Mixture Importance Sampling and Its Application to the Analysis of SRAM Designs in the Presence of Rare Failure Events

Rouwaida Kanj
IBM Austin Research Labs
Austin, Tx 78758
rouwaida@us.ibm.com

Rajiv Joshi
IBM TJ Watson Labs
Yorktown Heights, NY 10598
rvjoshi@us.ibm.com

Sani Nassif
IBM Austin Research Labs
Austin, Tx 78758
nassif@us.ibm.com

ABSTRACT
In this paper, we propose a novel methodology for statistical SRAM design and analysis. It relies on an efficient form of importance sampling, mixture importance sampling. The method is comprehensive, computationally efficient and the results are in excellent agreement with those obtained via standard Monte Carlo techniques. All this comes at significant gains in speed and accuracy, with speedup of more than 100X compared to regular Monte Carlo. To the best of our knowledge, this is the first time such a methodology is applied to the analysis of SRAM designs.

Categories and Subject Descriptors
B7.2 [**Hardware**]: Integrated Circuits – *Design Aids*

General Terms
Algorithms, Performance, Design, Reliability.

Keywords
Statistical Performance Analysis, Yield Prediction, SRAM

1. INTRODUCTION
In sub-100nm designs, within-chip variability has become a serious problem in circuit design [1, 2]. While its impact on the delay of logic circuits [3, 4] is evident, within-chip variability has an even more sound impact on SRAM cells and memory designs in general [5]. It has been shown that the random dopant fluctuations are inversely proportional to the device area [6], and SRAM cells have the smallest devices on the chip [5]. Also, single (or few) cell failures can lead to failing memory parts. Hence, analyzing the yield of the SRAM cell in the presence of variability is an indispensable part of the memory design and analysis cycle.

The authors in [7] proposed a method for modeling and analyzing the failure probability of SRAM cells. However, it is based on the assumption that the design (performance) metrics are gaussian random variables, whose means and standard deviations are approximated by Taylor series expansion of a response surface model; while this approach is suitable for early design stages, the method in general is prone to errors and may not be accurate for predicting low/rare failure probabilities. Specifically, the performance metric Gaussian approximations may not replicate well the tail probabilities. This is very crucial considering the fact that state-of-the-art designs can have millions of cells placed on a single memory unit and that the fails are largely independent. Consider for example a 1Mb memory design (with no replication). Its yield drops from 90% to 40% when the cell failure rate increases from 1e-5% to 1e-4%. Hence, accurately estimating memory yield is highly dependent on accurately estimating such rare tail probabilities.

Alternatively, it is possible to obtain more accurate results, and hence avoid approximations of the performance space, by integrating the probability density function (pdf) of the sources of variability over the (usually complex) feasibility region [4]. Such methods of statistical design in general are either deterministic or statistical [8], with standard Monte Carlo analysis being the most widely-adopted technique. However, standard Monte Carlo techniques are to a first order dependent on the sample size and are in general slow at estimating low failure/tail probabilities. Deterministic methods, on the other hand, suffer from the curse of dimensionality. Hence, variance reduction techniques, such as importance sampling were proposed as Monte Carlo alternatives [8-10] to speed up the statistical analysis. However, without the proper choice of the sampling function, importance sampling techniques may loose their efficiency and accuracy.

In this paper, we propose using mixture ratioed importance sampling (MixIS) [10]. Our method 1) is independent of any assumptions about the performance space distributions, 2) relies on the proper choice of importance sampling functions, and is therefore 3) accurate and highly efficient. Thus, we are able to achieve several orders of magnitude of gain in simulation run-time compared to standard Monte Carlo techniques. 4) Moreover, the method maintains its high efficiency in multi-dimensional parameter space problems.

2. STANDARD MONTE CARLO
Given the fact that one needs to estimate the SRAM cell low fail probabilities accurately, we will first study the efficiency of standard Monte Carlo at estimating such low probabilities.

2.1 Background
To estimate the yield of a design, we define a performance metric f(x); we also define f_0 to be the corresponding critical value that sets the pass/fail criteria for the design. Let I(x) be the indicator

DAC 2006, July 24–28, 2006, San Francisco, California, USA.

function defined in (1). Also, let p(x) be the true (natural) probability density function (pdf) of the design parameters, i.e., sources of variability. Then the failure probability, P_f, and its variance σ_{Pf}^2 may be estimated from N random samples that are independently drawn according to p(x) (eqn. (2)) [8- 10].

$$I(x) = \begin{cases} 0, & pass \quad (f(x) < f_0; x \in R^n) \\ 1, & fail \quad (f(x) > f_0; x \in R^n) \end{cases} \tag{1}$$

$$P_f = \frac{1}{N}\sum_{j=1}^{N} I(x^j), \text{ and } \sigma_{Pf}^2 = \frac{P_f(1-P_f)}{N} \tag{2}$$

2.2 Estimating Rare Failure Events

Let us assume that we want the 95% confidence interval [P_f-2σ_{Pf}, P_f+2σ_{Pf}] to be equal to [P_f-αP_f, P_f+αP_f], where α is a percent error criteria. Then, based on (2), the number of samples N must be

$$N = \frac{4}{\alpha^2} \bullet \frac{(1-P_f)}{P_f}. \tag{3}$$

For small P_f values, N is inversely proportional to P_f, and, we expect standard Monte Carlo methods to become less efficient as P_f decreases. Table 1 presents the values of N needed to accurately estimate P_f= Prob((f(x)=x) >z_0) with α=10%; x follows a standard normal distribution. Also Table 1 lists what-would-be the equivalent run-time if f(x) involved spice-like simulations of a memory cross-section (cell + peripherals). While the runtime, can be further minimized by using fast simulators, or even response surface models, the number of samples needed for estimating low failure probabilities is neither practical nor reasonable. Thus, there is a need for *fundamentally different* statistical methods.

Table 1. Number of Monte Carlo simulations needed to estimate the probability P_f=Prob(x>z_0) with a 95% confidence interval = [P_f-0.1P_f, P_f+0.1P_f]. The corresponding simulation runtime in days, if a spice-like simulator is used.

z_0	Probability value	Number of Monte Carlo Simulations	Runtime in Number of Days
0	0.500	4.0e2	0.18
1	0.159	2.1e3	0.972
2	0.0228	1.7e4	7.87
3	0.00135	2.9e5	Too long!
4	3.169E-05	1.3e7	Too long!
5	2.871E-07	1.4e9	Too long !

3. IMPORTANCE SAMPLING

Revisiting standard Monte Carlo simulation method, one realizes that Monte Carlo works well; however, it wastes a lot of time sampling around the mean rather than in the tails. Importance sampling [8 -10] is a well-known variance reduction technique which gets around this problem by distorting the (natural) sampling function, p(x), to produce more samples in the important region(s) (see Fig. 1). Mathematical manipulation follows to unbias the estimates. Mathematically, the concept is based on

$$E_{p(x)}[\theta] = E_{g(x)}[\theta \bullet \frac{p(x)}{g(x)}]. \tag{4}$$

where g(x) is the distorted sampling function. The method is theoretically sound, and with the proper choice of g(x), we are able to obtain accurate results with relatively small number of simulations.

3.1 Mixture Importance Sampling (MixIS)

A simple and logical choice of g(x) is the uniform distribution (see Fig. 1). However, it was shown in [8] that its efficiency decreases as the dimensionality increases. Many techniques have been proposed to choose the optimal sampling distribution. A common approach is to shift the natural distribution into the failure region [12]. Choosing g(x) = p(x-μ_s) enables more samples in the failure region (see Fig. 1).

We go one step further. Mixture ratioed importance sampling depends on the idea of generating random variables using mixtures of distributions [10]. Thus, we choose g(x) as follows.

$$g_\lambda(x) = \lambda_1 p(x) + \lambda_2 U(x) + (1-\lambda_1-\lambda_2) p(x-\mu_s); \tag{5}$$

where $0 \le \lambda_1 + \lambda_2 < 1$. g_λ(x) enables focusing on the failure region without leaving any cold spots (i.e., any non-sampled regions in the event of outliers …etc). Moreover, the method is generalizable to support multiple failure regions. The choice of λ_i is dependent on the location of μ_s; in general, if μ_s is far from the origin, λ_i is small.

We propose the following heuristic for estimating the shift μ_s.

1. Uniformly sample the parameter space
   a. Identify Failing points
   b. If (total number of failing samples < 30)
      i. Go to 1
2. Find the center of gravity (C.O.G.) of failures.
   a. Set μ_s=C.O.G.

Note that few failing samples are in general sufficient for estimating the C.O.G. because we are estimating a mean. Furthermore, another round of sampling follows this step; the estimates obtained by importance sampling are not sensitive to the exact position of the C.O.G., or μ_s. This is true as long as the samples in step '1' are representative of the population. To guarantee that the samples in step '1' span the parametric (variability) space properly, we relied on Sobol sequences, also referred to as quasi(sub)-random techniques. The sample points in quasi-random sequences are "maximally avoiding" of each other [14]. Finally, the fact that p(x) is part of the mixture distribution helps bound the weight function w (eqn (6)) used in estimating P_f (eqn. (8)); this in turn bounds the variance of P_f, $\sigma_{Pf:MixIS}$ [10] (eqn. (9)). N_{IS} is the number of samples drawn from g_λ(x).

$$w(x) = \frac{p(x)}{g_\lambda(x)}, \text{ and} \tag{6}$$

$$y(x) = w(x) I(x) \tag{7}$$

$$P_f = \frac{\sum_{i=1}^{N_{IS}} y(x^i)}{\sum_{i=1}^{N_{IS}} w(x^i)} = \frac{\overline{y(x)}}{\overline{w(x)}}. \tag{8}$$

$$\sigma_{Pf:MixIS}^2 = \frac{1}{N_{IS}} \frac{1}{N_{IS}-1} \frac{1}{\overline{w(x)}} \sum_{i=1}^{N} \left(y(x^i) - P_f w(x^i) \right)^2 \tag{9}$$

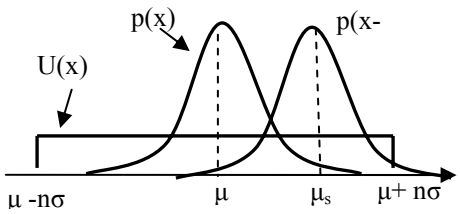

Figure 1. Without loss of generality p(x) is a normal gaussian in 1-D parameter space. U(x) is a uniform pdf; p(x- μ_s) is p(x) shifted to the new center μ_s.

4. MIXIS RESULTS AND ANALYSIS

4.1 Theoretical Efficiency and Accuracy

We repeated the experiments in Table 1, this time for MixIS. We relied on (10), the integral form of (9), to calculate the number of MixIS simulations N_{IS} needed to estimate P_f=Prob(x>z_0) with a 95% confidence interval and error criteria α=0.1; we used MATLAB to evaluate the integrals. Again x follows a standard normal distribution. Results show that the MixIS method provides significant speedup for low failure probabilities; the method converges to regular Monte Carlo for high probability estimates (see Table 2).

$$\sigma^2_{Pf:MixIS} = \frac{1}{N_{IS}} \int (y(x) - P_f w(x))^2 g(x) dx \qquad (10)$$

Table 2. Number of Monte Carlo simulations (2nd column) and MixIS simulations (3rd column) needed to estimate P_f=Prob(x>z_0) with 95% confidence interval = [P_f-0.1P_f, P_f+0.1P_f]. Speed up of MixIS compared to regular Montecarlo (4th column).

z_0	N_{MC} # Monte Carlo Simulations	N_{IS} #MixIS Simulations	Speedup = N_{MC}/N_{IS}
0	4.0e2	4.0e2	1e0
1	2.1e3	5.8e2	4e0
2	1.7e4	9.3e2	2e1
3	2.9e5	1.4e2	2e2
4	1.3e7	1.8e2	7e3
5	1.4e9	2.3e2	6e5

To test the efficiency of the MixIS method in higher dimensions, we relied on experimentation. We created a simple function, f(x_1, …, x_n), for which the yield (in terms of equivalent z; see eqn. (11)) could be computed analytically.

$$z_{eqv} = \varphi^{-1}(1 - P_f) = \varphi^{-1}(1 - prob(f(x_i) > f_0)); \qquad (11)$$

where φ is the standard normal cumulative distribution function (f_0 is the pass/fail critical value). We then used MixIS to estimate z_{eqv} for different f_0 values. Our objective was to determine the number of MixIS samples needed to obtain a good estimate. Table 3 presents the experimental results in 6-D space when the number of MixIS samples was fixed to 2000 (next we will be focusing on SRAM designs with 6 random variables). The confidence intervals are based on 50 replications.

The method maintained its efficiency compared to regular Monte Carlo techniques (see table 1 for the number of Monte Carlo samples needed for the different z-values). Furthermore, MixIS proved to be more efficient than other candidate importance sampling functions (e.g., using U(x) alone, which lost efficiency as the dimensionality increased). It is worth noting that at 6-D U(x) required at least 10-15X more simulations to obtain a good estimate.

Table 3. Comparing MixIS method against analytical solutions for 6-D parameter space. The first (third) column presents the confidence interval lower (upper) bound values for 50 replications. The second column presents the estimated mean value. Each replication involved 1500 simulations.

Analytical z_0	95% Confidence Interval Lower Bound	MixIS z_0	95% Confidence Interval Upper Bound
3.0	2.94	**2.99**	3.09
3.5	3.44	**3.49**	3.57
4.0	3.96	**4.01**	4.06
4.5	4.45	**4.48**	4.55
5.0	4.96	**4.99**	5.07
5.5	5.45	**5.52**	5.55
6.0	5.96	**5.99**	6.07

4.2 Statistical SRAM Analysis

For state-of-the-art memory chips with million or more cells, the yield can drop from 90% to 40% when z_{eqv} drops from 5.3 to 4.8. In the following examples, we will see how the MixIS method lends itself as an efficient statistical methodology to estimate such high yields (low failure probabilities).

4.2.1 Process variability impact on SRAM cells

The threshold voltage fluctuations of the SRAM cell transistors are impacted by the random dopant fluctuations and may be considered as six independent Gaussian random variables, δv_{ti}, i=1 to 6 [13]. Process variations between the neighboring transistors can degrade the cell performance and stability.

We used the MixIS method to estimate the yield of a 6-transistor SRAM cell built in sub-100nm technology in the presence of variability. Our performance metrics are similar to those described in [5]; specifically, we are interested in measuring the cell's dynamic stability in terms of read upsets and writability. For accurate results, the circuit under study consisted of the following components: the SRAM cell, bitline loading and peripheral circuitry. We relied on the following indicator functions to estimate the yield.

$$I(\delta v_{t1},...,\delta v_{t6}) = \begin{cases} 0, & stable \quad cell \\ 1, & read \quad upset \quad (write\_fail) \end{cases} \qquad (12)$$

Similar to [7], the overall yield is computed from the individual (metric) yields. For purposes of our experiments, we report the cell yield as an equivalent z-value as was described in eqn (11).

4.2.2 MixIS: An efficient methodology for statistical SRAM Analysis

Figure 2 compares MixIS method to regular Monte Carlo when estimating the cell yield. Both methods converge to the same estimated value. MixIS converged quickly (few thousand simulations), whereas Monte Carlo is very slow when dealing with rare failure events (very high yields). Table 4. compares

estimated z_{eqv} values obtained via MixIS to those obtained by regular Monte Carlo (whenever a converging Monte Carlo is realizable). MixIS estimates were in excellent agreement with Monte Carlo results. A converging MixIS estimate was achieved with ~2000-3000 samples; this was regardless of the z_{eqv} value. Whereas, the number of MonteCarlo samples increased exponentially with z_{eqv}; for $z_{eqv}>4$ it was no longer practical to rely on Regular Monte Carlo methods. Most importantly, MixIS is a computationally efficient alternative to regular Monte Carlo, and the runtime is independent of the yield estimate. This makes it a suitable methodology for accurately estimating rare fail probabilities of SRAM. Finally, in Table 5. we compare MixIS method results to hardware collected statistics. The results and the trends are in very good agreement. Some discrepancy is seen mainly due to the mismatch between the device models used for simulation and the true hardware behavior..

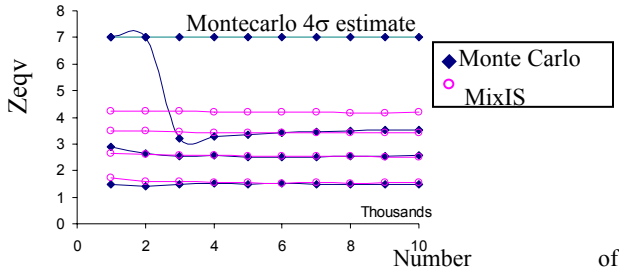

Figure 2. z_{eqv} estimated using both MixIS (circles) and Monte Carlo (diamonds) techniques. For all z_{eqv} values MixIS reached a converging estimate and a converging confidence interval within few thousand simulations. Monte Carlo method fell behind for higher z_{eqv}, i.e., cells with rare failure events. Around 100,000 simulations were needed for regular Monte Carlo to provide a converging estimate for $z_{eqv} \approx 4$. Many more samples are necessary for Monte Carlo method to satisfy the converging confidence interval criteria.

Table 4. Estimated z_{eqv} values. MixIS provided a converging estimate within 2000-3000 simulations. Monte Carlo requirements exceeded 100k simulations for $z>4$.

Monte Carlo	MixIS
1.49	1.53
2.56	2.51
3.49	3.42
4.15	4.22
---	4.96
---	5.68
---	6.06

Table 5. Log of the number of fails based on hardware measurements and those estimated by the MixIS method.

	V_{DD1}	V_{DD2}	V_{DD3}	V_{DD4}	V_{DD5}	V_{DD6}
Hardware	1-0	0-1	1-2	3-4	4-5	5-6
MixIS	0-1	0-1	2-3	3-4	4-5	4-5

5. CONCLUSIONS

We have proposed using mixture ratioed importance sampling (MixIS) for purposes of statistical design. The method is comprehensive and computationally efficient. Thus, we are able to achieve several orders of magnitude of gain in simulation run-time compared to standard Monte Carlo techniques. Moreover, the method maintains its high efficiency in multi-dimensional parameter space problems. We applied this method to the analysis of SRAM designs in the presence of variability. This enabled estimating the probability of the SRAM cell rare failure events. This is critical to the memory design cycle. Single (or few) cell failures can lead to failing memory parts, and the ability to accurately estimate low fail probabilities of an SRAM design is an indispensable part of the memory design cycle. Finally, where feasible, we compared our method to alternative statistical techniques. The results were in excellent agreement, and the speedup factor was significant.

6. REFERENCES

[1] V. De, and S. Borkar, "Technology and design challenges for low power and high performance", ISLPED '99, pp.163– 168.

[2] S. R. Nassif, "Design for variability in DSM technologies", 1st ISQED, 2000, pp. 451 - 454

[3] M. Eisele et al., "The impact of intra-die device parameter variations on path delays and on the design for yield of low voltage digital circuits", IEEE Trans. on VLSI, vol. 5, no. 4, pp. 360– 368, Dec. 1997.

[4] D. E. Hocevar, M. R. Lightner, and T. N. Trick, "A Study of Variance Reduction Techniques for Estimating Circuit Yields", IEEE Trans. on CAD, vol. 2, no. 3, pp. 180 – 192, July 1983.

[5] R. V. Joshi et al., "Variability analysis for Sub-100 nm PD/SOI CMOS SRAM cell", Proc. of the 30th ESSCC, 2004, pp. 211 – 214.

[6] Y. Taur and T. H. Ning, Fundamentals of Modern VLSI Devices. Cambridge University Press, 1998.

[7] S. Mukhopadhyay, H. Mahmoodi, and K. Roy, "Statistical design and optimization of SRAM cell for yield enhancement", ICCAD, 2004, pp. 10 – 13.

[8] J.A.G. Jess, K. Kalafala, S.R. Naidu, R.H.J. Otten, C. Visweswariah, "Statistical timing for parametric yield prediction of digital integrated circuits", DAC 2003, pp. 932 – 937.

[9] W. G. Cochran, Sampling Techniques, 3rd edition. New York: Wiley, 1977.

[10] T. C. Hesterberg, "Advances in importance sampling", Ph.D. Dissertation, Statistics Department, Stanford University, 1988.

[11] D.S. Gibson, R. Poddar, G. S. May, M. A. Brooke, "Statistically based parametric yield prediction for integrated circuits", IEEE Trans. on Semiconductor Manufacturing, vol. 10, no. 4, pp.445 - 458 Nov. 1997.

[12] G. Schueller, H. Pradlewarter, and P. S. Koutsourelakis, "A comparative study of reliability estimation procedures for high dimensions", 16th ASCE Engineering mechanics conference, 2003.

[13] A. J. Bhavnagarwala, T. Xinghai, and J. D. Meindl, "The impact of intrinsic device fluctuations on CMOS SRAM cell stability", IEEE JSSC, vol. 36, no. 4, pp. 658–665, April 2001.

[14] W.H. Press et al., Numerical Recipes in C, 2nd edition. New York: Cambridge University Press, 1997.