 = W_6 = 100nm$ and $W_1 = W_4$ varies from $100nm$ to $160nm$ in increments of 10nm. By sizing these transistors the standard deviate decreases 35% (comparing $W_1 = W_4 = 160nm$ against $W_1 = W_4 = 100nm$) while average decreases 7%. Skewness and average decrease as W increases.

Figure 8 points out the impact of resizing transistors M2 and M5. As the previous analysis of the contribution of these parameters indicated, they do not impact in the access time variance. Thus, the skewness is not correlated to W. Figure 9 presents the impact of resizing transistors M3 and M6. The standard deviate decreases by 20% and average decreases by 7% increasing these transistors by 60%.

The above simulations indicate that Monte Carlo and error propagation both present similar results which corroborate the hypothesis that the skewness can be decreased by optimizing the parameters pointed out by the methodology. The average difference between standard deviate computed by error propagation using 1 point and the one computed by Monte Carlo is 7%. For the simulations using 1 point around mean for derivatives, a total of 10 electrical simulations must be computed. Thus, this approach means to improve running time $1000\times$ compared to MC using 10^4 runs. Computation of partial derivatives for access time using 2 points gives and adjustment of $O(\varepsilon^2)$, and in this case 17 electrical simulations are required. Using this approach the average difference between standard deviate computed using MC and EP is 6%, and the speedup is up to $580\times$. Derivatives using 4 points requires 37 simulations, but for our case study the precision $O(\varepsilon^4)$ does not significantly improve solution accuracy (average difference is 6%, which is similar to approach using 2 points). In this case, approach using 2 points for numerical derivatives conciliates running time and solution quality, but for some application a higher order approach may be necessary.

8.3 Yield Analysis of the SRAM memory

SRAM memories present a regular architecture in which most of the chip area is dedicated to regularly disposed SRAM cells. Consider a memory grid designed with N_{COL} columns, N_{ROW} rows of SRAM cells and N_R redundant columns, as figure 10 illustrates. If process fabrication variability causes at least one memory cell to fail in a column, that column must be discarded and replaced by a redundant column – this can be done during circuit test phase, setting a set of fuses. If process variability causes more than N_R (at least $N_R + 1$) columns to fail, than the circuit is considered faulty and must be discarded, reducing yield and increasing product cost.

Denoting p as the probability of the SRAM cell to work properly in the presence of process variability, $P_{COL} = (p)^{N_{ROW}}$ is the probability that none cell fails in the column. We are interested in the probability to manufacture N_{COL} working columns in a total of $N_{COL} + N_R$ designed columns. Thus, the yield (percentage of working chips) of a SRAM memory design is given by a binomial distribution, [13]:

$$P_{MEM} = \sum_{i=N_{COL}}^{N_{COL}+N_R} \binom{N_{COL}+N_R}{i} (P_{COL})^i (1-P_{COL})^{N_{COL}+N_R-i} \tag{17}$$

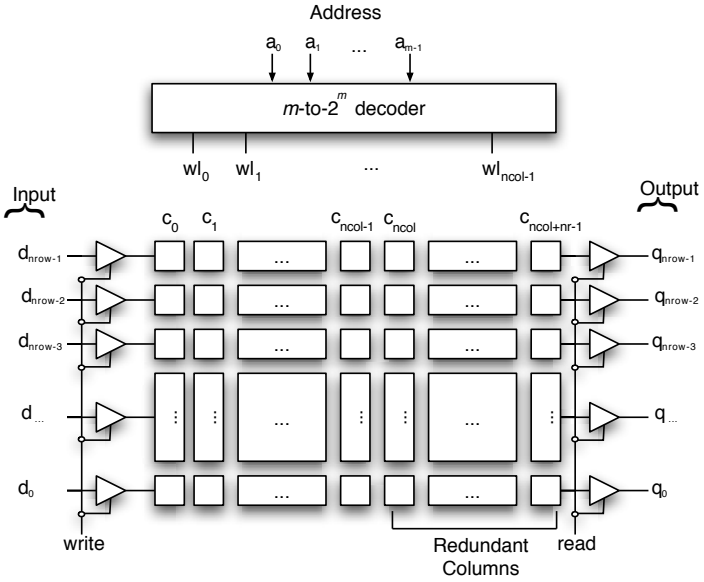

Fig. 10. Scheme of a SRAM memory

Consider a 2 Kbytes SRAM memory for which the architectural parameters are $N_{COL} = 512$ and $N_{ROW} = 32$ (rows of 4 bytes without redundancy in the row). Also assume $N_R = 24$ (4% of total number of columns). Figure 11 shows the memory yield as a function of T_{MAX} for the design where all transistors have $W = 100$ and for the design which transistors M1-M4 are re-sized to $W = 160$. The figure presents points computed using equation 17 (squares) as well as the fit of its points to a logistic function (line) given by

$$yield(T_{\max}) = a_2 + \frac{(a_1 - a_2)}{1 + (T_{\max}/T_0)^p} \tag{18}$$

where a_1 , a_2 , T_0 and p are parameters.

The logistic function fit to the design with W=100 for all transistors has $T_0 = 6.28 \times 10^{-10}$ and $p = 1702.7$, while the design where transistors M1-M4 are re-sized to W=160 presents $T_0 = 5.8^{10} - 10$ and $p = 2151.1$. Thus, the re-sized circuit presents a smaller mean for the access time and a yield that grows faster as function of T_{MAX}, if compared to the original design.

9 Conclusions

In this work we present a computer-efficient method for electrical yield simulation of combinational and sequential circuit blocks. The method is based on error propagation. The numerical methods employed for the computation of derivatives assures the independence of topology and parameters to be analyzed. The main contributions of this

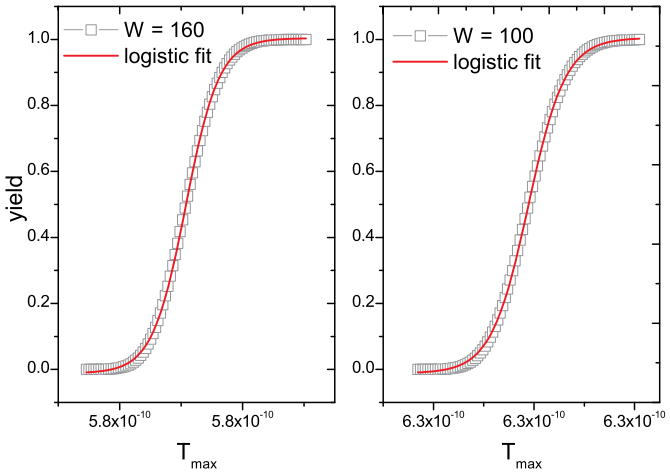

Fig. 11. SRAM memory yield. Increasing W of transistor M1-M4, the memory yield increases for the same T_{MAX}

work are (1) support for WD and D2D variations, as well as co-variances; (2) yield analysis based solely on numerical formulations (no analytical formulations are needed), including the study of accuracy, numerical complexity and higher order numerical approximations; (3) analysis of the sensitivity of the variance to the electrical parameters; and (4) development of a general method (algorithm) which can be employed for yield analysis of combinational or sequential blocks.

We studied the three numerical formulations for the computation of derivatives. They differ in the upper bound of the numerical error as well as the number of electrical simulations required to compute the derivatives. We verify an accuracy increase of 1% in the formulation which has $O(\varepsilon^2)$ in comparison to $O(\varepsilon)$. No significant improvement is observed when using the $O(\varepsilon^4)$ formulation. This is due to the limited precision of the electrical simulations, since the function being modeled is not smooth.

Based on an error propagation formulation, we derived the sensitivity of the circuit response variance as a function of each electrical parameter. This analysis plays a fundamental role when dealing with any kind of electrical block – memories, library cells, sequential and combinational blocks –, because it can guide the designer to figure out what parameters are the most preponderant to the variance in circuit behavior. During the yield optimization phase, this data can lead to a better understanding of how to improve circuit yield. This is an advantage of using error propagation instead of sampling techniques.

The proposed methodology keeps the generality of electrical (Spice) level simulations, and can thus be applied to yield analysis in many CMOS circuits. The method

shows results that are statistically equivalent to the usual sampling techniques, like Monte Carlo simulation, while increasing simulation speed by orders of magnitude.

References

1. A. Agarwal, D. Blaauw, and V. Zolotov. Statistical timing analysis for intra-die process variations with spatial correlations. In *Proc. International Conference on Computer Aided Design*, pages 900– 907, Nov. 2003.
2. A. Agarwal et al. A process-tolerant cache architecture for improved yield in nanoscale technologies. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 13(1):27–38, 2005.
3. A. Agarwal et al. Process variation in embedded memories: failure analysis and variation aware architecture. *IEEE Journal of Solid-State Circuits*, 40(9):1804– 1814, Sept. 2005.
4. A. Agarwal, B. C. Paul, and K. Roy. Process variation in nano-scale memories: failure analysis and process tolerant architecture. In *Proceedings of Custom Integrated Circuits Conference*, pages 353–356, 2004.
5. Jacques G. Amar. The monte carlo method in science and engineering. *Computing in Science and Engineering*, 8(2):9–19, 2006.
6. K. Bernstein, D. J. Frank, A. E. Gattiker, W. Haensch, B. L. Ji, S. R. Nassif, E. J. Nowak, D. J. Pearson, and N. J. Rohrer. High-performance cmos variability in the 65-nm regime and beyond. *IBM J. Res. Dev.*, 50(4/5):433–449, 2006.
7. Yang Byung-Do and Kim Lee-Sup. A low-power sram using hierarchical bit line and local sense amplifier. *IEEE Journal of Solid-State Circuits*, 40(6):1366– 1376, June 2005.
8. Y. Cao et al. New paradigm of predictive mosfet and interconnect modeling for early circuit design. In *Proc. Custom Integrated Circuit Conference*, pages 201–204, June 2000.
9. B. Cheng, S. Roy, G. Roy, F. Adamu-Lema, and A. Asenov. Impact of intrinsic parameter fluctuations in decanano mosfets on yield and functionality of sram cells. *Solid-State Electronics*, 49(5):740, 2005.
10. Kim Hyun-Woo et al. Experimental investigation of the impact of lwr on sub-100-nm device performance. *IEEE Transactions on Electron Devices*, 51(12):1984– 1988, Dec. 2004.
11. ITRS. *International Technology Roadmap for Semiconductors*, 2005. Available at ¡http://www.itrs.net¿. Visited on Nov. 2006.
12. Kunhyuk Kang, B. C. Paul, and K. Roy. Statistical timing analysis using levelized covariance propagation. In *Proceedings Design, Automation and Test in Europe*, pages 764–769 Vol. 2, 2005.
13. H. Mahmoodi, S. Mukhopadhyay, and K. Roy. Estimation of delay variations due to random-dopant fluctuations in nanoscale cmos circuits. *IEEE Journal of Solid-State Circuits*, 40(9):1787–1796, 2005.
14. S. Mukhopadhyay, H. Mahmoodi, and K. Roy. Statistical design and optimization of sram cell for yield enhancement. In *Proc. IEEE/ACM International Conference on Computer Aided Design*, pages 10– 13, Nov. 2004.
15. S. Mukhopadhyay, H. Mahmoodi-Meimand, and K. Roy. Modeling and estimation of failure probability due to parameter variations in nano-scale srams for yield enhancement. In *Proc. Symposium on VLSI Circuits*, pages 64–67, Piscataway, NJ, June 2004.
16. N. S. Nagaraj et al. Beol variability and impact on rc extraction. In *Proc. Design Automation Conference*, pages 758–759, Anaheim, California, USA, June 2005.
17. Lawrence W. Nagel. *SPICE2: A Computer Program to Simulate Semiconductor Circuits*. University of California, Electronics Research Laboratory, College of Engineering, Berkeley, CA, 1975.

18. Sani R. Nassif. Modeling and forecasting of manufacturing variations. In *Proc. International Workshop on Statistical Metrology*, 2000.
19. S.R Nassif. Design for variability in dsm technologies [deep submicron technologies]. In *Proc. IEEE International Symposium on Quality Electronic Design*, pages 451–454, 2000.
20. M. Orshansky et al. Impact of spatial intrachip gate length variability on the performance of high-speed digital circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 21(5):544–553, 2002.
21. L. G. Parrat. *Probability and Experimental Errors on Science*. John Wiley and Sons, New York, NY, USA, 1961.
22. Rajeev R. Rao et al. Parametric yield estimation considering leakage variability. In *Proc. Design Automation Conference, 41.*, pages 442–447, 2004.
23. SYNOPSYS INC. *HSPICE Simulation and Analysis User Guide*, 2005.
24. Yuan Taur et al. Cmos scaling into the nanometer regime. *Proceedings of the IEEE*, 85(4):486–504, Apr. 1997.
25. Yuan Taur and Tak H. Ning. *Fundamentals of modern VLSI devices*. Cambridge University Press, New York, NY, USA, 1998.
26. Chandu Visweswariah. Death, taxes and failing chips. In *Proc. Design and Automation Conference, DAC, 40.*, pages 343–347, Anaheim, CA, USA, 2003. New York: ACM Press.
27. P. S. Zuchowski et al. Process and environmental variation impacts on asic timing. In *Proc. IEEE/ACM International Conference on Computer Aided Design, ICCAD*, pages 336–342, 2004.

A 4^{th} Order Derivative

By expansion in Taylor Series we have:

$$\begin{aligned} f(\overline{x_1},\ldots,\overline{x_i}+2\varepsilon,\ldots,\overline{x_n}) &= f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + 2\varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \\ &+ \frac{4\varepsilon^2}{2!} f''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \frac{8\varepsilon^3}{3!} f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \\ &+ \frac{16\varepsilon^4}{4!} f^{(4)}(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \qquad (19) \\ f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n}) &= f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \\ &+ \frac{\varepsilon^2}{2!} f''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \frac{\varepsilon^3}{3!} f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \\ &+ \frac{\varepsilon^4}{4!} f^{(4)}(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \qquad (20) \\ f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n}) &= f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) - \varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \\ &+ \frac{\varepsilon^2}{2!} f''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) - \frac{\varepsilon^3}{3!} f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \\ &+ \frac{\varepsilon^4}{4!} f^{(4)}(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \qquad (21) \end{aligned}$$

$$f(\overline{x_1},\ldots,\overline{x_i}-2\varepsilon,\ldots,\overline{x_n}) = f(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) - 2\varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \frac{4\varepsilon^2}{2!}f''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) - \frac{8\varepsilon^3}{3!}f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \frac{16\varepsilon^4}{4!}f^{(4)}(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) \quad (22)$$

Combining the equations 19 and 22 we obtain:

$$f(\overline{x_1},\ldots,\overline{x_i}+2\varepsilon,\ldots,\overline{x_n}) - f(\overline{x_1},\ldots,\overline{x_i}-2\varepsilon,\ldots,\overline{x_n}) = 4\varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \frac{16\varepsilon^3}{3!}f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + O(\varepsilon^5) \quad (23)$$

Similarly, from equations 20 and 21 we can write:

$$f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n}) - f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n}) = 2\varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + \frac{2\varepsilon^3}{3!}f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + O(\varepsilon^5) \quad (24)$$

Multiplying equation 24 for (-8) we have:

$$\frac{16\varepsilon^3}{3!}f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) = -16\varepsilon f'(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n}) + 8f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n}) - 8f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n}) + O(\varepsilon^5) \quad (25)$$

and substituting $\frac{16\varepsilon^3}{3!}f'''(\overline{x_1},\ldots,\overline{x_i},\ldots,\overline{x_n})$ given by equation 25 on equation 23, then an $O(\varepsilon^4)$ approximation can be derived as follows:

$$\frac{\partial f}{\partial x_i}(\overline{x_1},\ldots,\overline{x_n}) = \frac{1}{3}\cdot\frac{[-f(\overline{x_1},\ldots,\overline{x_i}+2\varepsilon,\ldots,\overline{x_n}) + f(\overline{x_1},\ldots,\overline{x_i}-2\varepsilon,\ldots,\overline{x_n})]}{4\varepsilon} + \frac{4}{3}\cdot\frac{f(\overline{x_1},\ldots,\overline{x_i}+\varepsilon,\ldots,\overline{x_n}) - f(\overline{x_1},\ldots,\overline{x_i}-\varepsilon,\ldots,\overline{x_n})}{2\varepsilon} + O(\varepsilon^4) \quad (26)$$