Hindawi Publishing Corporation
VLSI Design
Volume 2013, Article ID 984376, 22 pages
http://dx.doi.org/10.1155/2013/984376

Research Article

Computational Performance Optimisation for Statistical Analysis of the Effect of Nano-CMOS Variability on Integrated Circuits

Zheng Xie and Doug Edwards

School of Computer Science, The University of Manchester, Manchester M13 9PL, UK

Correspondence should be addressed to Zheng Xie; z.xie@mmu.ac.uk

Received 16 November 2012; Revised 15 May 2013; Accepted 15 May 2013

Academic Editor: Chien-In Henry Chen

The intrinsic variability of nanoscale VLSI technology must be taken into account when analyzing circuit designs to predict likely yield. Monte-Carlo- (MC-) and quasi-MC- (QMC-) based statistical techniques do this by analysing many randomised or quasirandomised copies of circuits. The randomisation must model forms of variability that occur in nano-CMOS technology, including "atomistic" effects without intradie correlation and effects with intradie correlation between neighbouring devices. A major problem is the computational cost of carrying out sufficient analyses to produce statistically reliable results. The use of principal components analysis, behavioural modeling, and an implementation of "Statistical Blockade" (SB) is shown to be capable of achieving significant reduction in the computational costs. A computation time reduction of 98.7% was achieved for a commonly used asynchronous circuit element. Replacing MC by QMC analysis can achieve further computation reduction, and this is illustrated for more complex circuits, with the results being compared with those of transistor-level simulations. The "yield prediction" analysis of SRAM arrays is taken as a case study, where the arrays contain up to 1536 transistors modelled using parameters appropriate to 35 nm technology. It is reported that savings of up to 99.85% in computation time were obtained.

1. Introduction

Anticipating the impact of device variability on performance is a critical aspect of design procedures for integrated circuits. With nanoscale technology, "intradie" variability, causing behavioural variation from device to device within each die, is an important consideration [1]. With some dimensions approaching atomic scales, intrinsic atomic scale variations such as line edge roughness and dopant granularity have become the main sources of such variation [2, 3]. These "atomistic" variabilities are random in nature and result in "within-die" fluctuation which cannot be disregarded. Traditional variability analysis, based on "worst case" corner models [4] and guard bands for parameter variations, is likely to be very pessimistic in its estimations of the effects of the variability. Consequently, new circuit analysis techniques are needed which adopt a statistical treatment of the intradie variability of device performances. This has led to much interest in statistical analysis which can take into account parameter variation in all parts of a design in a single comprehensive computational procedure, allowing the impact on yield to be efficiently estimated.

Because of the high dimensionality of the parameter space, it is very difficult to derive analytical models of large-scale ICs for analysis. Sets of equations describing specific circuit performance parameters, like timing or yield, in terms of the huge number of parameters would be very difficult to derive analytically. Therefore, the use of conventional statistical methods, based on the analysis of such equations, has restricted applicability for the variability analysis of nanoscale ICs. With Monte Carlo (MC) analysis methods, the integrated circuits are simulated directly, and there is no need to derive differential equations that describe the dependency of the properties of interest on circuit parameters. The only requirement is that the circuit properties to be modeled are capable of being described by probability density functions

(pdfs) that are dependent on the pdfs of circuit parameters and input variables. Monte Carlo analysis proceeds by generating a random sample of the value of each circuit parameter and each input variable, based on their known statistical properties. It then simulates the circuit thus obtained, using a finite difference analysis package, to compute a random sample of the circuit property of interest. The process is repeated many times to obtain a sequence of samples of the property of interest from which statistical models can be derived.

In nanoscale technology, "atomistic" variabilities are random and generally uncorrelated from device to device. According to Srivastava et al. [4], a critical form of variability is gate-length variability which occurs "interdie" due to variation in exposure time and "intradie" because of other lithography effects [5]. Device threshold voltages will be dependent on gate lengths and channel doping concentration, and variation in these parameters will cause comparable amounts of independent interdie variability and spatially correlated intradie variability. Intradie variations will have therefore both spatially correlated and uncorrelated contributions. The effect of the correlation in "intradie" variation must be given consideration.

MC analyses are particularly suitable for nanoscale IC statistical simulation to achieve statistical estimates of properties of interest. General conclusions drawn from the analysis of many randomised samples of a circuit can be made representative of the effects of variability. To quantify these conclusions, statistical averages are produced. The statistical reliability of the conclusions generally improves as the number of circuit increases, though the rate of improvement can often be increased by carefully choosing the samples. MC techniques are especially useful for predicting the "yield" from many copies of the circuit when these are manufactured with typical accuracy and parameter variations.

A major problem is the computation required for carrying out sufficient analyses to produce statistically reliable averages. The introduction of intradie correlation into circuit models increases the computational complexity considerably. To simplify the computation, a form of Extreme Value Theory known as statistical blockade (SB) [6, 7] may be employed to identify "rare event" circuits whose responses fall in the "tails" of their pdfs and are therefore worth analyzing. These circuits are needed for circuit yield failure predictions. Circuits in the "body" of the distributions are not analysed. The use of "quasi-Monte Carlo (QMC)" analysis with "low-discrepancy sequences" achieves further computation reduction [8–10]. The use of "statistical behavioural circuit blocks (SBCB)" and "Principal Component Analysis (PCA)" to reduce the dimensionality of the analyses [8] also reduces computational complexity. PCA allows the effects of correlation within the parameters of each device and from device to device to be partitioned thus reducing the sizes of the matrices which characterize the correlation and making them easier to process. Despite these economies, the statistical analysis of large and complicated ICs requires computational resources that can only be provided by parallel and/or distributed computing, for example, by the Condor framework [11, 12].

This paper contains seven sections. The first is an introduction, and Section 2 deals with the use of MC techniques for the statistical analysis of nanoscale technology with the introduction of correlation due to the intradie proximity of components. Section 3 discusses two methods of reducing the dimensionality of the input parameter space to achieve computational efficiency in MC simulations, PCA and SBCB. Section 4 introduces the concept of Extreme Value Theory (EVT) and explores the algorithm known as "statistical blockade" (SB) [11]. The potential for using this technique to achieve major computational savings is explored and illustrated by examples. Section 5 investigates the use of quasi-Monte Carlo (QMC) techniques and "low-discrepancy" sampling to achieve further efficiency improvements, over what was achieved in earlier sections with Monte Carlo circuit simulation. The effect of using a "Sobol" low-discrepancy sequence generator to replace the uniformly distributed pseudorandom number generator previously used to produce the required Gaussian variation is discussed and illustrated by example. Section 6 evaluates the results obtained with VLSI SRAM circuits, and Section 7 presents conclusions. The use of parallel processing for efficiently undertaking the intensive computation required will be discussed, taking into account the intrinsically parallel nature of massive MC simulations.

2. MC Simulation for the Design of Nanoscale VLSI Circuits

2.1. Monte Carlo Methods. Monte Carlo methods use repeated pseudorandom sampling of the behaviour of mathematical equations, or real or simulated systems, to solve mathematical problems or to determine the properties of systems [13]. In this work, the systems are integrated circuits simulated using a finite difference circuit analysis technique, that is, the public licensed version of SPICE, NGSPICE [14]. The transient behaviour of sets of highly complex multidimensional equations describing integrated circuits may thus be analysed by repeated random sampling to produce observations to which statistical inference can be applied to obtain information about the equations or systems. Monte Carlo methods use pseudorandom processes implemented in software. The simulation is not required to be numerically identical to the true physical process since the aim is to produce statistical results such as averages, expectations, and distributions rather than deterministic numerical measurements. The derivation of such results requires a "sampling" of the population of all possible modes of behaviour of the system.

One of the most often quoted applications of Monte Carlo methods is the evaluation of multidimensional integrals [15] such as that in

$$I = \int_{a_1}^{b_1} \int_{a_2}^{b_2} \int_{a_3}^{b_3} \cdots \int_{a_K}^{b_K} f(x_1, x_2, x_3, \ldots, x_K) \times dx_K dx_{K-1} \ldots dx_2 dx_1 \quad (1)$$

$$\text{written as } \int_V f(\underline{x})\, d\underline{x}.$$

Here, $\underline{x}$ is the vector $(x_1, x_2, x_3, \ldots, x_K)$, f is some function of the K variables of $\underline{x}$, and V denotes the region of integration. $f(\underline{x})$ may be evaluated at regularly spaced points or uniformly distributed random points in K-dimensional space as a means of evaluating the integral. The problem with regularly spaced points is that the number of them, say N, must increase exponentially with the dimension K if the error is not to increase exponentially with K. The error with regular spacing and a fixed value of N is known to increase as the Kth root of the order of magnitude [16]. The error for one-dimensional "trapezoidal rule" integration with N regularly spaced points can be shown to be proportional to $1/N^2$, whereas the error for the K-dimensional equivalent of the trapezoidal rule has an error proportional to $1/N^{2/K}$. This is "the curse of dimensionality" [17]. With regular sampling and fixed N, as K increases, each dimension must be sampled more and more sparsely and less and less efficiently, since more and more points will have the same value in a given dimension.

Monte Carlo sampling avoids the inefficiency of the rectangular grids created by regular sampling by using a purely random set of N points uniformly distributed over the K-dimensional region V. The advantages of the Monte Carlo method with randomly distributed points now become apparent. The Monte Carlo error has the property that, for high dimensions K, its error is proportional to $1/(\sqrt{N})$, which means that, to reduce the error by a factor of 10, the sample size N must be increased by a factor of 100. This effect is independent of the order of magnitude K when the input variables are statistically independent which means that there is no correlation between them. It can be simply illustrated by comparing the performances of regular sampling and Monte Carlo integration for the following integral:

$$I = \int_0^{\pi}\int_0^{\pi}\int_0^{\pi}\cdots\int_0^{\pi} \sin\left(x_1 + x_2 + x_3 + \cdots + x_K\right) \times dx_K dx_{K-1}\cdots\ dx_2 dx_1. \tag{2}$$

Figures 1(a), 1(b), and 1(c) show the error for $K = 3$, 4, and 5, respectively, for both regular sampling and uniformly distributed random sampling for a range of values of N. The random sampling has zero correlation from dimension to dimension and from sample to sample. The increasing advantage of independent random sampling over regular sampling as K increases is clear.

Much research has been devoted to finding ways of decreasing the Monte Carlo error even further to make the technique still more efficient. One approach has been to use variance reduction techniques [18] which are able to reduce the variance of f by transforming it to another function whose integral is the same. Another approach is to use "quasirandom" or "low-discrepancy" sampling of the space V. The use of such quasirandom sampling for numerical integration is referred to as "quasi-Monte Carlo" integration. Quasirandom sampling based on "scrambled nets" [18–22] has the property that, for "well-behaved" functions, the error becomes proportional to $N^{-3/2}\ \log^{K/2N}$, which is much less than the $1/(\sqrt{N})$ for traditional Monte Carlo integration. Section 5 will discuss quasi-Monte Carlo methods. Other approaches use "recursive stratified sampling" [23] which breaks down a Monte Carlo calculation into a series of Monte Carlo subcalculations with feedback from each stage to decide how to best continue the series. Thus, the calculation is adapted to the application. In the case of integration, the region of integration is progressively divided up into subvolumes to concentrate the sampling into regions where the variance of the function is largest or of the most interest. This makes the sampling more efficient. In the example of the multidimensional sin function considered above, there would be a higher probability of samples occurring at the edges of the function, where the rate of change is greater than in the central part. The well-known "MISER" algorithm of Press and Farrar [24] is based on this approach.

The "Vegas" Monte Carlo approach of Lepage [25] is based on "importance sampling." The idea, when used for integration, is to use the pdf of the function to determine how sampling points can be concentrated in subspaces that produce values of the function that contribute most to the integral. Vegas has been amended to incorporate stratified sampling as well as importance sampling, and it also uses a form of variance reduction in subspaces where importance sampling is inappropriate because the sampling turns out to be too sparse [26]. From these well-known references and many others on Monte Carlo integration, it is clear that much improvement in efficiency can be gained over the original Monte Carlo approach.

The ideas proposed for integration have inspired similar ideas for efficiency improvement when Monte Carlo techniques are used for simulation, and these prove to be especially valuable for VLSI circuit simulation where the dimensionality and complexity are very high.

2.2. Monte Carlo Simulation. Monte Carlo simulation is the application of Monte Carlo methods to study properties of systems having stochastic components. It uses repeated pseudorandom sampling of input variables to determine the behaviour of some physical system as characterized by a computer model. In this work, the physical system is an integrated circuit modelled by SPICE, the input variables are component values which are variable due to the uncontrollability of manufacturing effects referred to in the introduction, and the behaviour we are interested in may be viability, or otherwise, of the circuit. With repeated sampling used to simulate the fabrication of batches of nominally identical integrated circuits with the specified component variation, the estimation of the probability of a circuit being viable, that is, that it works, can be considered an estimate of the expected "yield," that is, the percentage of working circuits within a batch. The criteria that determine viability are many, including correct logical operation, the power consumption, and the propagation delay in the whole circuit or parts of it.

As argued in [18], Monte Carlo simulation can be formulated in terms of integration. The direct simulation approach provides an intuitive way of defining the problem, while transforming it into an integration formulation can be useful when studying theoretical properties of the estimators

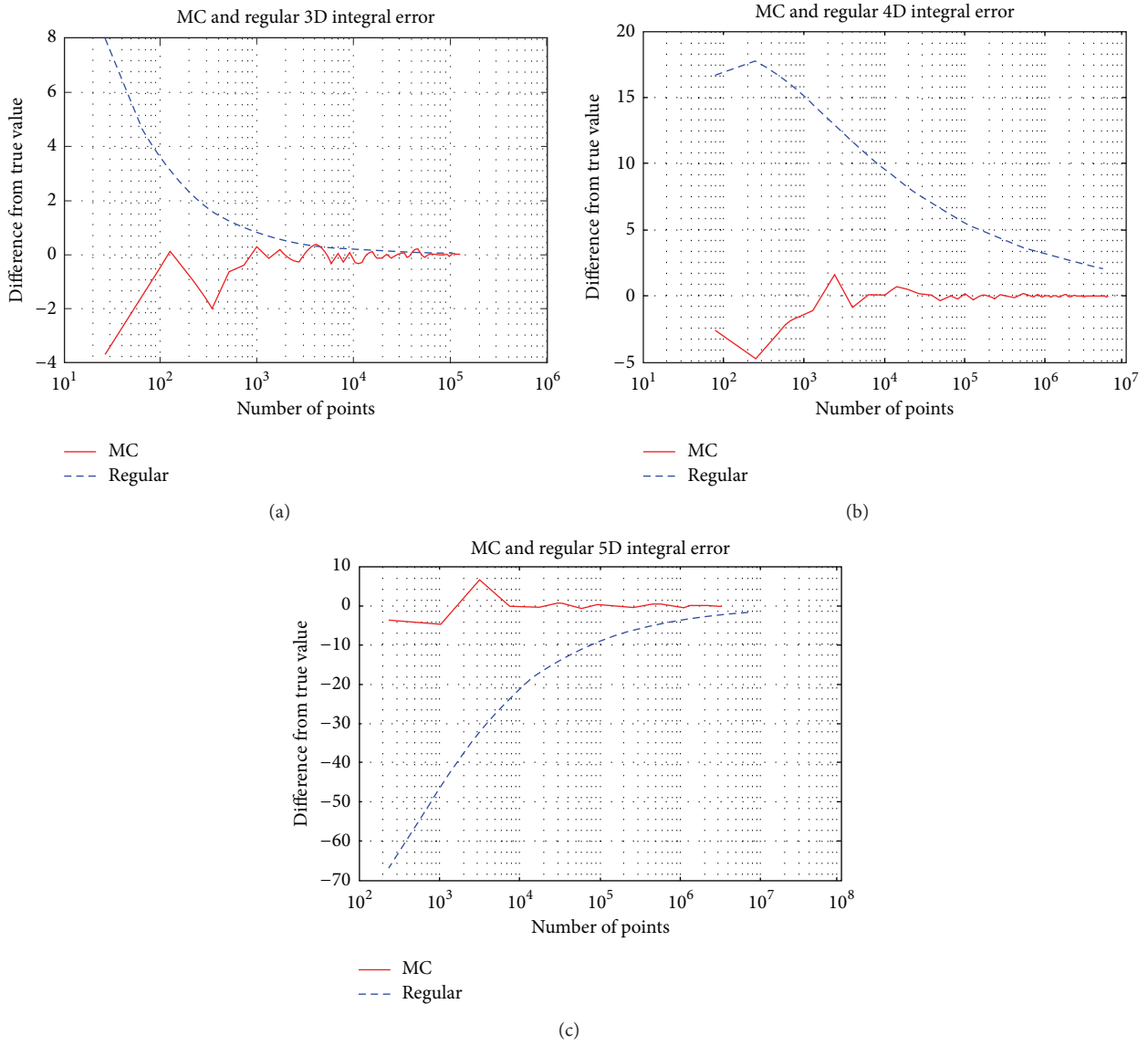


FIGURE 1: Convergence of MC and regular integration for (a) 3D integral, (b) 4D integral, and (c) 5D integral.

obtained, especially when variance reduction or quasi-Monte Carlo techniques are used. A comparison of the direct view of simulation versus an integration formulation can be summarised as follows.

The direct simulation approach samples observations of the random vector, $\underline{X}$, of inputs to and parameters of the simulated system and for each vector calculates the outputs that are of interest. We thus obtain a random distribution of each output measurement, say $f(\underline{X})$, which is of interest.

The integration formulation of simulation samples the "source of randomness" $\underline{U}$ as a vector of independent uniformly distributed random numbers. This is transformed into a vector of observations $\underline{X}$ with the appropriate multivariate distribution (e.g., Gaussian) and covariance matrix C by calculating

$$\underline{X} = Ag(\underline{U}), \tag{3}$$

where g is the appropriate "inverse cumulative distribution function" (ICDF) and A is a matrix such that

$$A^T A = C. \tag{4}$$

The covariance matrix C expresses the interdependency that exists between different elements of $\underline{X}$ and may be derived from assumptions about the effects of proximity as explained in chapter 2 of [4]. There are many ways of deriving A for a given covariance matrix C, the best known and fastest being Cholesky decomposition. The required matrix A is not unique, and an alternative is to calculate

$$A = \left[\sqrt{\lambda_1}\underline{v}_1 \quad \sqrt{\lambda_2}\underline{v}_2 \quad \sqrt{\lambda_3}\underline{v}_3 \quad \cdots \quad \sqrt{\lambda_N}\underline{v}_N\right] = V\Lambda^{1/2}, \tag{5}$$

where $\lambda_1\lambda_2\cdots\lambda_N$ are the eigenvalues and $\underline{v}_1, \underline{v}_2, \ldots, \underline{v}_N$ are the corresponding eigenvectors of C. C must be positive semidefinite to be a covariance matrix. V is the matrix of

eigenvectors and Λ is the diagonal matrix of eigenvalues as supplied by the MATLAB function:

$$[V\Lambda] = \text{eig}(C). \quad (6)$$

In comparison to the Cholesky decomposition, the eigenvalue/eigenvector method is more intuitive, and a recent paper by Keiner and Waterhouse [27] gives a more efficient way of performing the same transformation.

Calculating the distribution, between limits, of $f(Ag(\underline{U})) = h(\underline{U})$ for an output measurement of interest, say f, can now be seen as a process of integration over the range of the multidimensional vector $\underline{U}$. The integration formulation simply reexpresses the simulation in terms of input vectors, $\underline{U}$, of uncorrelated uniformly distributed pseudorandom elements which are transformed into input vectors $\underline{X}$ intended to be representative of the observations expected of true inputs and parameters. This approach has the immediate advantage of being able to model correlation between elements such as that may occur intradie on integrated circuit devices or interdie from sample to sample within a batch of integrated circuits.

The direct approach is clearly applicable when actual component or parameter measurements are available, or when they have been synthesised as for the transistor set provided by RandomSPICE [28]. The randomisation is then achieved by selecting randomly from the sets of parameters provided. Although a pseudorandom modelling process of physical effects was used to generate the RandomSPICE parameters, they could have been obtained by direct measurement of real devices. Then a model of the parameter variation is not needed since the true natural physical variation, with its inherent distribution and correlation, will be fed directly into the series of simulations.

However, even when real sets of parameters are available, instead of using them directly, it may be advantageous to produce a model of them as the transformation of a smaller set of independent uniformly distributed random variables. Then the integration formulation may be adopted, based on models derived from real data. The models may be derived by employing Principal Components Analysis (PCA) to extract a smaller set of statistically independent parameters that may be transformed back to the complete set with little distortion. The dimensionality may thus be reduced, and each of the independent parameters may be modelled as the transformation of a uniformly distributed random variable. If the independent variables may be considered Gaussian, it is straightforward to model each of them as a transformed uniform random variable. The transformation to Gaussian is achieved by the Gaussian ICDF function. Transformations to independent random variables with distributions other than Gaussian may be achieved by different ICDFs. Multivariate versions of these functions allow correlation to be introduced. With the different distributions, essentially the same methodology with respect to statistical delay estimation as used with Gaussian, Pareto, and low-discrepancy sequences in this work can remain valid.

Apart from the likely reduction in dimensionality, this transformation of a direct simulation approach to an integration-like formulation allows a much larger set of randomised devices to be generated than are available in the original set. Hence more simulations may be run with different sets of parameters based on parameters obtained by measuring real devices.

Viewed in either formulation, Monte Carlo simulation samples the probability distribution of all the input variables and system parameters to produce many repeated versions of the system. These are in turn analysed to determine how certain key output measurements vary due to the input variability. A histogram of each key output measurement gives an estimate of its likely distribution, the estimate becoming more and more reliable as the number of simulations increases. Since the number of simulations must be restricted for practical reasons, the accuracy of these results is also limited by practicality.

Where valid assumptions can be made about the shape of the distribution, for example, that it is Gaussian or Chi-squared, a maximum likelihood fit to the histogram can be made on such assumptions. Such a fit would produce a value of mean and variance thus allowing a pdf as shown in Figure 2 to be drawn. Assuming the measurement to be a delay that is required to be less than D for viability and a Gaussian distribution, the yield of the circuit can now be estimated as the area under the pdf "tail" from D to infinity which may be derived from the "complementary error function" evaluated at D.

The accuracy of this estimation will depend on the number of simulations which must be limited. Unfortunately, the effect of the limitation will be most serious over the tails of the distribution, which is the part we are most interested in. For example, for a Gaussian pdf, the probability of being more than three standard deviations greater than the mean is $Q(3) = 0.5 \times \text{erfc}(3/\sqrt{2})$, where erfc is the traditional Gaussian "complementary error function." $Q(3)$ is about 0.0013 meaning that, if 1000 circuits are simulated, we can only expect to find one value of the key output in this region. There would be a need to have many more than one value to have a chance of reasonable accuracy. This illustrates the need for complexity reduction techniques when applying Monte Carlo simulation to integrated circuit variability, and such techniques will be considered in the next sections.

In general, Monte Carlo methods proceed as follows.

(1) The characteristics of the input vectors are determined.

(2) Random vectors are generated with appropriate distributions and intercorrelation either directly or by transforming independent uniformly distributed random vectors.

(3) A deterministic computation is performed to simulate the behaviour of the system for each of the randomized input vectors.

(4) For each key output measurement, its pdf is estimated in the best way possible, given the inevitable limitations in the amount of data available.

(5) Deductions about the probability of certain events are made from the estimated pdf.

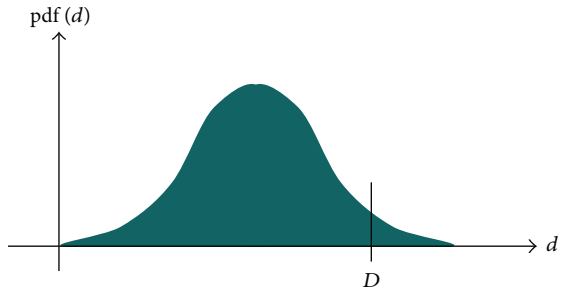


Figure 2: Gaussian probability density function of a circuit propagation delay and delay threshold D.

The procedure of an IC's Monte Carlo simulation is described as follows.

(1) Find the statistical distribution for each parameter.

(2) Sample the statistical process to produce a value for each parameter.

(3) Parameterize one circuit and simulate it.

(4) Repeat for many copies of circuit and obtain the statistical distribution of a specific measurement.

2.3. Introducing Intradie Correlation. Correlation between device parameters may be due to the intradie proximity of components, or other causes. The device parameters may be direct physical quantities, for example, resistance or capacitance, or they may be coefficients of principal component vectors which are ultimately transformed back to physical quantities. Define a matrix X_{mat} of "column" vectors $\underline{X}_j$, each containing M parameters:

$$X_{\text{mat}} = \left[\underline{X}_1 \;\; \underline{X}_2 \;\; \underline{X}_3 \;\; \cdots \;\; \underline{X}_R\right]. \tag{7}$$

Each column vector, $\underline{X}_j$, of parameters is for a different randomised circuit copy. Within X_{mat}, there is a row-to-row correlation due to interdependencies between devices or the parameters of individual devices. To model intradie correlation between parameters due to proximity, let parameter i be defined for a point $P_i = (a_i, b_i)$ on the chip. Many models of intradie correlation due to proximity have been proposed, including the widely known Pelgrom model [4] and two simpler approaches: "the exponential model" and the "Matern model," as analysed and compared by Hargreaves et al. [5]. The "exponential model" represents the correlation $k(i, m)$ between parameters i and m as

$$k(i,m) = e^{-\lambda d(P_i, P_m)}, \tag{8}$$

where $d(P_i, P_m)$ is the Euclidean distance between any two points P_i and P_m, in units of nanometers. This is assumed to be the Pearson correlation coefficient defined as

$$k(i,m) = \frac{\text{cov}\left(\text{row}_i, \text{row}_m\right)}{\sigma_i \sigma_m}, \tag{9}$$

where σ_i and σ_m are standard deviations of parameters i and m, respectively, "cov" means covariance, and the rows are those of X_{mat}. A value for λ may be calculated from measurements of actual device parameters, or manufacturer's data. Clearly $k(i, i) = 1$, and if it may be assumed that at a distance of μ nm the correlation reduces to some small value, say α,

$$\exp\left(-\lambda \cdot \mu\right) = \alpha, \tag{10}$$

it follows that

$$\lambda = -\frac{1}{\mu} \ln\left(\alpha\right). \tag{11}$$

Assuming $\alpha = 0.01$ and $\mu = 200$, we find that $\lambda = 0.023$. Therefore, if we have a location $P_i = (a_i, b_i)$ on the chip for each element i, a value of correlation $k(i, m)$ between rows i and m may be modelled. Hence, we obtain a "row-to-row" (parameter-to-parameter) correlation matrix:

$$K = \begin{bmatrix} 1 & k(1,2) & \cdots & k(1,R) \\ k(2,1) & 1 & \cdots & k(2,R) \\ \vdots & \vdots & \ddots & \vdots \\ k(R,1) & k(R,2) & \cdots & 1 \end{bmatrix}. \tag{12}$$

A similar matrix may be constructed for the more complex Matern model which was found [1] to be more accurate. To provide an example for the simpler model, assume that there are four devices at P_1, P_2, P_3, and P_4, each with a parameter whose mean and standard deviation are m_1, m_2, m_3, m_4 and s_1, s_2, s_3, s_4, respectively. Let

$$\underline{m} = \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \end{bmatrix}, \qquad \underline{s} = \begin{bmatrix} s_1 \\ s_2 \\ s_3 \\ s_4 \end{bmatrix}. \tag{13}$$

We need to generate, for each Monte Carlo randomised circuit j, random variables $x(1, j)$, $x(2, j)$, $x(3, j)$, and $x(4, j)$ as the values of the four parameters to obtain a parameter vector $\underline{X}j$. To generate a pseudorandom variable for each device with the right amount of intradie correlation, we first need to deduce, from K, an appropriate row-to-row covariance matrix, which is as follows:

$$C = \left[c(i,m)\right], \quad \text{where } c(i,m) = k(i,m) \times s_i \times s_m. \tag{14}$$

Then find matrix A such that $A^T \cdot A = C$ using the "Cholesky Decomposition" or the "eigenvector-eigenvalue method" [27], which calculates A with (5).

Finally, generate a multivariate (4-element) correctly correlated random vector as follows:

$$\underline{X}_j = \underline{m} + A \cdot \underline{r}, \tag{15}$$

where $\underline{r}$ is a 4 by 1 vector of independent random or pseudorandom variables. This is generated from a vector $\underline{u}$ of uniformly distributed random or pseudorandom variables in the range 0 to 1 as follows:

$$\underline{r} = \phi^{-1}\left(\underline{u}\right), \tag{16}$$

where ϕ^{-1} is the ICDF corresponding to the required pdf for the elements of $\underline{r}$. We can do this repeatedly for each randomised circuit j, producing many column vectors $\underline{X}_j$.

Introducing intradie variability into the multiple parameters of transistor devices is possible using the same approach as used for component values. In such applications, it is useful to partition a large correlation matrix into a number of smaller ones, each catering for one type of parameter or principal component. This is possible when there can be assumed no correlation between the different types or principal components. This is guaranteed for principal components. Clearly, a different value of λ can be used for each partition.

The example below demonstrates the effect of correlation on statistical performance analysis of a binary full adder with behavioural models of NAND gates. Consider the binary full adder shown in Figure 3, whose on-chip layout is assumed to be as shown in Figure 4.

With one delay parameter for each behavioural NAND gate [8, 9], as will be described in Section 3, there are nine parameters in total. Hence, there are just nine statistically varying parameters. Assuming the exponential model of intradie correlation with $\lambda = 0.007$, the parameter-parameter correlation matrix K is found to be as in (17).

Knowing the standard deviations of the parameters allows K to be converted to a parameter-to-parameter covariance matrix C using (14). Matrix A may be computed such that $A^T \cdot A = C$, and finally the vectors $\underline{X}_j$ of 500 random circuits may be generated and analysed using NGSPICE [14] to produce the histogram shown in Figure 5. The mean is 8.64 and the standard deviation is 0.345 picoseconds (ps).

Consider

$$K = \begin{bmatrix} 1 & 0.46 & 0.46 & 0.25 & 0.12 & 0.06 & 0.06 & 0.03 & 0.03 \\ 0.46 & 1 & 0.5 & 0.46 & 0.25 & 0.12 & 0.11 & 0.06 & 0.06 \\ 0.46 & 0.5 & 1 & 0.46 & 0.21 & 0.11 & 0.12 & 0.06 & 0.06 \\ 0.25 & 0.46 & 0.46 & 1 & 0.46 & 0.24 & 0.24 & 0.12 & 0.12 \\ 0.12 & 0.25 & 0.21 & 0.46 & 1 & 0.5 & 0.37 & 0.21 & 0.25 \\ 0.06 & 0.12 & 0.11 & 0.24 & 0.5 & 1 & 0.5 & 0.37 & 0.5 \\ 0.06 & 0.11 & 0.12 & 0.24 & 0.37 & 0.5 & 1 & 0.5 & 0.37 \\ 0.03 & 0.06 & 0.06 & 0.12 & 0.21 & 0.37 & 0.5 & 1 & 0.5 \\ 0.03 & 0.06 & 0.06 & 0.12 & 0.25 & 0.5 & 0.37 & 0.5 & 1 \end{bmatrix}. \quad (17)$$

Repeating the same procedure with $\lambda = 10$ (very low correlation) and 500 training circuits produces the histogram shown in Figure 6, whose mean is 8.647 ps and standard deviation is 0.318 ps. In comparison to the correlated case where $\lambda = 0.007$, the mean remains approximately the same, but the standard deviation reduces from 0.345 ps to 0.318 ps, that is, by about 7.8%.

3. Dimension Reduction of MC Circuit Simulation

The assertion that the error resulting from MC analyses is proportional only to the square root of the sample size does not mean that the same sample size is appropriate to any circuit no matter how complicated it is. There are

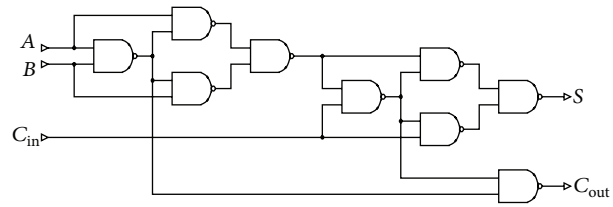


Figure 3: Binary full adder (BFA) circuit.

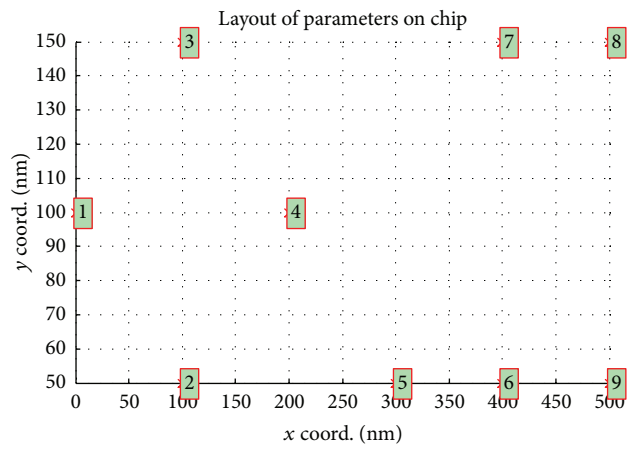


Figure 4: On-chip layout assumed for BFA circuit.

clear advantages in reducing the dimensionality of analysis problems including the simplification of the computational complexity of the analyses. This chapter deals with two methods of reducing the dimensionality of Monte Carlo analysis. The first is Principal Components Analysis (PCA) which transforms the random variables required to characterize a circuit to a reduced number of statistically independent variables. PCA is also useful as a means of introducing intradie and interdie correlations. The second is the use of statistical behavioural circuit blocks (SBCB) which substitute functional but computationally simpler circuit models for device-level analogue subcircuits. The aim of this research is to apply them in new ways which may be suitable for inclusion in the ongoing NGSPICE open-source project [14].

3.1. Principal Component Analysis (PCA). PCA is a technique for transforming samples of M variables into samples of a smaller number of variables by exploiting interdependency or correlation among the M variables. Referring to a set of R parameter vectors ("feature vectors") in an M dimensional vector space, the transformation converts each parameter vector to a lower dimensional vector in such a way that no information, or little information, is lost. This means that the original vectors, $\underline{X}_i$ for $i = 1, \ldots, R$, can be reconstructed from the transformed vectors either exactly or with a small error. The elements of the transformed (reduced dimensional) vectors are the coefficients (or "loadings") of "principal component" (PC) vectors. PC vectors are eigenvectors of the $M \times M$ covariance matrix C which are normalized and statistically independent.

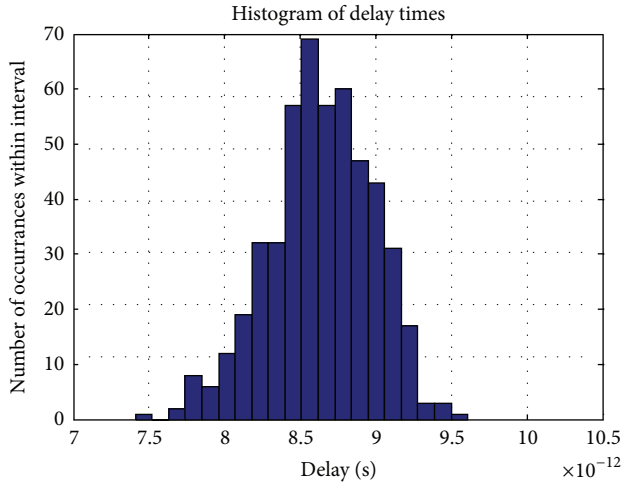


Figure 5: Histogram of delay times for 500 BFA circuits ($\lambda = 0.007$).

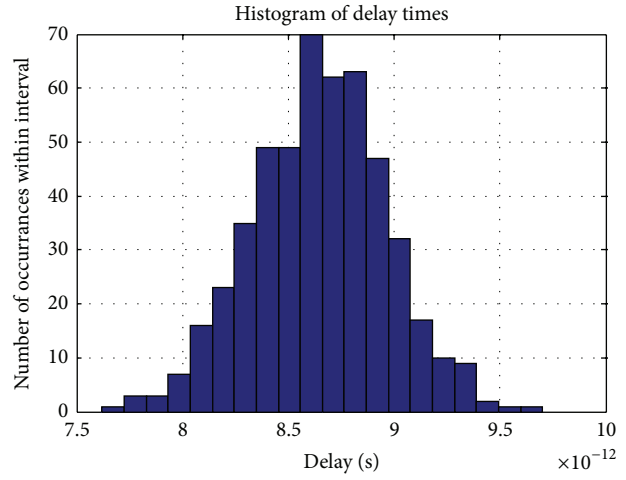


Figure 6: Histogram of delay times for 500 BFA circuits ($\lambda = 10$).

3.1.1. Performing the Analysis. Taking the summed squared differences between the elements of an original component ("feature") vector and a reconstructed one as a measure or the error or loss of information incurred by PCA, of all possible linear transformations to a lower dimensional space, PCA is optimal in minimising this error over all vectors. Further, the PC vectors are conveniently ordered in the sense that the first one has the highest variance and accounts for as much variability as possible. Each succeeding PC vector has lower variance, and therefore less importance, but has the highest variance possible while being uncorrelated to all the previous ones.

PCA may be carried out by eigenvalue/eigenvector decomposition of the M by M covariance matrix of the original vectors data matrix which is defined as

$$C = \left[c_{ij}\right], \qquad \text{where } c_{ij} = \frac{1}{R-1}\sum_{k=1}^{R}\left(x_{ik} - m_i\right)\left(x_{jk} - m_j\right). \tag{18}$$

In this expression, x_{ik} is the ith element of vector $\underline{X}_k$ with m_i denoting the mean of x_{ik} for each variable i over all R vectors, that is, the mean of row i of matrix $[x_{ik}]$ for $k = 1, 2, \ldots, R$. Therefore, C is the covariance matrix of variations from the mean of each parameter and is unaffected by the means themselves. The means of columns are not subtracted from columns. If the eigenvectors of C are $\underline{u}_1, \underline{u}_2, \ldots, \underline{u}_M$, with corresponding eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_M$, these are easily calculated by the MATLAB statement "$[U, D] = \text{eig}(C)$." This statement produces the matrix

$$U = \left[\underline{u}_1\ \underline{u}_2 \ldots\ \underline{u}_M\right] \tag{19}$$

composed of all the eigenvectors as columns and matrix D whose diagonal elements are $\lambda_1, \lambda_2, \ldots, \lambda_M$ with all other elements being zero. It follows from the definitions of eigenvectors and eigenvalues ($C\underline{u}_k = \lambda_k \underline{u}_k$ for all k) that

$$CU = UD \ \therefore\ U^T C U = D, \tag{20}$$

since eigenvectors are all orthogonal to each other and normal (i.e., of unit length) meaning that U^T is always the inverse of U. Therefore, $C = UDU^T$, meaning that

$$C = \lambda_1\left(\underline{u}_1\underline{u}_1^T\right) + \lambda_2\left(\underline{u}_2\underline{u}_2^T\right) + \cdots + \lambda_M\left(\underline{u}_M\underline{u}_M^T\right). \tag{21}$$

If matrix X is transformed to Y as follows:

$$Y = U^T \cdot\ X, \tag{22}$$

then

$$X = U \cdot Y. \tag{23}$$

If any of the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_M$ are zero and we remove their corresponding eigenvectors and corresponding rows of Y in (22), the remaining eigenvectors are principal components which can represent all the original data, without any loss of accuracy. Equation (23) will reconstruct the original vector exactly from the nonsquare matrix Y of reduced dimensional PC coefficient (loading) vectors. Taking this idea further, we can remove elements of Y with their eigenvectors when the corresponding eigenvalues are small on the grounds that, if they do not significantly affect C, they should not significantly affect X either. It may be shown that C will always be a positive semidefinite symmetric matrix (i.e., positive definite or singular); therefore, all eigenvalues will be real and positive or zero.

The previous outline of PCA hides the obvious difficulty of deciding what error is incurred by removing components with nonzero eigenvalues which are considered small and how small an eigenvalue must be to be considered negligible. Such considerations of PCA are application specific and best related to the specific objective and how the error is to be quantified.

3.1.2. Application of PCA for Modelling Statistical Variation. Assume that we have a database of sets of randomised device parameters which have been published by a manufacturer or generated on the basis of theoretical modelling. Taking the

RandomSPICE [28] Toshiba NMOS database as an example, for each device there is a set of 201 randomised parameter lists each with 300 parameters. Many of the parameters are zero or fixed, but in general they need not be. We compute the 300 by 300 element covariance matrix C for the parameter sets after subtracting the mean for each parameter. The 300 eigenvectors and eigenvalues of C are calculated and then the original mean-subtracted data is transformed by projecting each vector onto the set of 300 eigenvectors. The first ten ordered eigenvalues are plotted in Figure 7 and are seen to fall off rapidly in magnitude after number six. It is therefore reasonable to delete all but the first six eigenvalues and eigenvectors to obtain the required reduced dimensional PC coefficient vectors. Figure 8 shows how the mean squared difference between original (mean subtracted) data and reconstructed data varies with the number of eigenvectors. This shows how the approximation improves with the number of eigenvectors and that the mean squared error falls to around 10^{-20} when this number reaches about six. The modelling technique used by Glasgow University [29] explains this result. Similar results and graphs were obtained for the PMOS data provided by RandomSPICE. PCA analysis often needs some consideration of numerical stability because of the wide variation of parameter values.

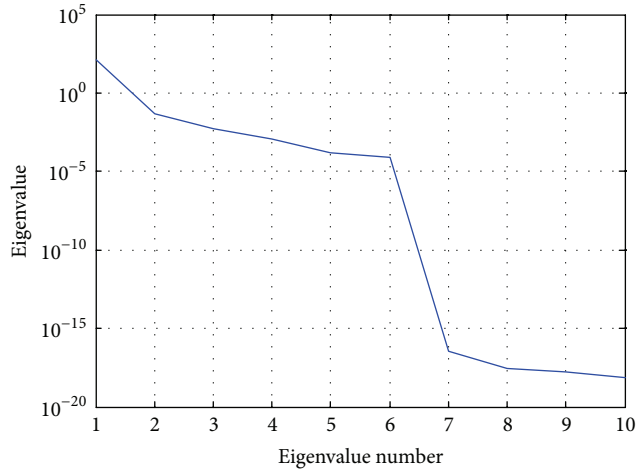


Figure 7: Values of first ten ordered eigenvalues for Toshiba NMOS data.

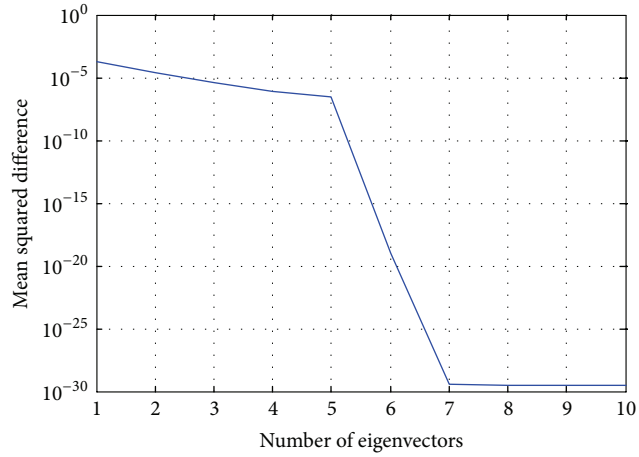


Figure 8: Mean square difference between original (mean-subtracted) data and PCA approximated data as the number of eigenvectors increases.

3.1.3. Application of PCA to Reduce Dimensionality in MC Simulation. PCA clearly has value in reducing the dimensionality of the randomisation required for MC analysis. It is now possible to randomize the principal component coefficient vectors rather than the complete list of device parameters, and then transform these back to a set of parameters for each device. An independent randomisation may clearly be performed for each device, where the effects of interdie or intradie correlations are not required to be modeled.

However, the modelling of intradie variability is afforded in a convenient way by the use of PC coefficient vectors, and the correlation introduced by proximity on the die can be conveniently applied to these. Interdie variability can also be introduced in this way. The approach is to determine a set of PCs for each device model and then to introduce correlation into the corresponding PC coefficient vectors for each device within a circuit or subcircuit, according to the exponential model outlined previously. The correlation matrix should be partitioned into a number of smaller ones, each catering for one principal component which will be independent of all the others. If the device model has N PCs, there will be N partitions, one for each PC. A different value of λ can be used for each partition.

3.2. Statistical Behavioural Circuit Blocks (SBCB). A statistical behavioural circuit block (SBCB) is a behavioural model of a device such as a transistor, or a circuit building block such as a gate or an adder. Such a block may be based on a combination of a look-up table and a simplified passive circuit. Its purpose is to model the most important aspects of the device's or circuit building block's behaviour, to an acceptable accuracy, with a relatively small number of parameters. An SBCB may be defined as a dependent voltage or current source combined with a linear time-invariant "tau" circuit. Such a combination can be optimised to match a required delay and switching waveshape, and this approach has been found to eliminate difficulties with the time-step adjustment algorithm that is sometimes encountered when using MC analysis with versions of SPICE.

3.2.1. Look-Up Tables and Tau Modelling. SPICE provides versatile dependent voltage and current sources [30] which can model switching behaviour which is a linear or nonlinear function of controlling-node voltages, branch currents, or time. These elements have many possible functions including "behavioural" voltage or current sources [30] and ideal delay elements. Ideal delay elements could be very useful in behavioural simulation, but their use for MC analysis raises problems which will be discussed later. A 2-input NAND gate ($X = A$ NAND B) using an "e-element" look-up table to

Table 1: Netlist for 2-input NAND gate using e-element.

e X 0 NAND (2) A 0 B 0	
+0.0	5.0 v
+0.5	4.8 v
+1.0	4.5 v
+4.0	0.5 v
+5.0	0.0 v

model ideal switching behaviour is represented by the SPICE netlist in Table 1. Figure 9 shows the response of this NAND gate to a voltage pulse with finite (10 ns) rise time and fall time.

The use of such ideal voltage-dependent elements provides a good way to build up behavioural models suitable for augmenting with delay modelling for statistical timing analysis. It is also useful to employ voltage-controlled resistors to implement a switch-level MOSFET.

The tau model of a transistor has long been used as a simple behavioural model in many transistor optimisation tools for designing integrated circuits, such as TILOS [31], COP [32], and EPOXY [33]. It is commonly used for both synchronous and asynchronous circuits [34]. In its simplest form, each transistor is modelled as an ideal switch with appropriate on/off resistance and source, gate, and drain capacitance introduced by discrete capacitors each being determined from the gate dimensions. The gate delays between inputs A and B and the output X of a CMOS 2-input "pull-down" circuit may be modelled [34] by the RC circuit in Figure 10.

The following expressions are obtained for $\alpha_{\uparrow AX}$ and $\alpha_{\uparrow BX}$ which are the time constants in the response to rising A and B transitions, respectively:

$$\alpha_{\uparrow AX} = R_1 C_1 + (R_1 + R_2) C_2,$$
$$\alpha_{\uparrow BX} = (R_1 + R_2) C_2. \tag{24}$$

Look-up tables, as exemplified in Table 1, and "tau models" as in Figure 10 are very simple but individually have limitations. Look-up tables can model limiting and nonlinear "slew rates" but not delay, whereas tau models are either restrictive if simple versions are used or complicated with more accurate ones. Time constants are achieved with characteristic exponentially rising or falling waveforms that may not be close to the true switching waveforms. However, combining these two techniques produces better behavioural models that are still quite simple. The tau model is used to produce the required delay, and the look-up table then modifies the waveshape with appropriately chosen entries.

If the output of a tau model is applied to the look-up table-defined e-element in Table 1, the result will be an output waveform affected both by delay and waveshaping according to the look-up table.

A curve fitting procedure can optimize the tau-model elements and the look-up table for the true waveshape produced by the device for which a behavioural model is required.

This "tau and delay" SBCB model may be used for statistical static timing analysis. In some ways, it is similar to

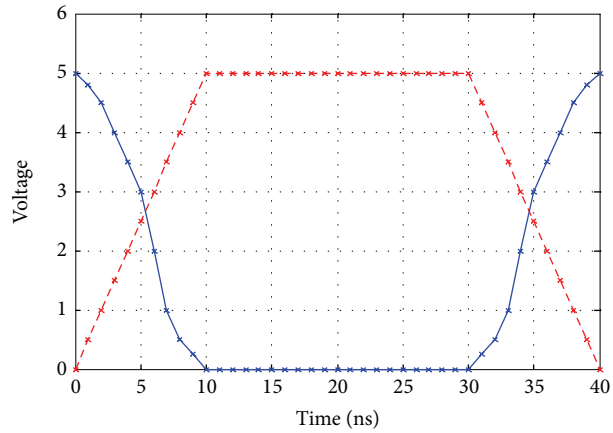


Figure 9: Response of 2-input NAND as defined in Table 1.

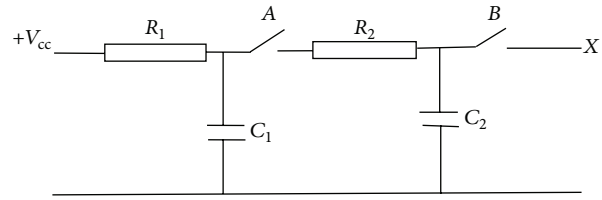


Figure 10: Tau model for CMOS "pull down" sub-circuit.

the "composite current source" (CCS) modelling technique produced by Synopsys [35] and the "effective current source model" (ECSM) provided by CADENCE Design Systems Ltd [36]. CCS and ECSM use similar modelling techniques and are supported by comprehensive libraries of cell models and interconnect models [37], which are based on industry-supplied data. Some details of these models are commercial and not openly available. Similar approaches are the "blade and razor" model [38] and a patent by Cadence Design Systems [39]. There have been many other publications inspired by the original idea of using controlled current sources with passive delay circuits to model the behaviour of interconnection and cells with respect to their timing delay, noise generation, power consumption, and statistical variability characteristics [40–43].

The "tau and delay" modelling approach proposed previously uses voltage-controlled voltage sources (VCVS) rather than voltage- or time-dependent current sources. The choice was made arbitrarily since we did utilise any existing libraries, though the dominance of interconnection delay in submicron technologies offers some justification for our approach. It is argued [42] that nonlinearised Thevenin or voltage-source-based driver cell models cannot easily be adapted to accurately modelling the effect of loads with highly nonlinear characteristics. Current source modelling has therefore been generally preferred for modelling the nonlinear aspects of the input-output relationships of cells and their interconnections. However, it is often pointed out (e.g., [35]) that VCCS-based models can easily be converted to VCVS-based models and vice versa.

Our SBCS modelling approach was used to produce a single model of each cell type. Each SBCS is characterised by parameters which may be randomised to simulate the effect of statistical circuit variation. The statistical characteristics of the randomisation (distribution, mean, and standard deviation) were derived from transistor-level simulations of true devices. In principal, a different look-up table is needed for each value of delay, but the required adjustments were found to be very small.

3.2.2. Behavioural Models for MC Simulation. SPICE offers a large number of different ways of defining and randomising behavioural models, and we have suggested yet another alternative. The most sophisticated version of SPICE, HSPICE, does not often specify how certain features are implemented, and the open-sourced NGSPICE, though based on the same long-established analysis engine, does not have many of these features. All dependent sources in HSPICE have "ideal" delay as an option, which is trivial to achieve in digital circuits but unachievable exactly in analogue circuits. Modelling ideal delay in analogue circuits can cause difficulties with MC analysis since randomising such delays can cause SPICE to make increasingly slow progress and eventually to "hang." Investigations revealed that, when this happened, it was not the models, but the step-size selection that was to blame. If ideal delays are specified with high numerical precision, the relative timing of events on a single chip can appear to vary almost continuously. Coincident switching events when randomised may become different but very close together. The time-step adaptation algorithm will try to model the very small timing differences and thus generate exceedingly small time steps. Quantizing the Monte Carlo variation eliminates this problem which means that the randomisation should ideally be done with reference to the anticipated time-step size. However, the effect on the results of the statistical analysis must then be investigated.

The modified tau-modelling approach has been adopted for the statistical behavioural circuit blocks to be described in the next subsection. Matching MC randomisation of behavioural model parameters to the step-adaptation algorithm of SPICE is a matter deserving further investigation as there may be great economies and insights to be gained.

3.2.3. Building Up SBCBs. SBCBs are used to replace true or accurately modelled subblocks to reduce the dimensionality and therefore the complexity of the analysis. By analysing a representative and random sample of fabricated gates, a delay distribution may be obtained whose statistical properties (pdf, mean, variance, etc.) can be used to define the SBCB.

4. Computation Reduction by Extreme Value Theory

Extreme value theory (EVT) [44] is a branch of statistics concerned with the estimation of probabilities which are "extreme" in the sense that the range of values of interest is many standard deviations from the mean of an assumed probability distribution. An algorithm known as "statistical blockade" (SB) [6, 7] applies EVT to circuit analysis by eliminating or "blocking out" randomised parameter vectors that are considered unlikely to produce circuits that fall in the low-probability tails. The intention is that only the ones likely to produce "rare events" are analysed. In our application, the rare events are the circuit yield failure predictions which are extreme in the sense that they are on the "tails" of Gaussian-like probability distributions for circuit quantities such as overall delay. Because they are designed to be rare, reliable estimates of these failures by conventional MC techniques require very large numbers of randomised input vectors.

The idea of SB is to try to concentrate on parameter vectors that are likely to generate the "rare events" of failing circuits and block out or disregard the ones that are unlikely to produce such failing circuits. Many input vectors are generated, but only the ones likely to produce "rare events" are simulated. This partial sampling of the performance distributions is the basis of EVT. The computational complexity involved in introducing the bias and compensating for it is much less expensive than performing many uninteresting circuit simulations. The "blockade filter" is a standard classifier used in machine learning and data mining. It is trained by simulating a relatively small "training set" of randomized circuits and is further refined as more and more simulations are carried out. Statistical blockade with this recursive updating is intended to make estimation of rare event statistics computationally feasible.

The implementation of SB is initiated by a "seed" netlist which specifies the basic circuit with its SBCB blocks. This netlist also specifies which parameters are to be randomized and the statistics (mean, standard deviation, etc.) of the required randomisation. It can be divided into four parts.

(i) The training of an estimator for predicting circuit performance with minimal computation. This requires a sufficiently large training set of randomized circuits to be generated and analysed by SPICE. In our work, the coefficients of a linear estimator are calculated using the "pseudoinverse" approach, and there must be many more randomized circuits than parameters. The coefficients are computed to make the estimator minimise the sum of the squared differences between the estimated circuit output measurements and the "true" circuit output measurement, as obtained from SPICE, over the whole set of training circuits.

(ii) The generation of a much larger set of randomized versions of the circuit, and the use of a classifier to "block" the versions that are not likely to be within the tail. The classifier consists of the "linear estimator" followed by a "threshold" comparison with a "start of tail" parameter. Only the circuit copies which are estimated to fall within the tail will be unblocked and submitted for analysis by simulation.

(iii) The "recursive" refinements of the linear estimator as more and more simulations are carried out. When a sufficient number of nonblocked "tail" circuits have been analysed, a second estimator is calculated using the "pseudoinverse" technique. The second estimator is more accurate than the original estimator for the

tail and may be used for more accurate blocking. The use of recursion can move the defined "start of tail" parameter further away from the mean: typically from two to 3 or 4 standard deviations. Through recursion, we can thus get more accuracy in more extreme parts of the tail.

(iv) The fitting of a Pareto Distribution (PD) to the measurements obtained from the nonblocked ("statistical tail") versions of the circuit. This is necessary because, with SB, the non-tail circuits are blocked (not analysed) so we can no longer use Gaussian statistics. Also, very few measurements will occur in the "far tail" even when large numbers of circuits are generated. The use of PD fitting to the rarely occurring "tail circuits" allows the prediction of likely yield without the very large number of circuit simulations that would be required with traditional MC analysis.

To illustrate the computation time savings that may be achieved when synchronous and asynchronous circuits employing SBCB blocks are statistically analysed by MC techniques with SB, a frequently used handshaking component in the asynchronous control circuits produced by the BALSA design package [45], that is, a C-element [46], was considered. The intention was to compare the speed and accuracy achievable with that of straightforward MC analysis. A binary full adder, using NAND gates as building blocks, and a 4-Phase Bundled Data Muller pipeline and a Muller "ring," each using the C-element as building blocks, were also used as test circuits. The use of SB to analyse the switching delay of the output of a single C-element was found to reduce the computation time by about 98.5%, when the start of the distribution tail was defined to be two standard deviations (2σ) from the mean.

The accuracy of the linear estimator obtained with the start of tail defined 2σ from the mean is illustrated by the scatter graph in Figure 11. It was obtained by applying SB to 1000 copies of the binary full adder circuit. The blue points represent the measurements which are in the tail and not blocked.

To obtain accurate predictions of behaviour further from the mean, recursion was employed to refine the accuracy of the original estimator using the results of nonblocked simulations. Figure 12(b) shows the effect of recalculating the estimator from the tail points, shown in Figure 12(a), as identified by the original 2σ estimator. The experiment is applying recursion to the tail points obtained from blocking 1000 copies of the 4-Phase Bundled Data Muller pipeline circuit shown in Figure 13.

The estimated failure probability distributions shown in Figures 16(a) and 16(b) were obtained for the behavioural model of a single C-element, as shown in Figure 14, with parameters for each behavioural gate model extracted from transistor-level simulations referred to in Section 3.2.3. Figure 15(a) was obtained by traditional MC analysis and plots the yield failure probability against yield threshold, showing how the failure probability reduces as the permissible delay increases. The threshold value is measured in seconds relative to a reference delay 200 ps from the start

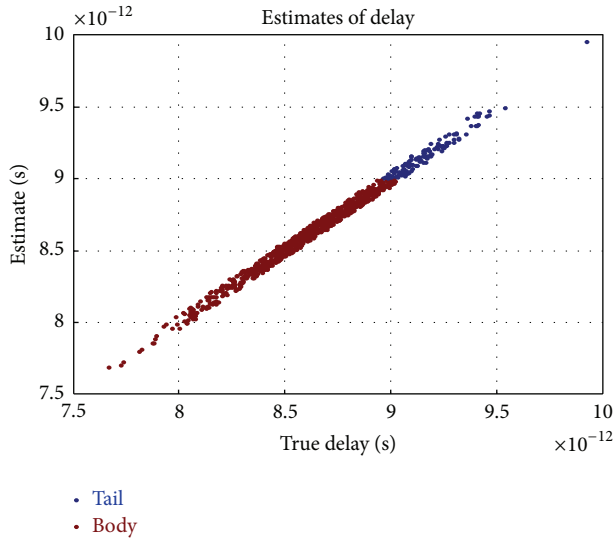
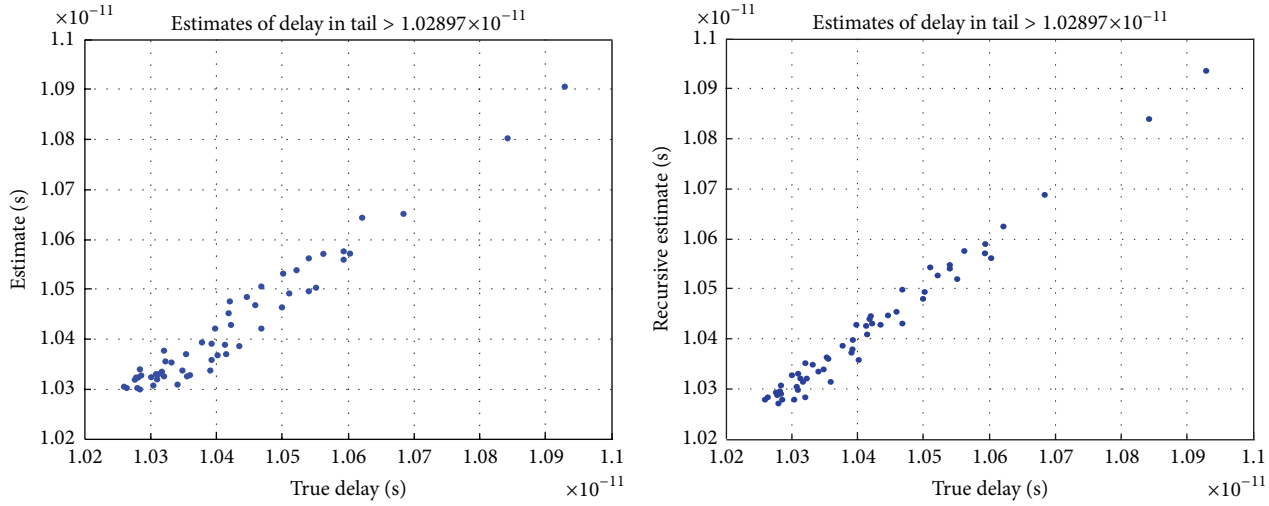


Figure 11: Accuracy of linear estimator. Tail defined to start at $9e-12s$. Classification errors out of 1000: wrong to block: 4, wrong not to block: 2.

of the analysis where the trigger occurs at 100 picoseconds from the start. The graphs show the distribution tails only, which are assumed to start at two standard deviations (i.e., 2×0.657 ps) from the mean which is 29.7 ps relative to 200 ps. Therefore, the graphs show a time scale from 31 ps (relative to 200 ps) and extending over 5 standard deviations. The graph shown in Figure 15(b) was obtained from MC analysis with SB and a Pareto fit to the tail assumed to start at two standard deviations from the mean, which was set to 31 ps. Figure 15(b) may be seen to be close to Figure 15(a) as obtained from traditional MC analysis with much greater computation. To assess the quality of fit, the two graphs are shown on the same axes in Figure 15(c).

It may be seen that the maximum difference in yield threshold delay between the two graphs for any yield failure probability is about 0.06 ps seconds, which is about 0.1 standard deviations. A more useful measure of difference is the maximum difference in yield failure probability. This cannot accurately be deduced from the graphs, but a resampling of the data plotted in one of the two graphs (since the sampling instants are different) revealed that this maximum difference occurred at a yield threshold of 31.23 ps and is equal to a probability difference of 0.002. This represents a discrepancy of approximately 14.2% from the probability 0.0129 predicted by the non-SB Monte Carlo simulation being used as a reference. Thus the maximum discrepancy in the yield failure probability is 14.2% which occurs when the yield threshold delay is 0.1 standard deviations.

Since a possible source of this discrepancy is the quality and suitability of the Pareto fit, some investigations were carried out. It was observed that one source of the discrepancy was the difference in mean and standard deviation of the Gaussian fits to the delay measurements produced on one hand by the non-SB MC simulations ("meanNB" and "sigmaNB") and on the other hand by the training procedure



(a) Original (b) Refined

Figure 12: Refining linear estimator by recursion.

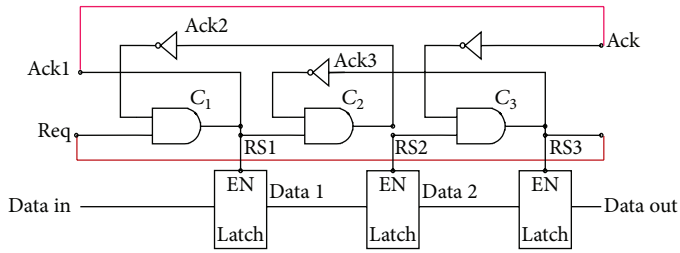


Figure 13: 4-Phase 3-stage Bundled Data Muller pipeline "ring."

("meanTR" and "sigmaTR"). These are used to determine the "start of tail" parameter at two standard deviations from the mean. The more accurate estimations "meanNB" and "sigmaNB" are available for producing the comparisons since a computationally expensive non-SB will have been carried out for test purposes. But in reality, only the less accurate "meanTR" and "sigmaTR" estimates (based on far fewer randomised circuits) will be available to the SB version and were therefore used in the comparison.

As a test, "meanTR" and "sigmaTR" were replaced by "meanNB" and "sigmaNB," thus eliminating two sources of discrepancy and allowing the suitability of the Pareto fit to be seen more clearly in Figure 15(d). It was found that the two graphs did indeed become closer, the maximum discrepancy being 0.00061 in a non-SB yield probability measurement of 0.0136, that is, a 4.5% discrepancy. A conclusion from this investigation is that the Pareto fit is capable of giving a reasonable approximation to a Gaussian tail, incurring error likely to be less than that resulting from other statistical estimates. Also, we concluded that there is scope for increased accuracy in the SB estimations, for example, by updating the estimates of mean and standard deviation as statistical analysis proceeds.

With a more accurate estimator, the "start of tail" parameter may then be redefined as two or even three standard deviations from the mean to obtain even greater time saving since even fewer circuits need to be analysed. This increases the possibility of finding measurements yet further from the mean, that is, "rarer events," in reasonable computational time and allows a yet more accurate estimation of the statistics of the "far tail."

Table 2 summarises the computation time savings that were obtained by applying SB to the statistical variability analysis of three of the circuits mentioned above. The computation was carried out on a standard desktop PC with a dual core 2.8 GHz Intel processor. A MATLAB program that harnesses an implementation of SPICE carried out the randomisation, implementation of SB, and statistical analysis. It may be seen that the most significant computation time saving, 98.7%, was achieved for a 4-Phase 3-stage Bundled Data Muller pipeline "ring," with the start of the tail defined at two standard deviations from the mean.

5. Computation Reduction by Quasi-Monte Carlo Techniques

This section investigates the use of "low-discrepancy" sampling to achieve further efficiency improvements, over what

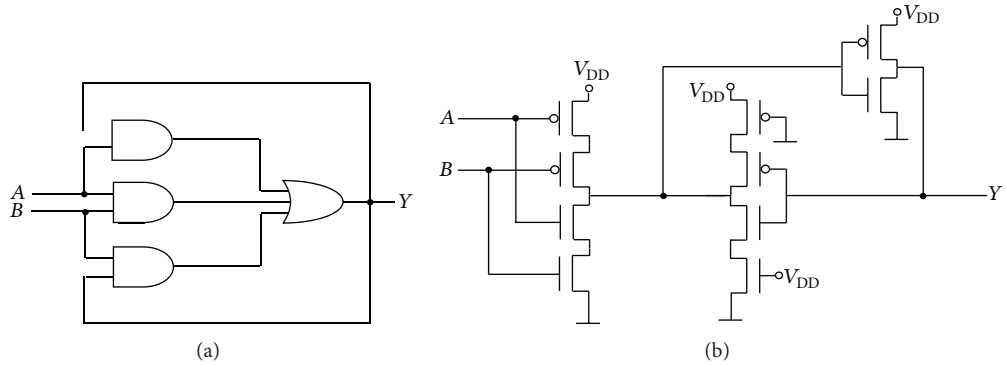


FIGURE 14: Muller C-element: (a) gate level, (b) transistor level.

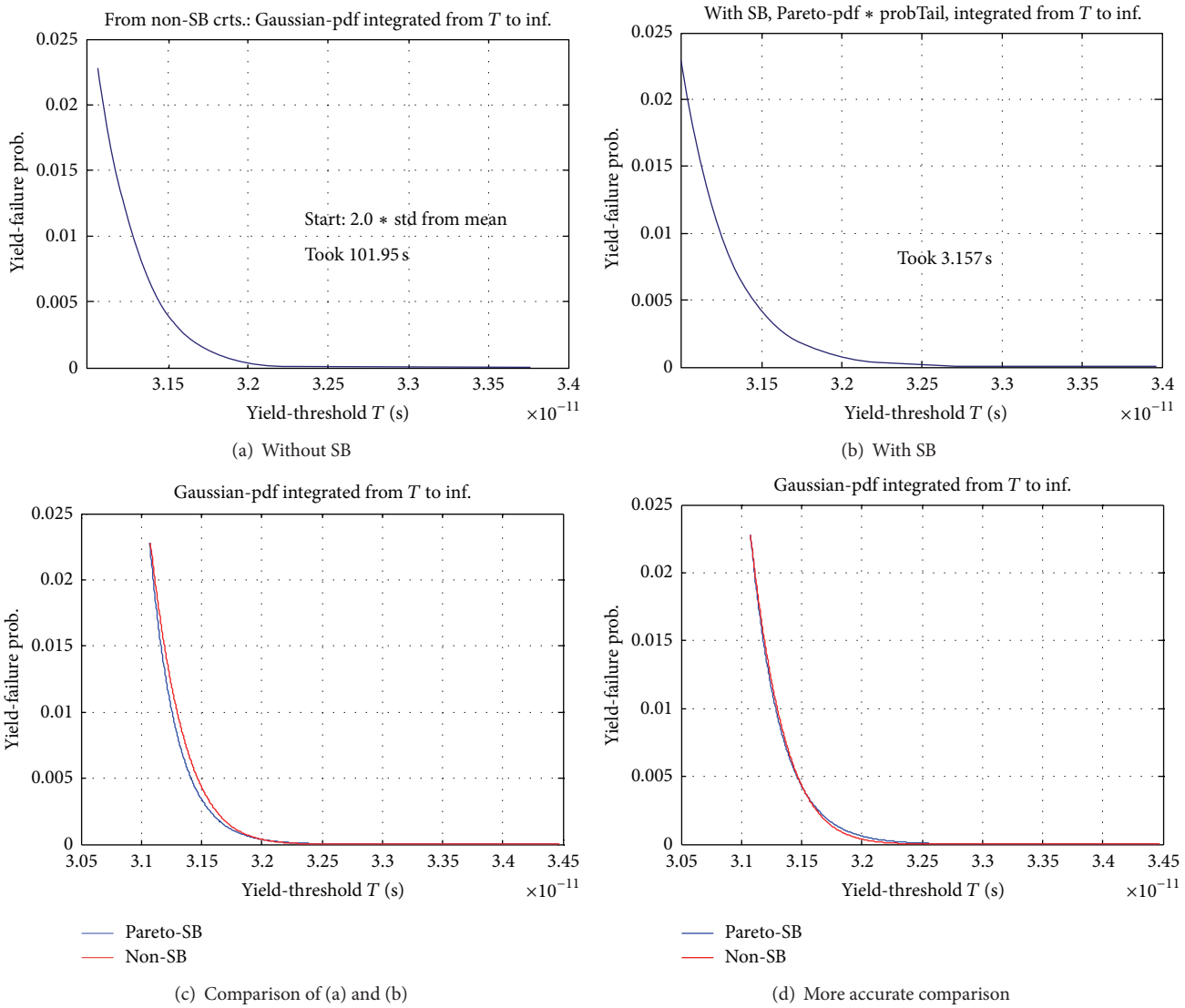


FIGURE 15: Failure probability for a "C-element" realisation from 500 versions: (a) without SB, (b) with SB (2σ from mean), (c) comparison of (a) and (b), and (d) comparison with more accurate estimates of mean and standard deviation used for Pareto-SB.

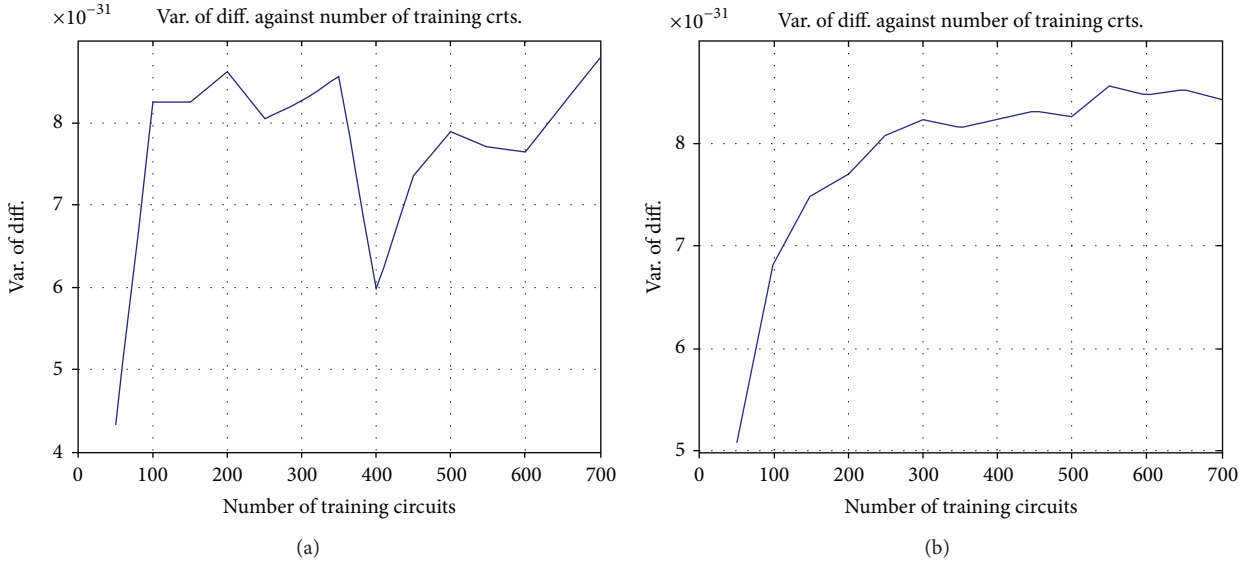


Figure 16: Error analysis of linear estimators in RandomLA training phase: (a) MC simulation and (b) QMC simulation.

Table 2: Time saving illustrated by comparing simulations with SB to simulations without SB.

Circuit	Binary full adder 9 parameters		C-element 12 parameters		Muller pipeline ring 21 parameters	
Start of tail	1.5σ	2σ	1.5σ	2σ	1.5σ	2σ
1000 circuit without SB	215.99 s	221.34 s	250.05 s	288.51 s	949.59 s	1003.9 s
1000 circuit with SB	6.75 s	3.96 s	7.63 s	4.24 s	27.17 s	13.15 s
Time saving	96.9%	98.2%	96.94%	98.5%	97.1%	98.7%

was achieved in earlier sections, with Monte Carlo circuit simulation. Low-discrepancy sampling is the basis of "quasi-Monte Carlo" (QMC) techniques as often applied to multidimensional integration; therefore, this approach to circuit simulation may be referred to as "quasi-Monte Carlo" simulation. QMC methods are modified Monte Carlo methods where the input vectors are not totally random but are to a degree deterministic in that they conform to "low-discrepancy sequences" [47, 48]. A low-discrepancy sequence is a sequence of N-dimensional vectors which covers a finite space more uniformly than is achieved by N-dimensional vectors of independent uniformly distributed random elements. The discrepancy of a sequence of vectors is a measure of how the number of points they define in a multidimensional cube varies with the position and size of the cube. If the discrepancy is low, the same-sized cube will always contain approximately the same number of points wherever it is located, and the number of points will be proportional to the volume of the cube. A sequence of vectors of independent uniformly distributed random elements does not give low discrepancy when the dimensionality is high. It is known that the use of low-discrepancy vector sequences can achieve significant speed gains over standard Monte Carlo integration techniques by reducing the number of input vectors needed for a given accuracy [49]. Similar gains are anticipated when QMC is used for statistical circuit simulation.

As mentioned in Section 2.1, traditional Monte Carlo methods with statistically independent input vectors are argued to have a convergence rate proportional to $R^{-1/2}$ which is independent of dimensionality N. Quasi-Monte Carlo (QMC) methods are claimed to have a much better rate of convergence approaching proportionality with R^{-1} in optimal cases [50]. A theoretical upper bound for the convergence rates of multivariate low-discrepancy sequences is reported to be proportional to $(\ln R)^N/R$ [5], where N is the number of dimensions. QMC performance therefore decreases with the dimensionality N.

Uniformly distributed numbers in the interval (0, 1) can be generated as pseudorandom numbers or quasirandom numbers, and the variables for all other distributions may be derived from these by means of the appropriate cumulative distribution function inversion. In practice the range must be restricted from (0, 1) to $(\alpha, 1-\alpha)$ for small positive α; otherwise, Gaussian variables very far from the mean may occur with QMC and cause numerical instability. Examining a Gaussian pdf graph reveals that taking $\alpha = 10^{-6}$ or 10^{-11} restricts the transformed variables between 5 or 7 standard deviations, respectively, from the mean.

The MATLAB functions "Haltonset" and "Sobolset" are provided for constructing initial sequences of N-dimensional quasirandom vectors with the required properties. To avoid the undesirable effects of any correlations, especially in

the initial segments, the random sequence obtained can be required to skip, leap over, or scramble values in the sequence as generated. Scrambling reduces correlations while also improving uniformity. These sequences use different prime bases to form successively finer uniform partitions of the unit interval in each dimension. Latin hyper-cube sequences, as used by SPICE, are generated by the "*lhsdesig*" function. Strictly, these are not LD sequences, but they nevertheless produce uniform samples of a sort.

As suggested in this work, the idea is to use a low-discrepancy sequence generator to replace the uniform random number generator as the source of randomisation in both the training and the recursive SB phases of RandomLA, the developed MATLAB software in the research. The choice of LDS will be "Sobol." First, we present an example that compares the effect of using QMC rather than MC for training the linear estimator. Then we investigate the effectiveness of QMC for MC simulation with and without Statistical Blockade.

To provide comparison for training, an SRAM32 × 1 array circuit was taken as an example that is described in more detail in Section 6. The convergence of the linear estimator training using MC and QMC was analysed and compared.

As shown in Figure 16, the QMC training converges more quickly and smoothly to an estimator producing the minimum "variance of difference" attainable, approximately 9×10^{-31}, than MC training with pseudorandom vectors, which is shown in Figure 16(a). The measure is just the variance of the prediction error. With QMC training, the variance becomes close to optimum with about 300 training circuits, whereas the number needed for MC is about 700 training circuits. The reason for this improved training is probably the more reliable coverage of QMC, for a given number of circuits (sample size) across the whole domain of parameters which gives us more "tail" circuits, even without the advantages of Statistical Blockade.

Figure 17 shows the yield predictions obtained by MC and QMC analysis with and without Statistical Blockade and Pareto fitting for the "binary full adder" circuit in Figure 3. There are 36 transistors, and each was randomized based on two PCA components giving 72 parameters for the "transistor-level" statistical analysis. The nonblockade analysis was carried out, with MC randomisation only, on 3000 circuits which took 3497.1 seconds. The results were taken as a benchmark for comparison with both MC-SB and QMC-SB, though a benchmark close to this could have been obtained with about 700 fewer circuits using nonblockade QMC.

Both MC and QMC Statistical Blockades were applied using 300 training circuits in both cases. The estimator order, as always, was equal to the number of parameters, that is, 72 in this case. The analysis time for recursive SB with MC and QMC was 146.6 s and 120.6 s, respectively, achieving close to 99% savings in each case with statistical variation from run to run, depending on how many circuits are blocked; it is not uncommon for QMC-SB to take longer than MC-SB when the same number of circuits is specified. Where the criterion is accuracy and reliability, QMC reduces the required sample size. For a given sample size, the advantages of QMC with SB over "non-SB" are not as striking as those of MC-SB over MC without SB. More analysis is needed on this matter. As in Section 5, the effects of the difference in mean and standard deviation produced by the non-SB simulations and the training procedure can be seen in Figure 17. If better estimates were available, the graphs would be closer. For QMC, the maximum discrepancy in yield failure estimates between SB and non-SB is a probability difference of about 0.003 or 0.3%. For MC the discrepancy was less, that is, about 0.1%.

6. Results and Evaluation with SRAM Arrays

6.1. Effect of Correlation on Failure Yield Analysis of an SRAM8 × 1 Array. The 8 × 1 SRAM array shown in Figure 18 was analysed to predict its yield, subject to a maximum delay constraint. All "Bitlines" were set to "1," the "bit" nodes were initialised to "0," and the "Wordline" inputs switched from "0" to "1" at a fixed point in time, causing the "bit" outputs to switch to "1" with random delay. A SPICE harness in MATLAB randomised the device parameters and made repeated calls to SPICE to analyse the effect of these parameter variations on the delays at the "bit" nodes.

There are 48 transistors within the circuit. Transistor-level simulations of the array were carried out to estimate the yield failure probability for different values of yield threshold when statistical variability is included in the model for each transistor. This was obtained by connecting the eight outputs from the array, all initialised to zero, to a behaviourally modelled NAND gate whose output switching delay was compared to the threshold. The statistics for the parameter variation were derived from analyses of the 35 nm transistor model data set provided by RandomSPICE. Two extreme cases were considered: firstly where there is assumed to be strong intradie cell-to-cell correlation between randomised device parameters of a particular type and secondly where there is no intradie correlation between devices from cell to cell. A graph obtained for yield failure probability against allowable delay threshold for the strong correction case is shown in Figure 19. The "vth0" parameters of all six devices within each of the eight cells were parameterised with $\lambda = 0.007$.

As estimated by the MC runs with 3000 randomised circuits, the mean delay was found to be 16.7 ps from the input pulse edge occurring 20 ps after a timing reference. The standard deviation was found to be 0.376 ps. The graph, in Figure 19, starts at the "start of tail" which is two standard deviations from the mean of the distribution, that is, at 37.5 ps from the timing reference.

The results obtained from the noncorrelation case are presented in Figure 20. The eight cells were randomised by independent variables with negligible correlation ($\lambda = 10$). The mean was observed to be 40.1 ps from the timing reference (i.e., 20 ps from the edge), and the standard deviation was 0.263 ps. There is a reduction in standard deviation from 0.376 ps for the strongly correlated case to 0.263 ps, which is a factor of 0.69. With independent Gaussian parameter

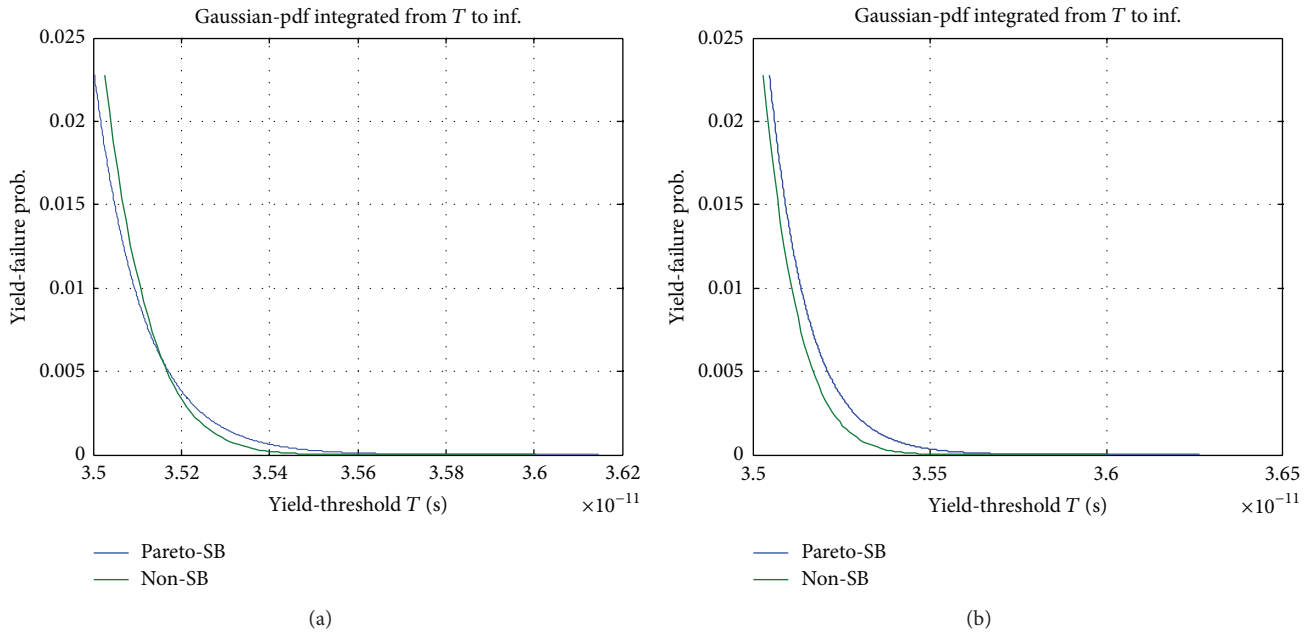


FIGURE 17: (a) MC-SB compared to MC-non-SB for BFA (3000 circuits), (b) QMC-SB compared to MC-non-SB for BFA (3000 circuits).

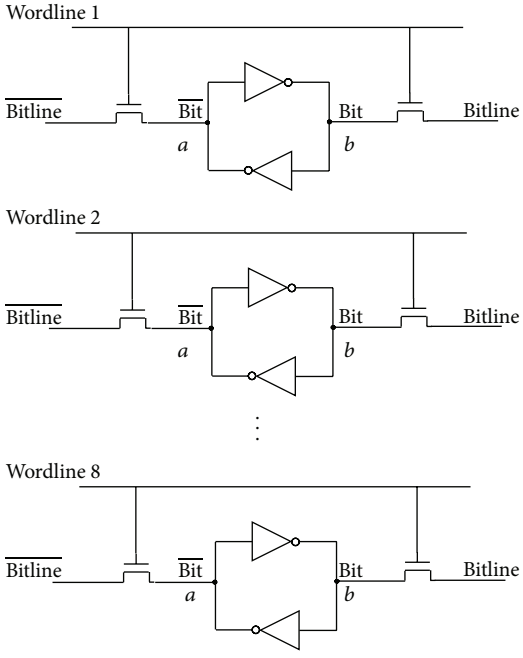


FIGURE 18: SRAM8 × 1 array circuit.

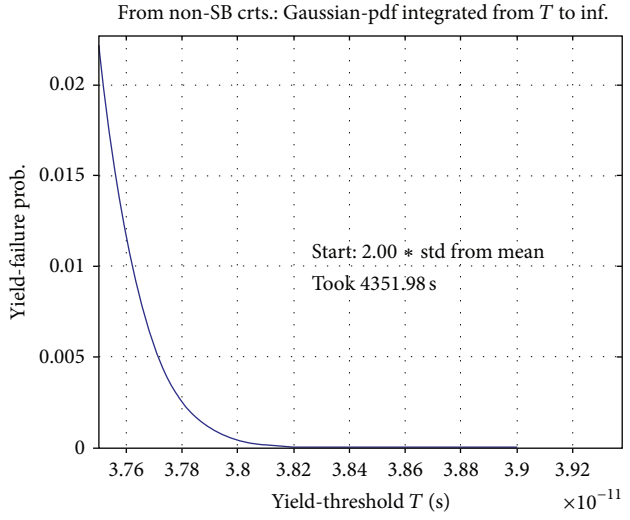


FIGURE 19: Yield obtained from 3000 transistor-level simulations of SRAM8 × 1 for strongly correlated case.

variations, the distribution of worst case cell delay which determines the yield failure probability is no longer Gaussian. It becomes a type of "Gumbel" distribution [51]. The results indicate that the standard deviation reduces by a factor of 0.69, and the mean increases by 1.436 standard deviations. Comparing the strongly correlated case (Figure 19) with the noncorrelated case (Figure 20), it can be seen that the intradie correlation can have a significant effect on predictions of yield failure probability.

6.2. Evaluation with an SRAM32×8 Array. Thirty-two copies of Figure 18 were cascaded to construct the 32×8 array shown as Figure 21. There are 1536 transistors within the circuit. The SBCB model of this array is established from the single cell model and connected together. The simulations were undertaken for the strong correlation case only.

Figures 22 and 23 present the graphs obtained from the simulations; the run times are given in Table 3. The observed delay mean and standard deviation are very close to the ones

Table 3: Run times and time savings for MC/QMC simulations of the SRAM32 × 8 array (strongly correlated).

	Transistor model: ngSRAM32 × 8.seed				SBCB model: ngswSRAM32 × 8.seed			
	MC		QMC		MC		QMC	
	Non-SB	SB	Non-SB	SB	Non-SB	SB	Non-SB	SB
CPU time/s	47263.96	1481.69 $W = 13$	39373.99	1183.59 $W = 0$	2376.14	63.70 $W = 6$	2148.03	69.84 $W = 0$
Time saving	96.87%		96.99%		97.32%		96.75%	
Overall time saving = (47263.96 − 69.84)/47263.96 = 99.85%								

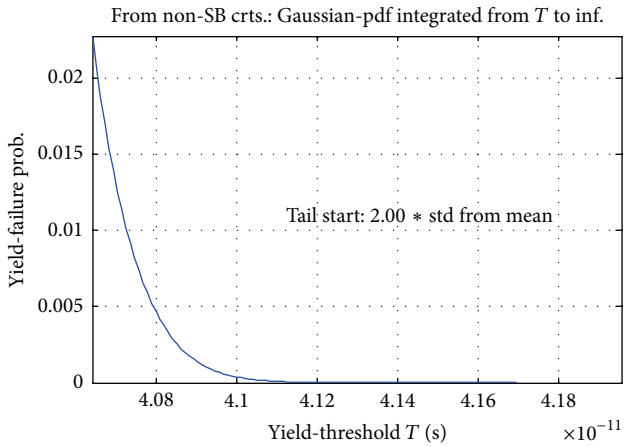


Figure 20: Yield obtained from 3000 transistor-level simulations of SRAM8 × 1 for noncorrelated case.

in Figure 19 for the SRAM8 × 1 array, which is as expected for the strongly correlated case with all cells in parallel.

Analysis of the results revealed the following.

(a) SB with Pareto fitting version is reasonably accurate and much faster in comparison to the "nonblockade" version.

(b) For the SB simulations with strong correlation, the number of wrong decisions not to block "W" was always close to zero for QMC with Sobol points, while the number for MC fluctuated between zero and about 20 when there were 3000 circuits being analysed. This is an indication of the accuracy of the linear estimator which was found to be better for "Sobol" vector training with under 200 training circuits than for MC with pseudorandom parameter vectors. The results also demonstrate that QMC with Sobol vectors can make nonblockade simulation more efficient, reaching a given accuracy with fewer runs than are required with MC.

(c) For the noncorrelation examples, the estimator was much less accurate for both MC and QMC training. This caused many more wrong decisions not to block. The results of these wrong decisions are discarded for the Pareto tail fitting procedure with some loss of efficiency. The behaviour of the linear estimator when adapting to the maximum delay criterion explains this loss of efficiency.

(d) The use of QMC with "Sobol" vectors makes nonblockade more efficient than with MC in that a given accuracy is achieved with fewer runs.

(e) QMC with SB compared with QMC alone offers further savings, but these remain to be fully analysed.

(f) When comparing overall time savings for MC with SB and QMC with SB, both the training and the analysis times must be taken into account. A lower number of training circuits were required for QMC simulations than for MC to reach a given estimator accuracy.

From the Gaussian distribution, it may be deduced that, if the delay distribution is Gaussian and the tail is assumed to start at two standard deviations from the mean, the percentage of unblocked circuits may be expected to be about 2.1%. Therefore, out of 3000 randomly generated circuits, about 63 unblocked circuits should be observed. Out of the first 500 circuits, about ten unblocked circuits should occur, and this observation suggests a simple adaptation mechanism for countering inaccuracy in the estimator. After a certain number of random circuits, say 500, have been generated, if the number of unblocked circuits is significantly different from what is expected, say 10, the tail threshold can be decreased or increased accordingly. The decision can be revisited later in the run, say after 1000 circuits, 2000 circuits, and so on. This adaptation was found to be useful in the noncorrelated examples presented in this chapter where the accuracy of the estimator was found to be lower than for the strongly correlated examples. Decreasing the threshold does not greatly affect the computation run time if the intention is to base the tail estimation on a specific number of circuits, say 2.1% of the total. This approach appears even more advantageous when higher deviations from the mean are to be examined, say three or more standard deviations. Instead of specifying a fixed number of randomised circuits, the simulations could be allowed to continue until a suitable number of unblocked circuits have been produced to allow a reliable estimation of the tail distribution.

The timing results quoted in this section are for single-core nondistributed computation. The RandomLA SPICE harness has been developed in such a way that it may be run on multicore machines and distributed frameworks such as Condor. Using parallel or distributed computing facilities can achieve great time savings. For example, rerunning the simulations in this chapter on a dual-core PC achieves a time saving which is very close to 50%, that is, a factor of two reductions in runtime. Using Condor, the run time of the

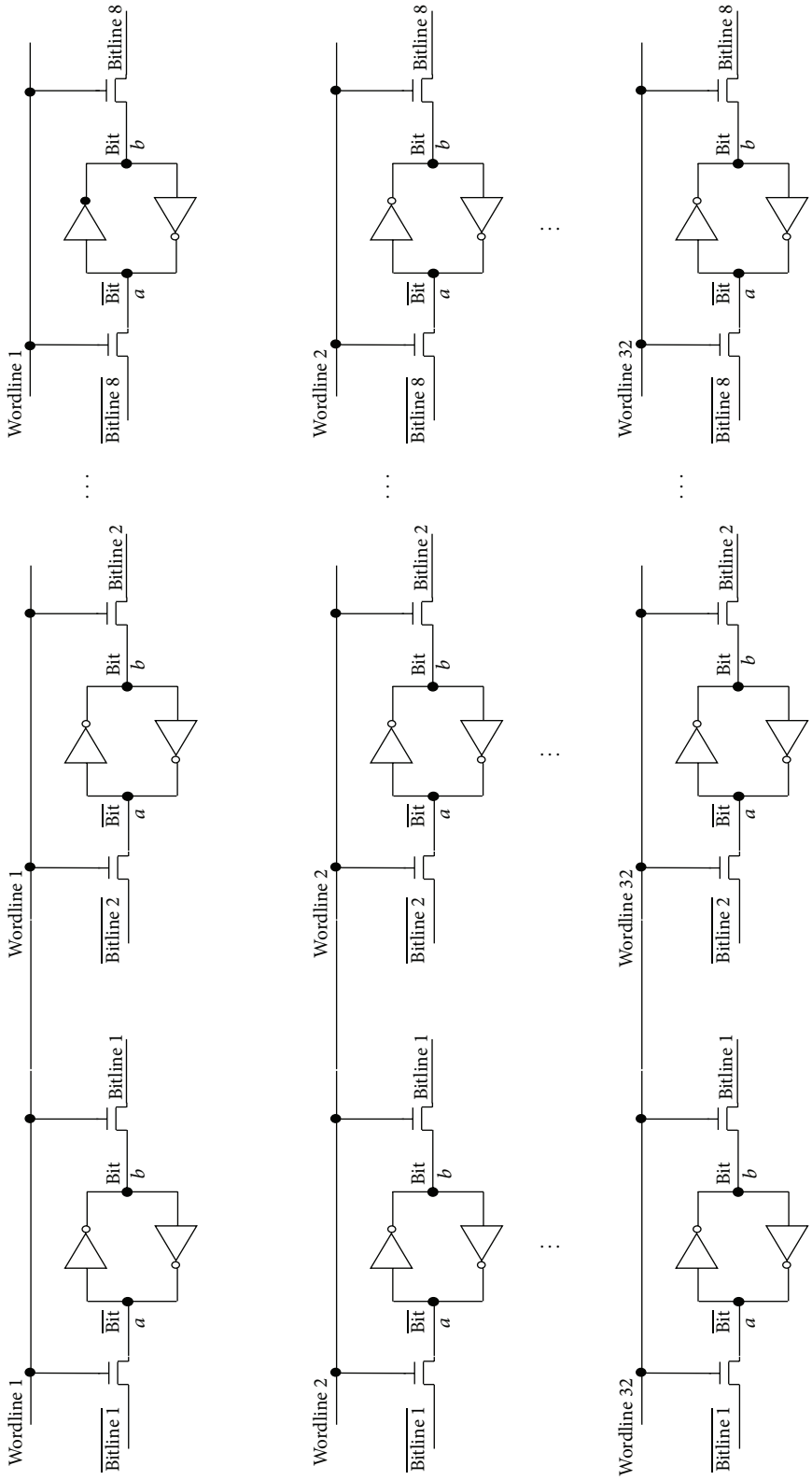
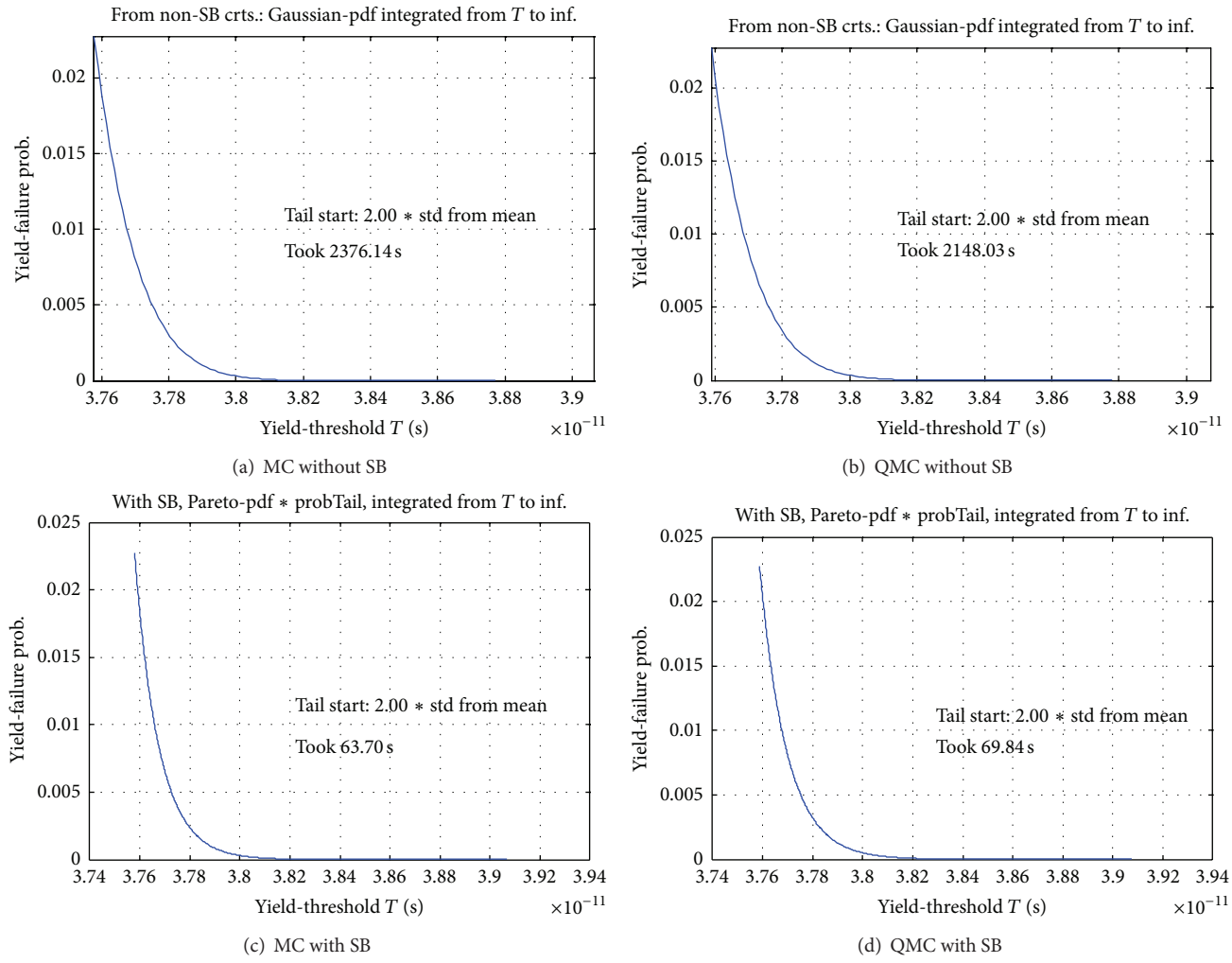


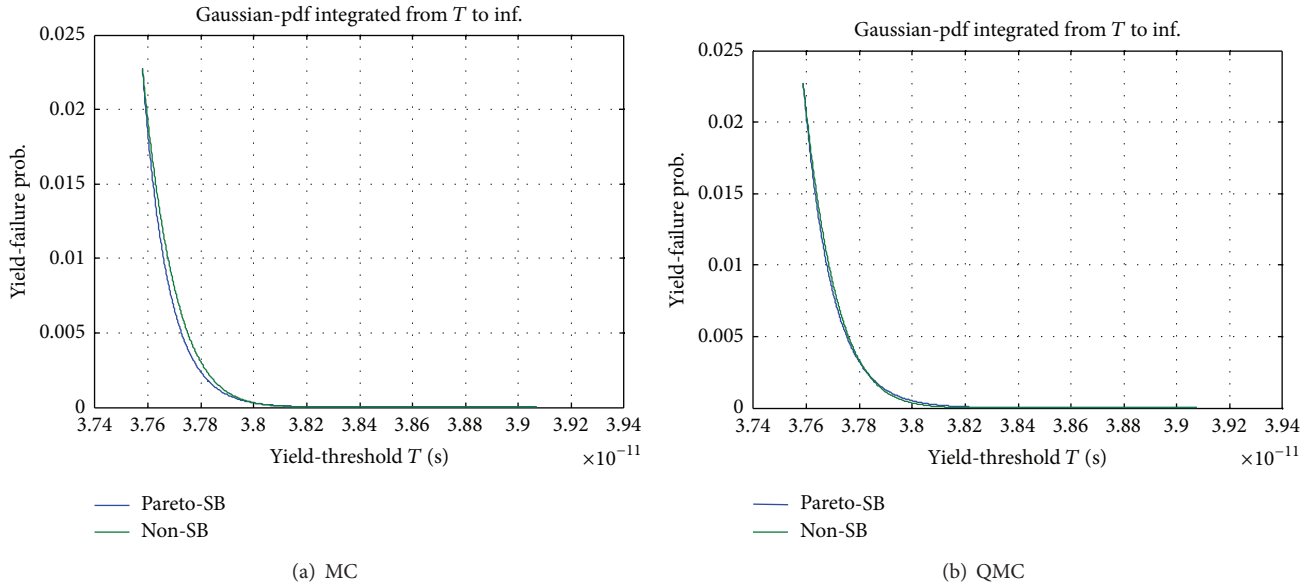
Figure 21: SRAM32 × 8 array circuit.



(a) MC without SB (b) QMC without SB

(c) MC with SB (d) QMC with SB

FIGURE 22: Yield obtained from 3000 behavioural-level simulations of SRAM32 × 8 for strongly correlated case, (a) MC without SB, (b) QMC without SB, (c) MC with SB, and (d) QMC with SB.



(a) MC (b) QMC

FIGURE 23: Comparison of yield analysis results of SB and non-SB for behavioural-level SRAM32 × 8 simulations for strongly correlated case, (a) MC and (b) QMC.

transistor-level simulations of SRAM32 × 8 reduced from 47263.96 s (13.13 hours) to 720 s (12 minutes).

7. Conclusions

Monte Carlo (MC) analysis with analogue simulation is an effective tool for the statistical variability analysis of nanoscale IC designs, taking into account the effects of intradie correlation as modelled by well-known techniques. Computational complexity is a major problem, though a variety of adaptations to the standard MC approach can greatly reduce this complexity as demonstrated by examples based on the simple "exponential" model of correlation due to proximity. These examples indicate that disregarding the effects of intradie correlation may give pessimistic estimates of yield. The results obtained from the simulations of SRAM arrays demonstrate the potential of RandomLA to achieve computation reduction for yield analysis with a delay specification. The RandomLA software is highly suitable for parallel and distributed implementations, which have already been shown to achieve great computation time savings.

Conflict of Interests

The software packages mentioned in this paper, such as Cadence, Synopsys, and MATLAB, were purely used for research under the licenses of the University of Manchester. The authors of this paper do not have a direct financial relationship with the owners of those software packages. RandomLA is the software developed by the authors, intending to contribute to "gEDA" project [14] for developing a full GNU public licensed suit of EDA tools.

Acknowledgments

This research was sponsored by the Engineering and Physical Sciences Research Council (EPSRC) under Grant no. EP/E001947/1. The authors acknowledge the financial support from EPSRC and the collaboration among the people of the Nano-CMOS pilot project, Meeting the Design Challenges of Nano-CMOS Electronics.

References

[1] H. Onodera, "Variability: modeling and its impact on design," *IEICE Transactions on Electronics*, vol. E89-C, no. 3, pp. 342–348, 2006.

[2] A. Asenov, A. R. Brown, J. H. Davies, S. Kaya, and G. Slavcheva, "Simulation of intrinsic parameter fluctuations in decananometer and nanometer-scale MOSFETs," *IEEE Transactions on Electron Devices*, vol. 50, no. 9, pp. 1837–1852, 2003.

[3] G. Roy, A. R. Brown, F. Adamu-Lema, S. Roy, and A. Asenov, "Simulation study of individual and combined sources of intrinsic parameter fluctuations in conventional nano-MOSFETs," *IEEE Transactions on Electron Devices*, vol. 53, no. 12, pp. 3063–3069, 2006.

[4] A. Srivastava, D. Sylvester, and D. Blaauw, *Statistical Analysis and Optimization for VLSI: Timing and Power*, Springer, Berlin, Germany, 2005.

[5] B. Hargreaves, H. Hult, and S. Reda, "Within-die process variations: How accurately can they be statistically modeled?" in *Proceedings of Asia and South Pacific Design Automation Conference (ASP-DAC '08)*, pp. 524–530, March 2008.

[6] A. Singhee and R. A. Rutenbar, "Statistical blockade: a novel method for very fast Monte Carlo simulation of rare circuit events, and its application," in *Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, pp. 1379–1384, April 2007.

[7] A. Singhee, J. Wang, B. H. Calhoun, and R. A. Rutenbar, "Recursive statistical blockade: an enhanced technique for rare event simulation with application to SRAM circuit design," in *Proceedings of the 21st International Conference on VLSI Design*, pp. 131–136, January 2008.

[8] Z. Xie, *Computation reduction for statistical analysis of the effect of nano-CMOS variability on integrated circuits [Ph.D. thesis]*, The University of Manchester, Manchester, UK, 2012.

[9] Z. Xie and D. Edwards, "Computation reduction for statistical analysis of the effect of nano-CMOS variability on asynchronous circuits," in *Proceedings of the 13th IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS '10)*, pp. 161–166, Vienna, Austria, April 2010.

[10] E. J. Gumbel, *Statistical Theory of Extreme Values and Some Practical Applications*, vol. 33 of *Applied Mathematical*, US Department of Commerce, National Bureau of Standards, Washington, DC, USA, 1954.

[11] D. Thain, T. Tannenbaun, and M. Livny, "Distributed computing in practice: the condor experience," *Concurrency and Computation*, vol. 17, no. 2–4, pp. 323–356, 2005.

[12] "High Throughput Computing Using Condor at Manchester," October 2011, http://condor.eps.manchester.ac.uk/.

[13] N. Metropolis and S. Ulam, "The Monte Carlo method," *Journal of the American Statistical Association*, vol. 44, no. 247, pp. 335–341, 1949.

[14] "NGSPICE release 23," June 2011, http://www.sourceforge.net,.

[15] G. Peter Lepage, "A new algorithm for adaptive multidimensional integration," *Journal of Computational Physics*, vol. 27, no. 2, pp. 192–203, 1978.

[16] H. Niederreiter, *" Random Number Generation and Quasi-Monte Carlo Methods*, vol. 63 of *Regional Conference Series in Applied Mathematics*, SIAM, Philadelphia, Pa, USA, 1992.

[17] R. Bellman, *Adaptive Control Processes: A Guided Tour*, Princeton University Press, Princeton, NJ, USA, 1961.

[18] C. Lemieux, *Monte Carlo and Quasi-Monte Carlo Sampling*, Springer Series in Statistics, Springer, New York, NY, USA, 2009.

[19] H. Niederreiter, "Quasi-Monte Carlo methods and pseudo-random numbers," *Bulletin of the American Mathematical Society*, 1978.

[20] S. K. Chaudhary, *Acceleration of Monte Carlo methods using low discrepancy sequences [Ph.D. thesis]*, UCLA, Los Angeles, Calif, USA, 2004.

[21] D. I. Asotsky, E. E. Myshetskaya, and I. M. Sobol', "The average dimension of a multidimensional function for quasi-Monte Carlo estimates of an integral," *Computational Mathematics and Mathematical Physics*, vol. 46, no. 12, pp. 2061–2067, 2006.

[22] A. B. Owen, "Scrambled net variance for integrals of smooth functions," *Annals of Statistics*, vol. 25, no. 4, pp. 1541–1562, 1997.

[23] W. H. Press and G. R. Farrar, "Recursive stratified sampling for multidimensional Monte Carlo integration," *Computers in Physics*, vol. 4, pp. 190–195, 1990.

[24] R. Schürer, "Adaptive Quasi-Monte Carlo integration based on MISER and VEGAS," in *Monte Carlo and Quasi-Monte Carlo Methods*, pp. 393–406, Springer, Berlin, Germany, 2002.

[25] G. P. Lepage, "A new algorithm for adaptive multidimensional integration," *Journal of Computational Physics*, vol. 27, pp. 192–203, 1978.

[26] G. P. Lepage, "VEGAS: an adaptive multi-dimensional integration program," Cornell preprint CLNS 80-447, March 1980.

[27] J. Keiner and U. Waterhouse, "Fast principal components analysis method for finance problems with unequal time steps," in *Monte Carlo and Quasi-Monte Carlo Methods 2008*, P. L'Ecuyer and A. B. Owen, Eds., Springer, New York, NY, USA, 2010.

[28] "RandomSPICE"-info@GoldStandardSimulations.com http://www.GoldStandardSimulations.com/services/circuit-simulation/random-spice/, 2013.

[29] B. Bindu, B. Cheng, G. Roy, X. Wang, S. Roy, and A. Asenov, "Parameter set and data sampling strategy for accurate yet efficient statistical MOSFET compact model extraction," *Solid-State Electronics*, vol. 54, no. 3, pp. 307–315, 2010.

[30] *HSPICE User Guide: Simulation and Analysis, Version A-2007*, Synopsys, Mountain View, Calif, USA, 2007.

[31] J. P. Fishburn and A. E. Dunlop, "TILOS: a polynomial programming approach to transistor sizing," in *Proceedings of IEEE International Conference on Computer-Aided Design (ICCAD '85)*, pp. 326–328, November 1985.

[32] D. P. Marple and A. El Gamal, "Optimal selection of transistor sizes in digital VLSI," in *Proceedings of the Stanford Conference on Advanced Research in VLSI*, pp. 151–172, MIT Press, 1987.

[33] W. Obermeier, *An open architecture for improving VLSI circuit performance [Ph.D. thesis]*, University of California, Berkekey, Calif, USA, 1989.

[34] S. M. Burns, *Performance analysis and optimization of Asynchronous Circuits [Ph.D. thesis]*, California Institute of Technology, Pasadena, Calif, USA, 1990.

[35] G. Mekhtarian, *Composite Current Source (CCS) Modeling Technology Backgrounder*, Synopsys, Mountain View, Calif, USA, 2005, . 11/05.KF.WO .05-13816.

[36] R. Goyal and N. Kumar, *Current Based Delay Models: A Must for Nanometer Timing*, Cadence Design Systems, Noida, India, 2005.

[37] Synopsys, "Liberty Library Modeling," July 2012, http://www.synopsys.com/community/interoperability/pages/libertylibmodel.aspx.

[38] J. F. Croix and D. F. Wong, "Blade and Razor: cell and interconnect delay analysis using current-based models," in *Proceedings of the 40th Design Automation Conference*, pp. 386–389, June 2003.

[39] J. Li, H. Zhao, and H.-Y. Chiu, "Accuracy Timing Models for Integrated Circuit Verification," U.S. patent number 6721929, April 2004.

[40] D. Dasx, W. Scotty, S. Nazariany, and H. Zhoux, "An efficient current-based logic cell model for crosstalk delay analysis," in *Proceedings of the 10th International Symposium on Quality Electronic Design (ISQED '09)*, pp. 627–633, March 2009.

[41] M. Rewieński and J. White, "A trajectory piecewise-linear approach to model order reduction and fast simulation of nonlinear circuits and micromachined devices," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 2, pp. 155–170, 2003.

[42] P. Li, Z. Feng, and E. Acar, "Characterizing multistage nonlinear drivers and variability for accurate timing and noise analysis," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 15, no. 11, pp. 1205–1214, 2007.

[43] C. Visweswariah, K. Ravindran, K. Kalafala et al., "First-order incremental block-based statistical timing analysis," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 25, no. 10, pp. 2170–2179, 2006.

[44] S. I. Resnick, *Extreme Values, Regular Variation and Point Processes*, Springer, New York, Ny, USA, 1987.

[45] D. Edwards, A. Bardsley, L. Janin, and W. Toms, *Balsa: A Tutorial Guide*, School of Computer Science, University of Manchester, Manchester, UK, 2008, http://apt.cs.man.ac.uk/projects/tools/balsa/.

[46] J. Sparsø and S. Furber, *Principles of Asynchronous Circuit Design*, Kluwer Academic Publishers, Norwell, Mass, USA, 2005.

[47] A. Singhee, *Novel Algorithms for Fast Statistical Analysis of Scaled Circuits [Ph.D. thesis]*, Carnegie Mellon University, Pittsburgh, Pa, USA, 2007.

[48] R. E. Caflisch, "Monte Carlo and quasi-Monte Carlo methods," *Acta Numerica*, vol. 7, pp. 1–49, 1998.

[49] L. Kuipers and H. Niederreiter, *Uniform Distribution of Sequences*, Dover Publications, Dover, UK, 2005.

[50] J. Spanier, "Quasi-Monte Carlo methods for particle transport problems," in *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, H. Niederreiter and P. J. -S. Shiue, Eds., pp. 121–148, Springer, New York, NY, USA, 1995.

[51] E. J. Gumbel, *"Statistical Theory of Extreme Values and Some Practical Applications*, vol. 33 of *Applied Mathematical, Series*, US Department of Commerce, National Bureau of Standards, Washington, DC, USA, 1954.