Two Fast Methods for Estimating the Minimum Standby Supply Voltage for Large SRAMs

Jiajing Wang, *Member, IEEE,* Amith Singhee, *Member, IEEE,* Rob A. Rutenbar, *Fellow, IEEE,* and Benton H. Calhoun, *Member, IEEE*

***Abstract*—The data retention voltage (DRV) defines the minimum supply voltage for an SRAM cell to hold its state. Intra-die variation causes a statistical distribution of DRV for individual cells in a memory array. We present two fast and accurate methods to estimate the tail of the DRV distribution. The first method uses a new analytical model based on the relationship between DRV and static noise margin. The second method extends the statistical blockade technique to a recursive formulation. It uses conditional sampling for rapid statistical simulation and fits the results to a generalized Pareto distribution (GPD) model. Both the analytical DRV model and the generic GPD model show a good match with Monte Carlo simulation results and offer speedups of up to four or five orders of magnitude over Monte Carlo at the 6σ point. In addition, the two models show a very close agreement with each other at the tail up to 8σ. For error within 5% with a confidence of 95%, the analytical DRV model and the GPD model can predict DRV quantiles out to 8σ and 6.6σ respectively; and for the mean of the estimate, both models offer within 1% error relative to Monte Carlo at the 4σ point.**

***Index Terms*—Data retention voltage, Monte Carlo, SRAM, static noise margin, supply voltage scaling, variation.**

I. Introduction

STANDBY leakage power can dominate the total power budget of memories or system-on-a-chips that dedicate increasingly large percentages of die area to memory. Supply voltage (V_{DD}) scaling is an effective approach for leakage power savings during SRAM/Cache standby mode. Besides the direct effect of smaller voltage on power savings, V_{DD} scaling reduces both sub-threshold leakage current due to drain induced barrier lowering and gate leakage current. Lowering V_{DD} as far as possible maximizes leakage power reduction but might also lead to data loss. The data retention voltage (DRV) is the lower bound of the standby supply voltage that still preserves data in the SRAM cells.

Device variability has been a big challenge for circuit design in nanometer technologies. The most problematic variation is caused by the random inter-device variation sources, like random doping fluctuation (RDF). RDF induced variation increases with technology scaling. The randomness of threshold voltage (V_T) due to RDF can be modeled as a normal distribution with the standard deviation inversely proportional to the channel area [1]. SRAM cells often use the smallest geometry to increase memory density, thus becoming particularly susceptible to RDF. Consequently, the DRV of one cell can be very different from another cell in the same array. Note that the DRV of an entire array is the DRV of the worst cell in the array. The random nature of the DRV of cells makes the array DRV also a random variable. Say, for a 1 Mb array, we desire that at least 99% of manufactured arrays have a DRV of 0.7-V or lower. This places a very strict yield requirement on the cell: the probability of the cell DRV exceeding 0.7-V must be 1.005e-8 (1 part per 100M) or less. For larger array sizes, this *exceedance probability* for the cell must be even lower. These extreme yield requirements on the cell DRV pose a difficult problem for yield estimation, given a cell design.

A straightforward method for obtaining the array DRV is to run a full Monte Carlo (MC) simulation until we obtain DRV values at the required probability levels. However, this is often prohibitively slow for multi-Mb memories (e.g., months on a single machine). For instance, to estimate the DRV with the probability of 1.005e-8, standard MC should run at least 100 million sample points to reach such an extreme probability level. Even then, the estimate of DRV quantile will be suspect because of the lack of statistical confidence. However, running the requisite billions of samples (circuit simulations) is utterly intractable.

Some recent efforts have tried to address this problem of estimating extreme probabilities. For example, [2] shows how importance sampling can be used to predict failure probabilities. Recently, [3] applied an efficient formulation of these ideas for modeling rare failure events of single 6T SRAM cells, based on the concept of *mixture importance sampling* from [4]. Reference [5] proposed some heuristics to enhance the accuracy of this method. However, the method only estimates the failure probability of a *single* threshold value of the performance metric. A re-run is needed to obtain probability estimates for another failure threshold; no

Manuscript received March 10, 2009; revised November 13, 2009 and March 21, 2010; accepted June 11, 2010. Date of current version November 19, 2010. This work was supported by the Focus Center for Circuit and System Solutions, one of five research centers funded under the Focus Center Research Program, a Semiconductor Research Corporation Program. This paper was recommended by Associate Editor S. Vrudhula.

J. Wang is with the Advanced Design Group, Intel Corporation, Hillsboro, OR 97124 USA (e-mail: jiajing.wang@intel.com).

A. Singhee is with the IBM T. J. Watson Research Center, Yorktown Heights, NY 10598 USA (e-mail: asinghe@us.ibm.com).

R. A. Rutenbar is with the University of Illinois at Urbana-Champaign, Urbana, IL 61820 USA (e-mail: rutenbar@illinois.edu).

B. H. Calhoun is with the Department of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA 22904 USA (e-mail: bcalhoun@virginia.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2010.2061810

complete model of the tail of the distribution is computed. Knowledge of the tail distribution allows the designer to take a yield-centric approach: for example, target a specific yield level, rather than a performance specification, or look at the sensitivity of yield to the supply voltage to make array-level design decisions. The YENSS method of [6] abandons MC and uses a boundary search method to estimate the yield. Although it provides significant speedup in low dimensions, the complexity of the search may increase exponentially with increasing dimensionality.

An alternative is to run a small MC and fit some pre-determined distribution form (e.g., Gaussian) to the resulting DRV sample set. This fitted distribution can then be used to extrapolate far into the tail of the distribution. However, the DRV follows some non-Gaussian distribution and *ad hoc* fits can lead to large errors. In this paper, we present and apply two rigorous techniques based on this alternative, for efficient estimation of extreme DRV tail distributions.

1) A statistical model for DRV. Although [7] has proposed a theoretical model for the DRV of a single cell, a rigorous model for the DRV distribution has not been proposed yet. This paper derives a new statistical model for the DRV distribution from the relation between DRV and static noise margin (SNM). Since the model is obtained from the physical behavior of the cell, it remains valid even in the extreme tail of the DRV distribution and allows us to estimate the array-wide DRV for arbitrary array size.
2) A generic tail model from recursive statistical blockade. The statistical blockade (SB) method, proposed in [8], improved upon standard MC for simulating rare events by simulating only those samples that are likely to appear in the tail of the distribution of any performance metric (e.g., DRV). The simulated tail points are used to fit a generalized Pareto distribution (GPD) model to the conditional distribution of events far out in the tail. However, the original SB method does not directly support DRV estimation because of the presence of a conditional in the formulation of the metric. Also, if rare samples with extremely low probability are required, SB can still become prohibitively expensive. In this paper, we extend the SB method to a recursive formulation, that can also handle conditionals. These extensions allow us to efficiently build GPD-based models for the tail of the DRV distribution.

Both methods operate in two steps to achieve accuracy and efficiency: 1) use a small MC-based simulation to learn relevant information about the cell design, and 2) estimate a theoretically sound model for the DRV distribution (or its tail) based on this information. The first method requires fewer circuit simulations, as it exploits specific characteristics of DRV, while the second method can be applied to other metrics. A preliminary version of this paper was presented in [9] and [10]. We present in this paper a more rigorous treatment of the recursive statistical blockade algorithm, and a new, extensive analysis of the statistical accuracy of both our methods.

The remainder of this paper is organized as follows. Section II discusses the definition of DRV and basic characterizations. Section III presents the details of our statistical model for DRV, based on the connection between DRV and SNM. Section IV presents the extension of SB for estimating DRV at extremely rare tail points. Section V describes the results from different methods and further discusses the confidence in DRV estimation. Finally, we draw conclusion and discuss future work in Section VI.

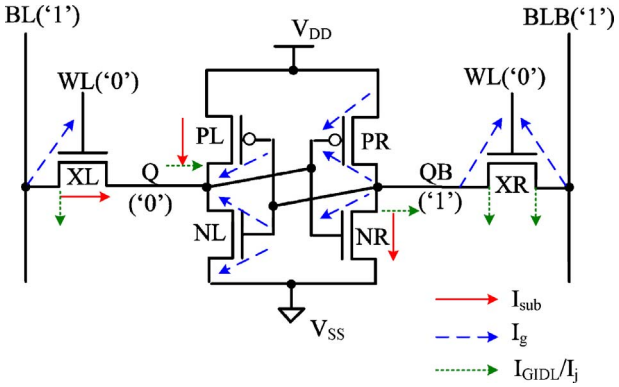


Fig. 1. 6T SRAM cell and various leakage current paths inside the cell.

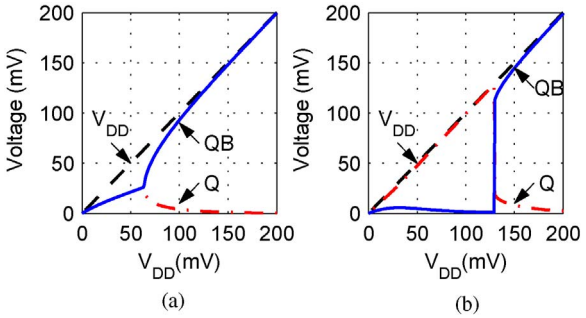


Fig. 2. Cell nodes Q and QB (a) converge if the cell is balanced or (b) flip if the cell is imbalanced when V_{DD} is lower than DRV.

II. DRV and its Statistics

Fig. 1 shows the structure of the traditional 6T SRAM cell as well as the major leakage paths during standby mode, where the pass-gate transistors (XL and XR) are turned off. Lowering V_{DD} can exponentially reduce all the leakage components, including sub-threshold current (I_{sub}), gate leakage (I_g), junction leakage (I_j) and the gate induced drain leakage (GIDL) current (I_{GIDL}). Due to the direct effect of V_{DD}, the cell leakage power can be further reduced with a lower V_{DD}. Many designs have exploited V_{DD} scaling for SRAM leakage power reduction during standby and/or active operation [7], [11]–[16].

However, collapsing V_{DD} degrades cell stability. Fig. 2 elaborates on the behavior of Q and QB as V_{DD} is lowered. Fig. 2(a) shows the case when the cell is balanced (symmetric), with identical left and right halves. With V_{DD} scaling, the cell nodes Q and QB converge to a metastable point as a result of degraded gain, making the "0" and "1" states indistinguishable. Fig. 2(b) shows the case when the cell is imbalanced by some variation induced mismatch in the

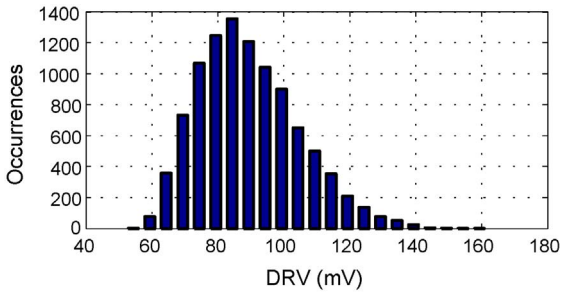


Fig. 3. Simulated DRV histogram for a 10K-b SRAM in a 90 nm node.

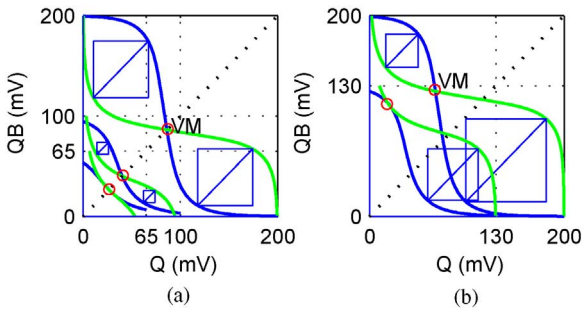


Fig. 4. VTCs of (a) balanced and (b) imbalanced cells with varying V_{DD}; V_M is the trip point of the VTCs.

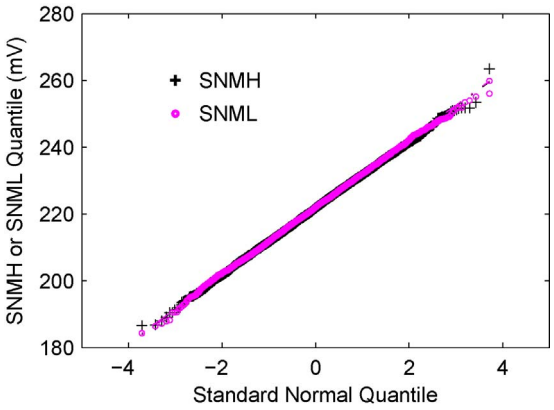


Fig. 5. Quantiles of SNMH and SNML vs. the theoretical standard normal quantiles when V_{DD} = 0.6-V.

transistors. In this case, Q and QB flip to the more stable state ("1" here). The DRV defines the minimum V_{DD} that can be applied to an SRAM cell without losing data. Since a cell can store either a "0" or a "1," the actual DRV is computed as follows:

$$\text{DRV} = \max(\text{DRV}_0, \text{DRV}_1) \tag{1}$$

where DRV_0 and DRV_1 are the DRV when the cell is storing a "0" and "1" respectively. If the cell is balanced, then $\text{DRV}_0 = \text{DRV}_1$. However, if there is any mismatch due to process or lithography variations, they become unequal. Particularly, one becomes much larger and the other becomes much smaller or even close to 0. Smaller DRV means the cell is more stable. This implies that mismatch will make the cell more stable for one data value while less stable at the other. Therefore, to obtain the real DRV, we must pick the worse (larger) from both DRV_0 and DRV_1.

We run MC simulation with independent normally-distributed V_T variation on each transistor of the 6T cell. Fig. 3 shows the histogram of a 10K-point MC simulation for the DRV of SRAM cells in a commercial 90 nm CMOS process. The DRV exhibits a skewed distribution with a heavy tail on the right side. DRV values in this heavy tail are most relevant for a fault-free SRAM array, since the worst cell in the array determines the minimum standby V_{DD} that can be applied to the array. Accurate estimation of the DRV values in the tail is essential for optimizing the tradeoff between SRAM yield and standby power savings. If the tail value is over-estimated, the cell will be over-designed or over-protected thus limiting the standby power savings. On the contrary, if it is under-estimated, more cell failures than predicted will occur, and the yield cannot be met. However, for large memories, we need to estimate extremely rare DRV quantiles, and standard MC simulation is too expensive, computationally. This problem motivates our work in this paper. In the next two sections, we present two fast and accurate methods for DRV tail estimation. The first one takes a designer's approach by exploiting the electrical characteristics of the SRAM cell. The second method takes an algorithmic and mathematical view to solve the general problem of rare event estimation.

III. DRV Statistical Model Based on SNM

Since DRV is the minimum V_{DD} below which a cell cannot preserve its data, we can also consider it as the V_{DD} at which SNM first equals zero in a noiseless system. Therefore, we propose to use SNM as a starting point to explore DRV statistics.

A. Cell Hold Stability and its Sensitivity to V_{DD}

SNM measures the amount of DC voltage noise that a cell can tolerate and equals the length of the largest square that can be embedded between the voltage characterization curves (VTCs) of the two half-cells [17] as shown in Fig. 4. Particularly, the largest square in the upper-left lobe is the SNMH, the noise margin for holding "0"; the one in the lower-right lobe is the SNML, the margin for holding "1". The actual SNM is computed as follows:

$$\text{SNM} = \min(\text{SNMH}, \text{SNML}). \tag{2}$$

Fig. 4 also shows the change of VTCs and the embedded SNM squares as we decrease V_{DD} using the same example cells as in Fig. 2. Fig. 4(a) shows that symmetry allows the cell to remain bistable to lower V_{DD}. It becomes clear that the DRV equals the supply voltage at which SNM is equal to zero in a noiseless system. Both SNMH and SNML decrease symmetrically to zero in the absence of mismatches, implying that DRV_0 and DRV_1 are equal. However, the stability of the imbalanced cell (and its DRV) is determined by its worst-case data pattern. For example, in Fig. 4(b) this particular imbalanced cell is less stable when $Q = 0$. Its SNMH first decreases to zero at a lower V_{DD}, and its DRV is thus determined by DRV_0.

Intra-die V_T variation is the major source of cell imbalance (i.e., mismatch), thus it has a huge impact on SNM. Fig. 5 plots the SNMH and SNML quantiles from a 5K-point MC simulation vs. the theoretical standard normal quantiles at $V_{\rm DD}$ = 0.6-V. The near strict linearity in this quantile-quantile (Q-Q) plot implies that both SNMH and SNML can be well approximated by a normal distribution. Therefore, we can accurately estimate the mean (μ) and standard deviation (σ) of SNMH or SNML by fitting a normal distribution to data from a small-scale (e.g., 1.5 K–5 K points) MC simulation. Note that besides intra-die process variation, higher temperature and aging effects such as NBTI and HCI degrade SNM and DRV. In this paper, we focus on the change of SNM and DRV due to intra-die variation, but the proposed analytical model can also be extended to address the temperature and aging effects.

To find DRV, we must lower $V_{\rm DD}$ until SNM reaches zero, so it is necessary to examine the sensitivity of SNM to $V_{\rm DD}$. We observe that SNMH or SNML remains normally distributed at different $V_{\rm DD}$ conditions. Moreover, the sensitivity of μ and σ of SNMH or SNML to $V_{\rm DD}$ actually exhibits a nice trend. Fig. 6 plots the simulated SNMH μ and σ from 5-K MC samples under different $V_{\rm DD}$s. σ remains almost constant while μ linearly decreases with $V_{\rm DD}$ scaling. This is reasonable since the shape of the distribution is mainly determined by the intrinsic parametric V_T variation, which is unchanged with $V_{\rm DD}$ scaling. Thus, the sensitivities of the SNMH μ and σ to $V_{\rm DD}$ can be approximated as

$$\frac{\partial \sigma}{\partial \mathrm{V_{DD}}} \approx 0 \quad \frac{\partial \mu}{\partial \mathrm{V_{DD}}} \approx k \tag{3}$$

where k is the coefficient obtained by fitting a linear curve to the mean values. To obtain those mean values, we have to run thousands of MC simulations for each $V_{\rm DD}$ point. We observe that the curve of the nominal SNMH vs. $V_{\rm DD}$ has about the same slope as the curve of the μ values vs. $V_{\rm DD}$. Thus we can also extract k from the linear fit to the curve of the nominal results, which only requires a single short DC-sweep SPICE simulation and thus offers a great speedup. However, this simplified procedure might lead to some errors. In Section V-B, we will further discuss how to maintain a good accuracy when using k extracted from a single DC simulation.

B. SNM and DRV Statistical Model

The real SNM of the cell is the minimum of SNMH and SNML. As shown in Fig. 5, the distribution of SNMH and SNML are almost identical because of the symmetry of the 6T cell. If we assume SNMH and SNML are also independent random variables, then the cumulative density function (CDF) of SNM at supply voltage v is

$$\begin{aligned} F_{\mathrm{SNM}_v}(s) &= \mathrm{P}(\mathrm{SNM}_v < s) \\ &= \mathrm{P}(\min(\mathrm{SNMH}_v, \mathrm{SNML}_v) < s) \\ &= \mathrm{P}(\mathrm{SNMH}_v < s) + \mathrm{P}(\mathrm{SNMH}_v \geq s, \mathrm{SNML}_v < s) \\ &= 2F_{\mathrm{SNMH}_v} - F^2_{\mathrm{SNMH}_v} \end{aligned} \tag{4}$$

where F_{SNMH_v} is the CDF of SNMH at supply voltage v.

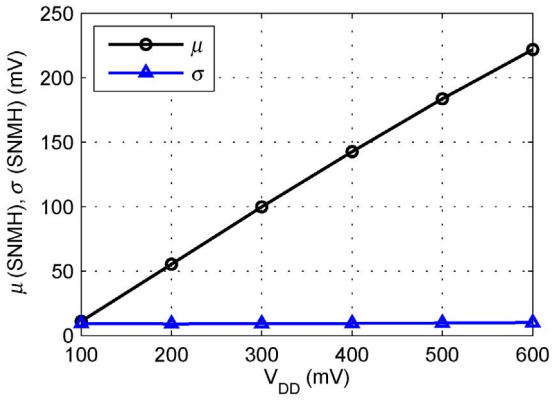


Fig. 6. Mean (μ) and standard deviation (σ) of SNMH against $V_{\rm DD}$ under mismatch.

A cell hold failure occurs when the cell's SNM is less than the desired noise margin s, which is often a positive value. If more noise immunity is needed (e.g., for protecting against larger voltage fluctuation), s should be larger. We define $P_f(v, s)$ as the cell hold failure probability when $V_{\rm DD} = v$ and the minimum acceptable noise margin is s. $P_f(v, s)$ can be computed with (4). Since SNMHv is a normal random variable $\mathcal{N}(\mu, \sigma^2)$, F_{SNMH_v} can be computed with $\frac{1}{2}\mathrm{erfc}(\frac{\mu - s}{\sqrt{2}\sigma})$ [18], where erfc$(\cdot)$ is the complementary error function. We can finally compute $P_f(v, s)$ as (5) by using (3)

$$\begin{aligned} P_f(v, s) &= \mathrm{P}(\mathrm{SNM}_v < s) \\ &= \mathrm{erfc}(x) - \frac{1}{4}\mathrm{erfc}^2(x) \\ &\text{where} \quad x = \frac{\mu_0 + k(v - v_0) - s}{\sqrt{2}\sigma_0} \end{aligned} \tag{5}$$

and μ_0 and σ_0 are some estimates of the mean and standard deviation of SNMH at an initial voltage, v_0. Equation (5) allows us to quickly estimate the cell failure probability at any new $V_{\rm DD}$ without rerunning simulations at the new voltage condition.

DRV is the minimum operation voltage during standby mode. More specifically, we denote the random variable DRV_s as the cell DRV for a specific noise margin requirement s. Thus, the failure of the cell at the supply voltage v can also be defined as the event when DRV_s is larger than v

$$P_f(v, s) = \mathrm{P}(\mathrm{DRV}_s > v) = 1 - F_{\mathrm{DRV}_s}(v). \tag{6}$$

By equating (5) and (6), we can compute the inverse CDF of DRV_s as follows:

$$F^{-1}_{\mathrm{DRV}_s}(p) = \frac{1}{k}\left(\sqrt{2}\sigma_0 \cdot \mathrm{erfc}^{-1}\left(2 - 2\sqrt{p}\right) - \mu_0 + s\right) + v_0 \tag{7}$$

$$\text{where } \mathrm{P}(\mathrm{DRV}_s \leq F^{-1}_{\mathrm{DRV}_s}(p)) = p$$

and erfc$^{-1}(\cdot)$ is the inverse function of erfc$(\cdot)$. Equation (7) allows us to directly compute the standby supply voltage required to maintain a desired cell yield.

Both (5) and (7) only require four parameters. They are the initial V_{DD} value (v_0), and three fitting coefficients: the mean and standard deviation of SNMH at v_0 (μ_0 and σ_0) as well as the sensitivity of SNMH to V_{DD} (k). Now let us summarize the steps of using our model as follows.

1) Pick a value for v_0.
2) Extract μ_0 and σ_0 from a 1.5K–5K-point MC simulation of SNMH when $V_{DD} = v_0$.
3) Extract k from a short DC-sweep of the nominal SNMH vs. V_{DD}.
4) Pick a value for s as a minimum acceptable noise margin.
5) Use (5) to compute the cell hold failure probability $P_f(v, s)$ when $V_{DD} = v$.
6) Compute the minimum V_{DD} value (i.e., DRV_s) satisfying the required cell hold failure probability p_f from (7) with $p = 1 - p_f$.

We will discuss the sensitivity of the model to the four parameters in Section V-B.

IV. Recursive Statistical Blockade and its Application to DRV

The analytical model derived in the previous section provides excellent estimates of the DRV. However, as with most analytical models, it is tightly coupled to the specific context of DRV estimation. In this section, we examine an alternative, more general method that modifies the approach to MC simulation. In particular, we extend the SB method proposed in [8]. The original method generates a model for the tail distribution of any performance metric in two steps: 1) simulate only rare tail events by filtering standard MC samples, and 2) fit a GPD model to the simulated tail values. The method imposes no *a priori* limitations on the form of the statistics for the statistical parameters, device models, or performance metrics.

Although statistical blockade provides us an effective method for sampling rare events and modeling their statistics, there are still some practical issues left unresolved by the original algorithm in [8]. In this paper, we extend the SB framework to handle metrics with conditionals [e.g., max(), min()] that result in disjoint rare event regions, and incorporate a recursive formulation to produce reliable estimates for extremely rare events (6σ to 8σ).

A. Background: Statistical Blockade

The authors in [8] recognize that for memory cells, the statistics of only rare events is relevant. In our case, we are concerned with the distribution of very high DRV quantiles in the "tail" of the DRV distribution, and not with the "body" of the distribution. SB defines a *tail threshold* (for example, the 99% point) t. Without loss of generality, we define the part of the distribution greater than this threshold as the tail, and we are interested in the shape of this tail. Here, we can exploit an important result from extreme value theory [19] that says, roughly, that the conditional distribution of the events in the tail tend toward a GPD as we move further out in the tail toward ∞. The conditional CDF of any DRV value in the tail is the CDF *given that the DRV is in the tail*, written as

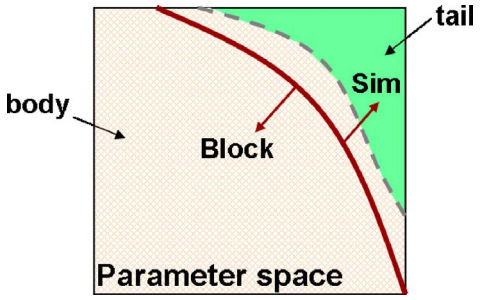


Fig. 7. Tail and body regions in the statistical parameter space. The dashed line is the exact tail region boundary for tail threshold t. The solid line is the relaxed boundary modeled by the classifier for a classification threshold $t_c < t$.

$$F_{\text{DRV}_{s,t}}(z) = P(\text{DRV}_s - t \le z | \text{DRV}_s > t) \tag{8}$$

$$= \begin{cases} \frac{F_{\text{DRV}_s}(z+t) - F_{\text{DRV}_s}(t)}{1 - F_{\text{DRV}_s}(t)} & z >= 0 \\ 0 & z < 0 \end{cases} \tag{9}$$

where F_{DRV_s} is the CDF of the DRV. The CDF of the GPD is

$$G_{\xi,\beta}(z) = \begin{cases} 1 - \left(1 - \xi \frac{z}{\beta}\right)^{1/\xi} & \xi \neq 0 \quad z \in D(\xi, \beta) \\ 1 - e^{-z/\beta} & \xi = 0 \quad z \ge 0 \end{cases} \tag{10}$$

where

$$D(\xi, \beta) = \begin{cases} [0, \infty) & \xi \le 0 \\ [0, \beta/\xi] & \xi > 0 \end{cases}.$$

It is defined by two parameters ξ and β. [19] shows that under certain widely satisfied conditions on F_{DRV_s}

$$\lim_{t \to \infty} F_{\text{DRV}_{s,t}}(z) = G_{\xi,\beta}(z) \tag{11}$$

for some (ξ, β). Relevant theoretical details of this theorem can be found in [20] and the references therein.

Using $G_{\xi,\beta}$ for $F_{\text{DRV}_{s,t}}$ for large t, we can approximate the inverse CDF of DRV_s [as in (7)]

$$F^{-1}_{\text{DRV}_s}(p) = t + G_{\xi,\beta}(p)^{-1}\left(\frac{p - p_t}{1 - p_t}\right) \tag{12}$$

where $p_t = F_{\text{DRV}_s}(t)$ is the CDF for $\text{DRV}_s = t$. For example, if we choose t as the 99th percentile, $p_t = 0.99$. Of course, for a given p_t, we would need to estimate t. Statistical blockade then needs to estimate three parameters t, ξ, and β. This is done with a filtered MC scheme [8] that we briefly review here.

The key idea is to identify the region in the parameter (process variable) space that yields circuit performance values (e.g., SRAM DRV) greater than t. After identifying this region, we only simulate those MC samples that lie in the tail region, ignoring (or blocking) the other samples. This characterization step significantly reduces the number of simulations required for identifying rare events. For example, if t is the 99th

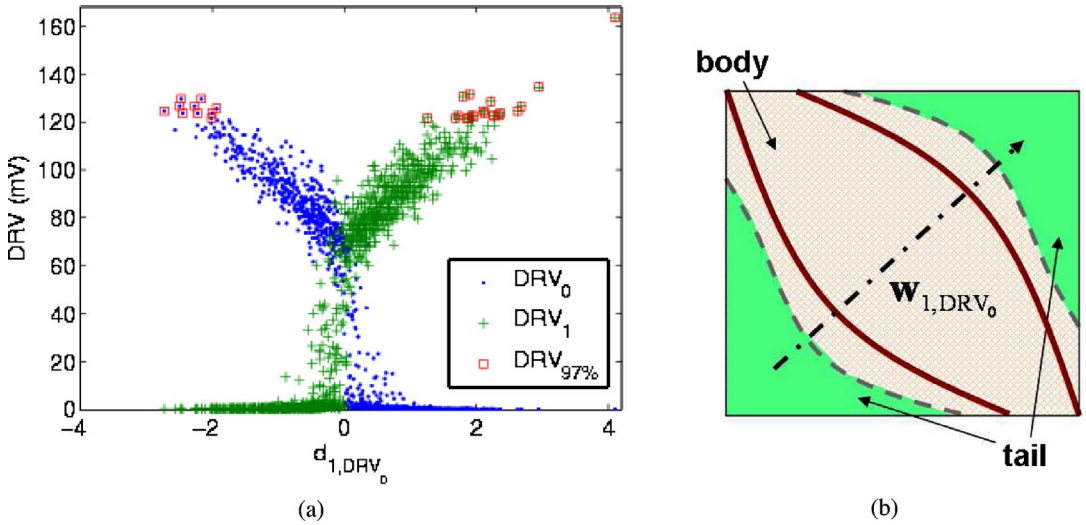


Fig. 8. Illustration of disjoint tail regions resulting from conditionals. (a) Behavior of DRV_0 and DRV_1 along the direction of maximum variation in DRV_0. The worst 3% DRV values are plotted as (red) squares, clearly showing the disjoint tail regions (along this direction in the parameter space). (b) Circuit metric (e.g., DRV) with two disjoint tail regions. The tail regions are separated from the body region by dashed lines; $\mathbf{w}_{1,DRV_0}$ is the direction of maximum variation of the circuit metric.

percentile, we must simulate only 1% of the MC samples, resulting in an immediate speedup of 100× over standard MC.

To build this model of the boundary of the tail region, we use a small MC sample set (1000 points) to train a classifier. However, it is difficult, if not impossible, to build an exact model of the tail region boundary. [8] suggests relaxing this requirement to allow for classification error by building the classification boundary at a *classification threshold* t_c that is less than the tail threshold t. Fig. 7 shows this relaxed classification boundary in the 2-input parameter space. The dashed line is the exact boundary of the tail region for the tail threshold t, and the solid line is the relaxed classification boundary for the classification threshold t_c. Reference [21] suggests using t_c as the 97th percentile, based on empirical analysis of the tradeoff between classifier accuracy, simulation time, and tail model fit. SB filtering and GPD model building is then accomplished as follows.

1) *Perform initial sampling* to generate data to build a classifier. This initial sampling can be standard MC or importance sampling. Also estimate t and $t_c < t$ from this data.
2) *Build a classifier* using the classification threshold t_c.
3) *Generate more samples* using MC, following the CDF F, but simulate only those that are classified as tail points. Update the estimate of t.
4) *Fit GPD model* to the simulated tail points.

There are several methods for fitting the GPD model to data (step 4) (see [8]). Here we will use the maximum likelihood estimate (MLE).

We will now describe the practical insufficiencies of SB for our DRV statistics problem, and our proposed solutions.

B. Conditionals and Disjoint Tail Regions

1) *Problem:* SRAM performance metrics are often computed for two states of the SRAM cell: storing a 1, and storing a 0. The final metric value is then a maximum or a minimum of the vales for these two states. The presence of such *conditionals* (max, min) can result in *disjoint* tail regions in the statistical parameter space, making it difficult to use a single classifier to define the boundary of the tail region. This creates a situation where the standard SB classification technique would fail because of the presence of disjoint tail regions. SRAM DRV is a typical example of the maximum problem since DRV is the maximum of DRV_0 and DRV_1. Suppose we run a 1000-point MC, varying all the mismatch parameters in the SRAM cell according to their statistical distributions. This would give us distributions of values for DRV_0, DRV_1, and DRV. In certain parts of the mismatch parameter space $DRV_0 > DRV_1$, and in other parts, $DRV_0 < DRV_1$. This is clearly illustrated by Fig. 8(a): let us see how. Using the SiLVR method proposed in [22], we extract the direction in the parameter space that has maximum impact on DRV_0. This direction is essentially the projection vector $\mathbf{w}_{1,DRV_0}$ for the first latent variable of DRV_0. The figure plots the simulated DRV_0 and DRV_1 values from the 1000-point MC run, along this direction; i.e., against the first latent variable d_{1,DRV_0}. It is clear that the two DRV measures are inversely related: one decreases as the other increases.

Now, let us take the maximum as in (1), and choose the classification threshold t_c equal to the 97th percentile. Then we pick out the worst 3% points from the classifier training data and plot them against the same latent variable in Fig. 8(a), as squares. Note that we have not trained the classifier yet, we are just looking at the points that the classifier would have to classify as being in the tail. We can clearly see that these points (the red squares) lie in two disjoint parts of the parameter space. Since the true tail region defined by the tail threshold $t > t_c$ will be a subset of the classifier tail region (defined by t_c), the true tail region consists of two disjoint regions of the parameter space. This is illustrated with a 2-D example in Fig. 8(b). The figure also shows the maximum impact direction

vector, similar to the projection vector $\mathbf{w}_{1,\mathrm{DRV}_0}$ extracted by SiLVR. Although this vector is different from $\mathbf{w}_{1,\mathrm{DRV}_0}$, which lies in the higher dimensional space of per-device statistical parameters (e.g., the V_T skews of the six SRAM devices), we mark it as $\mathbf{w}_{1,\mathrm{DRV}_0}$ to make the relation obvious. The dark tail regions on the top-right and bottom-left corners of the parameter space correspond to the large DRV values shown as squares in Fig. 8(a).

In the general case for arbitrary number of arguments in the conditional, we can write the circuit metric as

$$y = \max(y_0, y_1, \ldots). \tag{13}$$

Such conditionals are very common for SRAM cell metrics, and hence, a classification strategy for such cases is essential for practical use of statistical blockade. If we know beforehand that the disjoint regions are symmetric, we can just scale up the probability mass of the tail by a factor equaling the number of arguments in (13). For example, in the case of DRV, if we run standard SB on DRV_0 (or DRV_1) with $p_t = 0.99$, we can use (12), but with p_t replaced by $(1 - 2(1 - p_t)) = (2p_t - 1)$. However, in the general case there may be asymmetries as in the case of an 8T SRAM cell [23]. Hence, we now propose a strategy for the general case, that is independent of the circuit characteristics.

2) *Solution:* Instead of building a single classifier for the tail of (1), let us build two separate classifiers, one for the 97th percentile (t_{c,DRV_0}) of DRV_0, and another for the 97th percentile (t_{c,DRV_1}) of DRV_1. The generated MC samples can then be filtered through both of these classifiers: points classified as "body" by *both* the classifiers will be blocked, and the rest will be simulated. The resulting general algorithm is then as follows.

1) Perform initial sampling to generate training data to build the classifiers, and estimate tail and classification thresholds, t_i and $t_{c,i}$, respectively, for each $y_i, i = 0, 1, \ldots$. Estimate the tail threshold t for y.
2) For each argument, $y_i, i = 0, 1, \ldots$, of (13), build a classifier C_i at a classification threshold $t_{c,i}$ that is $< t_i$, the corresponding tail threshold.
3) Generate more points using MC, but block the points classified as "body" by *all* the classifiers. Simulate the rest and compute y for the simulated points. Update the estimate of t.
4) Fit the GPD model to the points with $y > t$.

Hence, in the case of Fig. 8(b), we build a separate classifier for each of the two boundaries. The resulting classification boundaries are shown as solid lines. Note that this same algorithm applies to the case of multiple circuit metrics. Each metric would have its own thresholds and its own classifier, just like each argument in (13), the only difference being that we would not be computing any conditional.

C. Recursive Formulation of Statistical Blockade

1) *Problem:* Suppose we wish to support our GPD model with data up to the 6σ point to increase statistical confidence in predictions far into the tail. The failure probability of a 6σ value is roughly 1 part per *billion*, corresponding to a 99% chip yield requirement for a 10 Mb cache (with no error protection). This is a reasonable requirement for large memories. However, for a 99% tail threshold, even a perfect classifier ($t_c = t$) will only reduce the number of simulations to an extremely large 10 million. If we decide to use a 99.9999% threshold, the number of simulations will be reduced to a more practical 1000 tail points (with a perfect classifier). However, we will need to simulate an extremely large number of points ($\geq$1 million) to generate a classifier training set with at least one point in the tail region. In both cases, the circuit simulation counts are too high. We now describe a recursive formulation of statistical blockade that reduces this count drastically.

2) *Solution:* Let us first assume that there are no conditionals. For a tail threshold equal to the ath percentile, let us represent it as t^a, and the corresponding classification threshold as t_c^a. For this threshold, build a classifier C^a and generate sufficient points beyond the tail threshold, $y > t^a$, so that a *higher* percentile (t^b, t_c^b, $b > a$) can be estimated. The confidence interval-based stopping criterion suggested in [21] can be used to determine the number of points that is sufficient. In this paper, we have used a fixed number of 1000. For the new, higher threshold t_c^b, a new classifier C^b is trained

Algorithm 1 The recursive statistical blockade algorithm with fixed sequences for the tail and classification thresholds: t = 99%−, 99.99%−, 99.9999%−, ... points and t_c = 97%−, 99.97%−, 99.9997%, ... points. The total sample size is given by (14).

Require: initial sample size n_0 (e.g., 1000); total sample size n; performance metric function $y = \max(y_0, y_1, \ldots)$

$\mathbf{X} = \mathrm{MonteCarlo}(n_0)$
$n' = n_0$
$\mathbf{Y} = \mathbf{f}_{\mathrm{sim}}(\mathbf{X})$ // Simulate initial MC sample
$\mathbf{y}_{\mathrm{tail},i} = \mathbf{Y}_{\cdot,i}, i = 0, 1, \ldots$ // The ith column contains values for y_i in $y = \max(y_0, y_1, \ldots)$
$\mathbf{X}_{\mathrm{tail},i} = \mathbf{X}, i = 0, 1, \ldots$
while $n' < n$ **do**
  $\Delta n = 99n'$ // Number of points to filter in this recursion step
  $n' = n' + \Delta n$ // Total number of points filtered at the end of this recursion step
  $\mathbf{X} = \mathrm{MonteCarloNext}(\Delta n)$ // The next Δn points in the MC sequence
  for all $i : y_i$ is an argument in $y = \max(y_0, y_1, \ldots)$ **do**
    $(\mathbf{X}_{\mathrm{tail},i}, \mathbf{y}_{\mathrm{tail},i}) = \mathrm{GetWorst}(1000, \mathbf{X}_{\mathrm{tail},i}, \mathbf{y}_{\mathrm{tail},i})$ // get the 1000 worst points
    $t = \mathrm{Percentile}(\mathbf{y}_{\mathrm{tail},i}, 99)$
    $t_c = \mathrm{Percentile}(\mathbf{y}_{\mathrm{tail},i}, 97)$
    $C_i = \mathrm{BuildClassifier}(\mathbf{X}_{\mathrm{tail},i}, \mathbf{y}_{\mathrm{tail},i}, t_c)$
    $(\mathbf{X}_{\mathrm{tail},i}, \mathbf{y}_{\mathrm{tail},i}) = \mathrm{GetGreaterThan}(t, \mathbf{X}_{\mathrm{tail},i}, \mathbf{y}_{\mathrm{tail},i})$ // get the points with $y_i > t$
    $\mathbf{X}_{\mathrm{cand},i} = \mathrm{Filter}(C_i, \mathbf{X})$ // Candidate tail points for y_i
  end for
  $\mathbf{X} = [\mathbf{X}_{\mathrm{cand},0}^T \ \mathbf{X}_{\mathrm{cand},1}^T \ \ldots]^T$ // union of *all* candidate tail points
  $\mathbf{Y} = \mathbf{f}_{\mathrm{sim}}(\mathbf{X})$ // Simulate all candidate tail points
  $\mathbf{y}_{\mathrm{cand},i} = \{\mathbf{Y}_{j,i} : \mathbf{X}_{j,\cdot} \in \mathbf{X}_{\mathrm{cand},i}\}, i = 0, 1, \ldots$ // Extract the tail points for y_i
  $\mathbf{y}_{\mathrm{tail},i} = [\mathbf{y}_{\mathrm{tail},i}^T \ \mathbf{y}_{\mathrm{cand},i}^T]^T, \ \mathbf{X}_{\mathrm{tail},i} = [\mathbf{X}_{\mathrm{tail},i}^T \ \mathbf{X}_{\mathrm{cand},i}^T]^T, i = 0, 1, \ldots$ // All tail points till now
end while
$\mathbf{y}_{\mathrm{tail}} = \mathrm{MaxOverRows}([\mathbf{y}_{\mathrm{tail},0} \ \mathbf{y}_{\mathrm{tail},1} \ \ldots])$ // compute the conditional
$\mathbf{y}_{\mathrm{tail}} = \mathrm{GetWorst}(n_t, \mathbf{y}_{\mathrm{tail}})$
$(\xi, \beta) = \mathrm{FitGPD}(\mathbf{y}_{\mathrm{tail}} - \min(\mathbf{y}_{\mathrm{tail}}))$

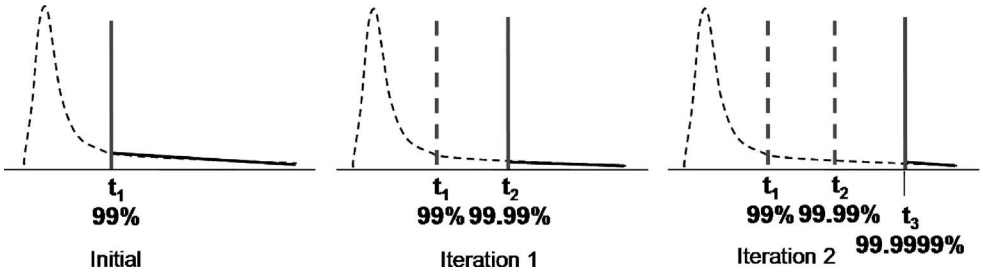


Fig. 9. Recursive formulation of statistical blockade as in Algorithm 1.

and a new set of tail points ($y > t^b$) is generated. This new classifier will block many more points than C^a, significantly reducing the number of simulations. Repeating this procedure pushes the threshold out more until the tail region of interest is reached.

Algorithm 1 shows the complete pseudo-code for this recursive formulation of statistical blockade, applied to general conditionals (13). This pseudo-code uses a *conditional* tail threshold of the 99th percentile at each recursion stage: $(t^a, t_c^a) = (99\%, 97\%)$ points, $(t^b, t_c^b) = (99.99\%, 99.97\%)$ points, and so on. Consequently, the total sample size (without filtering) n is restricted to some power of 100, times 1000

$$n = 100^j \cdot 1000 \quad j = 0, 1, \ldots \tag{14}$$

These threshold values are an extension of the values chosen in the original SB paper [8], where the authors relied on empirical evidence. A general version of the algorithm for arbitrary thresholds is presented in [20]. Practical implementation of this general algorithm is, however, difficult and is a topic for further research.

The pseudo-code is largely self-explanatory, but we detail a few of the variables and function calls for clarity. $\mathbf{f}_{\text{sim}}$ returns multiple outputs: it computes the values of all the arguments of the conditional in $y = \max(y_0, y_1, \ldots)$. For example, in the case of DRV, it will return the values of DRV_0 and DRV_1 after running the requisite circuit simulation. These values, for any one MC point, are stored in one row of the result matrix $\mathbf{Y}$. The function `MonteCarloNext(`Δn`)` returns the *next* Δn points in the sequence of points generated until now. The function `GetWorst(`n`,` $\mathbf{X}$`,` $\mathbf{y}$`)` returns the n worst values in the vector $\mathbf{y}$ and the corresponding rows of the matrix $\mathbf{X}$. This functionality naturally extends to the two arguments `GetWorst(`n`,` $\mathbf{y}$`).GetGreaterThan(`t`,` $\mathbf{X}$`,` $\mathbf{y}$`)` returns the elements of $\mathbf{y}$ that are greater than t, along with the corresponding rows of $\mathbf{X}$.

The algorithm presented here is in iterative form, rather than recursive form. To see how the recursion works, suppose we want to estimate the 99.9999% tail. To generate points at and beyond this threshold, we first estimate the 99.99% point and use a classifier at the 99.97% point to generate these points efficiently. To build this classifier in turn, we first estimate the 99% point and use a classifier at the 97% point. Fig. 9 illustrates this recursion on the PDF of any one argument in (13).

V. Results

A. DRV Estimate Comparisons

We now test our DRV analytical model described in Section III and the recursive statistical blockade method described in Section IV on the DRV test case with an SRAM cell in a commercial 90 nm process. Without loss of generality, we choose zero noise margin (i.e., $s = 0$) as the cell failure criterion.

Fig. 10 plots the DRV quantiles against the quantiles of a standard normal distribution. That is, if the qth quantile of the standard normal distribution is equal to m (i.e., m-σ point for a standard normal) and the qth quantile of the DRV distribution is equal to y, we plot the point at (m, y) coordinates of the figure. Since SRAM arrays usually have at least 1000 cells, we are only interested in the quantiles larger than 99.9th percentile, which is $\sim 3\sigma$ point of the standard normal distribution. Fig. 10 uses different methods to estimate the DRV quantiles for $m \in [3, 8]$.

1) *Analytical DRV model*: use (7) with p equal to the probability of the normal quantile at m (i.e., $p = 0.5\text{erfc}(-m/\sqrt{2})$). We extracted $k = 0.425$ from a DC sweep simulation for SNMH. We selected 100 mV as v_0 and obtained the parameters $\mu_0 = 11.0$ mV and $\sigma_0 = 9.3$ mV from a 5K-point MC simulation.
2) *Standard MC or fast MC with recursive statistical blockade*: use standard MC for estimations below 4σ. Estimates greater than 4σ were obtained by the recursive blockade method, thus allowing dramatically reduced simulation time. Note that here we are not using the GPD model, but only the empirical estimate from the sampled values. Algorithm 1 is run for $n = 1$ billion. This results in three recursion stages, corresponding to total sample sizes of $n' = 100\,000$, 10 million, and 1 billion MC points, respectively. The worst DRV values for these three recursion stages are estimates of the 4.26σ, 5.2σ, and 6σ points, respectively.
3) *GPD model from recursive statistical blockade*: the 1000 tail points from the last recursion stage of the recursive statistical blockade run are used to fit a GPD model, which is then used to predict the DRV quantiles.
4) *Gamma function*: we also fit one set of 5000 DRV samples to a variety of commonly used one-sided distributions (e.g., lognormal, exponential, Weibull, Rayleigh, and gamma) as well as the normal distribution. They either overestimate or underestimate the DRV values in

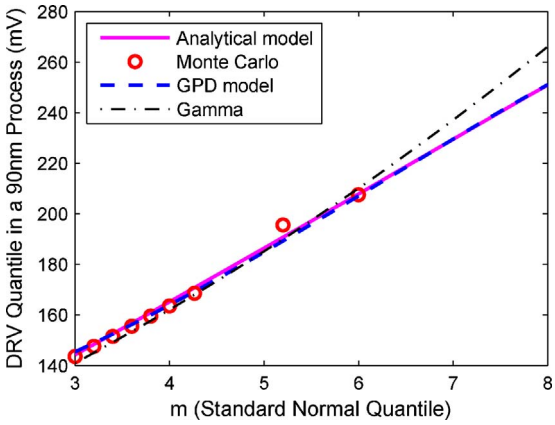


Fig. 10. Estimates of DRV quantiles in a commercial 90 nm process from different methods. The GPD model closely fits the analytical model (7). The (red) circles with $m \leq 4$ are obtained from standard MC simulation and the three circles with $m > 4$ show the worst DRV values from the three recursion stages of statistical blockade sampling. Although the gamma function shows the best fit among a variety of commonly used one-sided distributions for this data set, it overestimates DRV for $m > 6$.

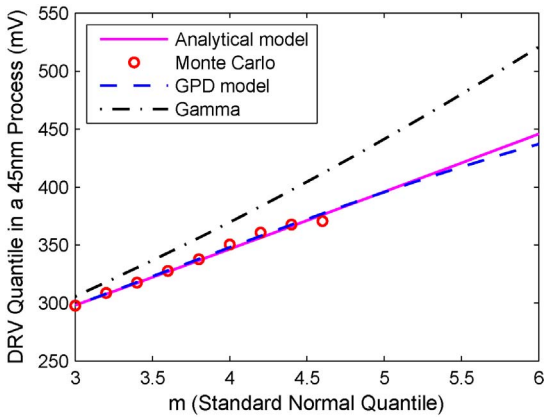


Fig. 11. In a 45 nm process, the gamma fit becomes inaccurate, while the proposed analytical model and the GPD model closely match MC results and maintain excellent agreement.

the tail region. Among them, the gamma distribution shows the best fit for this set of data.

From Fig. 10, we can immediately see that the results from both the analytical DRV model and the GPD model closely track the MC results up to 6σ. In addition, the two proposed models match each other even up to 8σ. This matching of independently derived models increases the confidence of their accuracy. Note here that, since our final GPD model uses t as the 99.9999th percentile point ($\sim 4.75\sigma$ point), the model does not have a real probabilistic meaning below $m = 4.75$. However, it can still be employed as a purely shape fitting function, as long as we are far from the mode in the distribution of DRV.

In addition to the two proposed methods, the approach of directly fitting a known distribution is used to estimate DRV quantiles. Since the DRV is a non-negative skewed distribution, a variety of known one-sided distributions (e.g., lognormal, Weibull, and gamma) as well as the normal distribution are fitted to the DRV data. For the data set plotted in Fig. 10, the gamma distribution shows the best fit. However, its extrapolation in the tail beyond 6σ overestimates relative to the two proposed models. A major issue with this distribution fitting approach is that the fit may suffer from overfitting, since the distribution pick is purely data based, with not much use of the problem structure. Therefore, even if the gamma distribution fits well for one set of data, the extrapolation in the tail may be very incorrect for a different circuit, or even another data set in the same circuit. Fig. 11 shows an example when we test the DRV for a 6T cell in a commercial 45 nm process. The best fitting distribution (gamma) for the 90 nm test case is no longer appropriate for the 45 nm test case. In contrast, our two proposed models both attempt to maximally capture the problem characteristics, albeit in two different ways, i.e., one based on circuit characteristics (SNM) and the other on the tail statistics. As seen in Fig. 11, they maintain high accuracy relative to MC and excellent agreement with each other in the 45 nm process. Since we have explicitly used the mathematical properties of the DRV tail statistics [(4)–(6) and (8)–(10)] in both our proposed models, they are much more accurate for estimating the tail values than any other one-sided reliability distribution empirically picked from a large class of distributions. In fact, the proposed GPD model is also a reliability distribution, but it is applied to the conditional tail distribution and not to the DRV distribution itself. In specific, it encodes the Pareto, type-II Pareto, and the exponential distributions.

With high accuracy, both of our proposed methods offer significant speedup over the standard MC method. The analytical DRV model only needs a few thousand MC simulations to extract μ and σ of the SNMH distribution, and then can be used to predict any extreme DRV tail value. It can thus provide a speedup of $10^5\times$ over MC for a 6σ point, which requires at least 1 billion simulations. For our recursive formulation of statistical blockade, Table I shows the number of circuit simulations performed at each recursion stage for a 6σ point prediction. The total number of simulations is 41 721. This is drastically smaller than standard MC and basic, non-recursive statistical blockade (approximately, 30 million with t_c = 97th percentile). 41 721 simulations for DRV computation of a 6T SRAM cell can be completed in several hours on a single computer. With the advent of multi-core processors, the total simulation time can be drastically reduced further with proper implementation. Note that in cases where the disjoint tail regions are symmetrical (as in the case of the DRV), we need to simulate only one of the arguments in (13). This would reduce the simulation count in each recursion stage (after the initial stage), by a factor equal to the number of arguments. In this case of DRV, we need to simulate only DRV_0 (or DRV_1), and using the numbers in Table I, the simulation count will be further reduced, to approximately 21 361.

B. Confidence in the Estimates

Intuitively, the statistical confidence of our estimates decreases as we predict farther out in the tail. In other words, the variance of the predictions will probably increase as we move out in the tail. Next, we will assess the confidence interval of the DRV estimation from the analytical DRV model, the GPD model as well as the MC method.

TABLE I
NUMBER OF CIRCUIT SIMULATION NEEDED BY RECURSIVE STATISTICAL BLOCKADE TO GENERATE A 6σ POINT

Recursion Stage	Number of Simulations
Initial	1000
1	11 032
2	14 184
3	15 505
Total	41 721
Speedup over MC	23 969×
Speedup over statistical blockade	719×

Suppose we have n estimates $y_i(m), i = 1, \ldots, n$ for the $m\sigma$ point, say by building the statistical DRV model from n different MC runs of SNM. From these estimates we can empirically compute the 97.5% percentile and 2.5% percentile points, $y_{97.5\%}(m)$ and $y_{2.5\%}(m)$, respectively. A 95% confidence interval $\kappa_{95\%}(m)$ can then be estimated [24] as

$$\kappa_{95\%}(m) = y_{97.5\%}(m) - y_{2.5\%}(m). \tag{15}$$

The 95% confidence interval can also be expressed as $[\hat{y} - \alpha\hat{y}, \hat{y} + \alpha\hat{y}]$, where α is the radius of 95% confidence interval as a percentage of the mean of the estimates. Smaller α implies lower variance in the DRV estimate. For a $m\sigma$ point, we then compute α as

$$\alpha(m) = \frac{\kappa_{95\%}(m)}{\frac{2}{n}\sum_{i=1}^{n} y_i(m)}. \tag{16}$$

We use this empirical method to compute 95% confidence intervals for the estimate of the $m\sigma$ point of the SRAM cell DRV, where $m \in [3, 8]$. Fig. 12(a) shows the radius of the 95% confidence interval for the analytical DRV model, the GPD model, and the MC method (using 1 million sample points). Next, we will describe how we obtain the results for each approach.

1) *Confidence Interval of the Analytical Model:* The analytical model in (7) only requires four parameters k, v_0, μ_0, and σ_0. The variance of the estimation is determined by the sensitivity of the model to these parameters. Since μ_0 and σ_0 are fitting coefficients from a small-scale MC simulation of SNMH when $V_{DD} = v_0$, we first assess the variance of the DRV estimates from the variance of these two parameters.

We can empirically estimate the confidence or variance of this model as follows. We first fix v_0 and k, and run n runs of MC simulations with n_{MC} samples each. That will give us n different pairs of μ_0 and σ_0. Then we use (7) to compute n estimates of DRV at the m point, and use (15) and (16) to compute the tightness of the 95% confidence interval, $\alpha(m)$, under this pair of (v_0,k). We choose v_0=100 mV and run 50 MC iterations using 1000 samples for each. k is approximated to 0.425 by fitting the linear curve to the data from the DC simulation of the nominal SNMH vs. V_{DD}. The dashed curve in Fig. 12(a) shows the computed $\alpha(m)$ values, which are all below 4% for $m \in [3, 8]$.

We then use this method to compute $\alpha(m)$ for different pairs of (v_0,k) to check the sensitivity of the variance to v_0 and k. We alter v_0 from 100 mV to 200 mV and 300 mV. For k, besides the value 0.425 from the nominal SNMH vs. V_{DD}, we evaluate another value, 0.4438, which is obtained from the linear fit to the estimated mean of SNMH from those MC samples at the three v_0 points. As we mentioned in Section III-A, it is much faster to obtain k from the nominal SNMH by just running a single short DC sweep simulation. However, it is necessary to examine whether this faster method can also offer the comparable accuracy. Here, we first compare the two k values in terms of the variance of the estimate. In Section V-B3, we will also show the accuracy of the mean of the estimate with these two k values. Fig. 12(b) plots the radius of the 95% confidence interval for each pair of (v_0,k). For all the curves, although the statistical error (α) slightly increases with m, it remains within 4% (i.e., above 96% accuracy for 95% confidence interval) up to the 8σ point. This indicates that the variance of the DRV estimates from our model is not sensitive to either parameter.

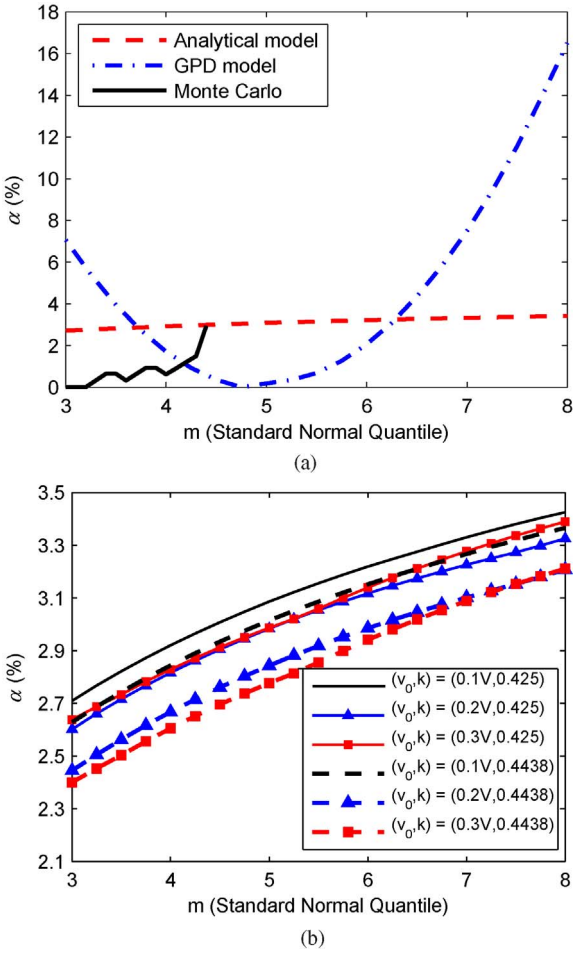


Fig. 12. Radius of 95% confidence intervals as a percentage of the mean value for DRV estimation (a) for the analytical model, the GPD model and the MC method, and (b) using the analytical model with different pairs of v_0 and k.

2) *Confidence Interval of the GPD Model:* The variance of the prediction from the GPD model depends on statistical error in estimating three parameters t, ξ, and β. Let us look at t first. t is an estimate of the p_tth quantile, estimated using a total sample size of n (e.g., 1 billion) points.

Suppose that $x_i, i = 1, 2, \ldots, n$, are the sample order statistics (e.g., DRV values); i.e., the sampled values *sorted* in ascending order. Then, any x_i is also an estimate of the i/nth quantile. But, there is a non-trivial probability that some other order statistic x_j, $j \neq i$, may match the actual i/nth quantile. Now suppose that p_t is such that np_t is an integer. Then, the probability that the ith order statistic, x_i, equals the p_t quantile is given by a binomial distribution [24], which for large n can be well approximated by the normal distribution

$$P(x_i = p_t\text{th quantile}) \sim N(np_t, np_t(1-p_t)). \tag{17}$$

A $\pm 2\sigma$ 95.45% confidence interval in terms of the quantile estimate index i is then given by

$$[l, h] = \left[\lfloor np_t - 2\sqrt{np_t(1-p_t)}\rfloor, \lceil np_t + 2\sqrt{np_t(1-p_t)}\rceil\right]. \tag{18}$$

For the large values of n relevant to us, this range of indices constitutes an extremely small fraction of n. For our DRV example of $n = 1$ billion, this fraction is $\sim \pm 63 \times 10^{-9}$. This fraction is also the probability measure for this set of indices. The corresponding confidence interval for the quantile estimates is $[x_l, x_h]$, where (l, h) are defined by (18). This range of x has the same small probability measure, and is also expected to be very small. For our case, the width of this confidence interval of t (the estimated 99.9999th percentile) is a negligible 0.009% of t. In other words, the statistical error in the estimate of t is negligibly small ($\pm 0.0046\%$). Consequently, we ignore it in comparison to the statistical error in the GPD parameter estimates $(\hat{\xi}, \hat{\beta})$.

As the number of points used to fit the GPD parameter tends to ∞, the MLE parameter estimates $(\hat{\xi}, \hat{\beta})$ tend to a normal distribution with covariance matrix $\Sigma_{\xi,\beta}$, given by [25]

$$\Sigma_{\xi,\beta} = \frac{1-\xi}{n}\begin{bmatrix} 1-\xi & \beta \\ \beta & 2\beta^2 \end{bmatrix} \quad \xi < \frac{1}{2}. \tag{19}$$

Substituting the estimated $(\hat{\xi}, \hat{\beta})$, gives us an estimate of this covariance matrix. For our test case, we get

$$\Sigma_{\hat{\xi},\hat{\beta}} = \begin{bmatrix} 0.0366 & 0.0042 \\ 0.0042 & 0.0010 \end{bmatrix} \tag{20}$$

i.e., standard deviations of $(\hat{\xi}, \hat{\beta})$ are $(0.1912, 0.0309)$.

Now, we sample 10 000 pairs of GPD parameters from the joint normal distribution with this mean vector and its covariance matrix to compute different estimates of DRV. With these DRV estimates, we compute the confidence interval-based accuracy measure, α, using (15) and (16). The dash-dotted curve in Fig. 12(a) shows the result of applying this method. It suggests that for error within 5% with a confidence of 95% we can predict out to 6.6σ (1 in 48.6 billion).

3) *Comparison with Standard MC:* Finally, we compare the confidence intervals of the estimates from our two methods with the confidence intervals of standard MC estimates. The confidence interval of the $m\sigma$ from an n-point MC run is given by $[x_l, x_h]$, where l and h are given by (18), but now we replace p_t with the CDF associated with m: $\Phi(m)$. Once again, x_l and x_h are the lth and hth order statistics from the n-point sample. We now estimate confidence intervals for our 1 million point MC run. The result is plotted as the solid curve in Fig. 12(a). The width of the confidence interval from MC increases as we estimate further out to the larger m points. Note that with 1 million MC samples, we can only obtain the confidence interval up to $\sim 4.5\sigma$ because there is no available DRV estimate beyond that.

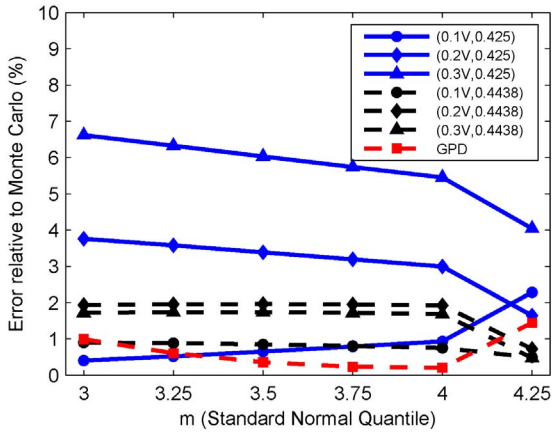


Fig. 13. Error of the mean of the DRV estimates from the analytical model and the GPD model relative to a 1-million-point MC. For the analytical model, results from different pairs of (v_0, k) indicate that a higher accuracy can be achieved by choosing v_0 near the DRV of an ideal cell, which is $\sim$100 mV for this 90 nm test case.

We further compare the mean of the DRV estimation from the two models with MC. Fig. 13 shows the error relative to the result of MC when $m \in [3, 4.25]$. Although we use the term error, it should be noted that the MC estimate itself has some statistical error [Fig. 12(a)]. For $m \leq 4$, the GPD model offers less than 1% error. A slightly larger error occurs when $m = 4.25$, where the MC result is itself less confident as shown in Fig. 12(a). In Fig. 13, we also plot the error of the analytical model for different pairs of (v_0,k). Estimation with $v_0 = 100$ mV shows the best agreement with MC and the GPD model ($\sim$1% of error for $m \leq 4$). It should be noted that the nominal DRV is less than 100 mV for the 90 nm node we used. With $v_0 = 100$ mV, we can obtain more samples with negative SNMH or SNML values so that the approximated μ_0 and σ_0 can be closer to the true statistics of SNMH or SNHL, which are the key ingredients of the analytical DRV model. The results also show that when we use the k value 0.425 that is obtained from a single DC simulation of the nominal SNMH vs. V_{DD}, the accuracy of the DRV model is more sensitive to v_0. In this case, estimation with $v_0 > 100$ mV shows relatively larger errors. However, if we choose k as 0.4438, the value from the linear fit to the curve of the mean of SNMH vs. V_{DD}, the sensitivity of the error to v_0 is reduced. This is because the effect of different μ_0 at different v_0 is eliminated and the shift of the estimate is only limited by the relative small difference of σ_0 at distinct v_0 points. Therefore, we suggest that a value of v_0 close to, but larger than the DRV of an ideal cell is a better choice for a higher accuracy, especially when using k from the quick DC simulation of the nominal case for a faster estimate.

VI. Conclusion and Future Work

Local variation, or mismatch, causes a spread of DRV for cells in the same SRAM array. The worst-case tail of the DRV distribution becomes the critical metric and sets the minimum standby supply voltage for the whole memory. We proposed two fast and accurate methods to estimate DRV at the rare tails. Based on the relationship between DRV and SNM, we proposed an analytical model to estimate the DRV value. We also proposed an enhanced statistical blockade method to estimate DRV tail. We overcome two shortcomings in the original SB technique. First, we extended SB to metrics like DRV that include conditionals. Then we presented a recursive way to improve the efficiency and accuracy of SB for extremely rare events.

The results from both the analytical DRV model and the GPD model based on data from SB match well with the result from MC simulation and show a close agreement with each other at the tail out to 8σ. Both of the methods offer huge speedups over MC. For one estimate of a 6σ point, the analytical model gives speedups of up to five orders of magnitude and the recursive SB tool offers up to four orders of magnitude. We also assessed the 95% confidence interval for the two models as well as the MC method. To keep the estimations within 95% accuracy of 95% confidence interval, we can use the analytical model to predict DRV quantiles out to the 8σ point and use the GPD model to predict out to 6.6σ with $\leq$50 000 samples. While for standard MC, even 1 000 000 samples can only allow the prediction of the DRV quantiles out to the $\sim 4.5\sigma$ point. For the mean DRV estimation at 4σ, both methods show error within 1% relative to MC.

Due to the different nature of different methods, they are more advantageous in different cases. For small or medium scale memories with tail points $<6\sigma$, we can use either the analytical DRV model or the GPD model based on recursive SB, because both are fast and accurate with high confidence. However, for large memories, we suggest using the analytical DRV model in the early stages of SRAM cell design because it provides smaller variance as well as larger speedup in extremely rare tail region ($>6\sigma$). When further design refinement is needed, we can accelerate MC simulations with the recursive SB approach to obtain the real MC samples at the tail point and get the most accurate estimation.

Recently, some novel cells have been proposed to offer better performance over the 6T cell in deeply scaled technologies. An example is the 8T cell [23] which has gained in popularity because it effectively eliminates cell disturbs during a read access. For those new types of cells, the analytical DRV model can be directly applied to any symmetrical ones. In the case of asymmetrical cells like the 8T cell, the only difference is that the statistics of SNMH and SNML might be different. Thus we need two new parameters, μ and σ of SNML. We can easily obtain them with the same method for SNMH and extend the DRV model to include them. Since it is a generic method, the recursive SB method can be directly used for the DRV of any types of cells and even for metrics other than DRV.

In addition to supporting asymmetrical cells, the analytical model can be extended in several directions. First, we can extend the model to estimating the minimum supply voltage ($V_{\min}$) for read and write operation, which is more critical for active power reduction. Because the read and write SNM also behave regularly with V_{DD} scaling, we can use a similar method to derive a $V_{\min}$ model for read and write. Second, quick and accurate DRV prediction at different temperature and aging conditions is desired. A straightforward way is to run SNM simulations at different conditions and directly use our current model to estimate DRV with the new model parameters (μ, σ, and k) that reflect changes induced by those effects. Besides the results at room temperature described in Section V, we tested our model at two extreme temperatures, $-50\,^{\circ}$C and $100\,^{\circ}$C. The model shows errors of less than 2.1% relative to the standard MC for DRV estimates within 4σ at those two extreme temperatures. To save more simulation time, we can further extend the model to predict the impact of temperature and aging on DRV. From our preliminary simulation, we observed that the change of SNMH mean is approximately linear with temperature or NBTI induced V_T shift at a given V_{DD} point. The standard deviation of SNMH might also change with those effects. Moreover, the sensitivity of SNMH to those effects might alter when V_{DD} is lowered. To derive an accurate DRV model incorporating temperature and aging effects, thorough and comprehensive investigations of the sensitivity of SNMH to those effects across the voltage range are needed. Finally, the DRV estimation from the two proposed methods should be verified with silicon measurements and be evaluated in smaller technologies. Fig. 11 has demonstrated that both models closely match with MC and maintain excellent agreement with each other in a commercial 45 nm process. We expect that they can maintain high accuracy and significant speedup beyond 45 nm because both can learn process and design related information from a small MC simulation.

References

[1] 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 Trans. Electron Devices*, vol. 50, no. 9, pp. 1837–1852, Sep. 2003.

[2] D. Hocevar, M. Lightner, and T. Trick, "A study of variance reduction techniques for estimating circuit yields," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 2, no. 3, pp. 180–192, Jul. 1983.

[3] R. Kanj, R. Joshi, and S. Nassif, "Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare event failures," in *Proc. IEEE/ACM Des. Autom. Conf.*, 2006, pp. 69–72.

[4] T. C. Hesterberg, *Advances in Importance Sampling*. Stanford, CA: Department of Statistics, Stanford University, 1998.

[5] L. Dolecek, M. Qazi, D. Shah, and A. Chandrakasan, "Breaking the simulation barrier: SRAM evaluation through norm minimization," in *Proc. Int. Conf. Comput.-Aided Des.*, 2008, pp. 322–329.

[6] C. Gu and J. S. Roychowdhury, "An efficient, fully nonlinear, variability-aware non-Monte-Carlo yield estimation procedure with applications to SRAM cells and ring oscillators," in *Proc. Asia S. Pacific Des. Autom. Conf.*, 2008, pp. 754–761.

[7] H. Qin, Y. Cao, D. Markovic, A. Vladimirescu, and J. Rabaey, "SRAM leakage suppression by minimizing standby supply voltage," in *Proc. ISQED*, 2004, pp. 55–60.

[8] 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 *Proc. Des. Autom. Test Eur.*, 2007, pp. 1–6.

[9] J. Wang, A. Singhee, R. A. Rutenbar, and B. H. Calhoun, "Statistical modeling for the minimum standby supply voltage of a full SRAM array," in *Proc. Eur. Solid State Circuits Conf.*, 2007, pp. 400–403.

[10] 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 *Proc. Int. Conf. VLSI Des.*, 2008, pp. 131–136.

[11] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy caches: Simple techniques for reducing leakage power," in *Proc. Symp. Comput. Architecture*, May 2002, pp. 148–157.
[12] K. Kanda, T. Miyazaki, M. K. Sik, H. Kawaguchi, and T. Sakurai, "Two orders of magnitude leakage power reduction of low voltage SRAMs by row-by-row dynamic Vdd control (RRDV) scheme," in *Proc. IEEE Int. ASIC/SoC Conf.*, Sep. 2002, pp. 381–385.
[13] T. Enomoto, Y. Oka, and H. Shikano, "A self-controllable voltage level (SVL) circuit and its low-power high-speed CMOS circuit applications," *IEEE J. Solid-State Circuits*, vol. 38, no. 7, pp. 1220–1226, Jul. 2003.
[14] N. S. Kim, K. Flautner, D. Blaauw, and T. Mudge, "Single-VDD and single-VT super-drowsy techniques for low-leakage high-performance instruction caches," in *Proc. ISLPED*, 2004, pp. 54–57.
[15] A. J. Bhavnagarwala, S. V. Kosonocky, S. P. Kowalczyk, R. V. Joshi, Y. H. Chan, U. Srinivasan, and J. K. Wadhwa, "A transregional CMOS SRAM with single, logic V_{DD} and dynamic power rails," in *Proc. Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2004, pp. 292–293.
[16] Y. Wang, H. Ahn, U. Bhattacharya, T. Coan, F. Hamzaoglu, W. Hafez, C.-H. Jan, R. Kolar, S. Kulkarni, J. Lin, Y. Ng, I. Post, L. Wel, Y. Zhang, K. Zhang, and M. Bohr, "A 1.1 GHz 12 μA/Mb-leakage SRAM design in 65 nm ultralow-power CMOS with integrated leakage reduction for mobile applications," in *Proc. IEEE ISSCC Dig. Tech. Papers*, Feb. 2007, pp. 324–606.
[17] E. Seevinck, F. List, and J. Lohstroh, "Static-noise margin analysis of MOS SRAM cells," *IEEE J. Solid-State Circuits*, vol. 22, no. 5, pp. 748–754, Oct. 1987.
[18] J. A. Gubner, *Probability and Random Processes for Electrical and Computer Engineers*. Cambridge, MA: Cambridge University Press, 2006.
[19] A. A. Balkema and L. de Haan, "Residual life time at great age," *Ann. Prob.*, vol. 2, no. 5, pp. 792–804, 1974.
[20] A. Singhee, "Novel algorithms for fast statistical analysis of scaled circuits," Ph.D. dissertation, Dept. of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 2007.
[21] A. Singhee and R. A. Rutenbar, "Statistical blockade: Very fast statistical simulation and modeling of rare circuit events, and its application to memory design," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 28, no. 8, pp. 1176–1189, Aug. 2009.
[22] A. Singhee and R. A. Rutenbar, "Beyond low-order statistical response surfaces: Latent variable regression for efficient, highly nonlinear fitting," in *Proc. IEEE/ACM Des. Autom. Conf.*, 2007, pp. 256–261.
[23] L. Chang, D. M. Fried, J. Hergenrother, J. W. Sleight, R. H. Dennard, R. K. Montoye, L. Sekaric, S. J. McNab, A. W. Topol, C. D. Adams, K. W. Guarini, and W. Haensch, "Stable SRAM cell design for the 32 nm node and beyond," in *Proc. Int. Symp. VLSI Tech.*, 2005, pp. 128–129.
[24] R. V. Hogg and A. T. Craig, *Introduction to Mathematical Statistics*, 3rd ed. New York: MacMillan, 1971.
[25] J. R. M. Hosking and J. R. Wallis, "Parameter and quantile estimation for the generalized Pareto distribution," *Technometrics*, vol. 29, no. 3, pp. 339–349, 1987.

Jiajing Wang (S'06–M'10) received the B.S. degree from the East China University of Science and Technology, Shanghai, China, in 2000, the M.S. degree from Fudan University, Shanghai, in 2003, and the Ph.D. degree from the University of Virginia, Charlottesville, in 2010, all in electrical engineering.

From 2003 to 2005, she was with the IC Design Center, Agere Systems, Inc., Shanghai. In 2007, she was a Graduate Intern with Freescale Semiconductor, Austin, TX, where she worked on low-power SRAM design. In March 2010, she joined the Advanced Design Group, Intel Corporation, Hillsboro, OR, as a Design Engineer. Her current research interests include SRAM design and modeling for nanometer complementary metal-oxide-semiconductor technologies, low power and variation tolerant digital circuits, and sub-threshold digital circuits.

Amith Singhee (S'06–M'09)received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2007.

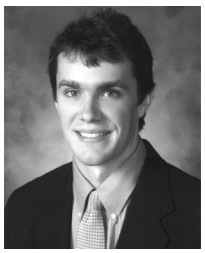
He was with Neolinear, Inc., Pittsburgh, and subsequently with Cadence Design Systems, Inc., San Jose, CA, from 2002 to 2004. He is currently a Research Staff Member with the IBM T. J. Watson Research Center, Yorktown Heights, NY. His current research interests include statistical simulation, process variation modeling, and general design for manufacturability.

Dr. Singhee is a recipient of several awards, including the European Design Automation Association Outstanding Dissertation Award, the A. G. Milnes Award for his Ph.D. dissertation, the Silver Medal at IIT, the Best Paper Award at the Design Automation Conference in 2002 and the Design, Automation and Test in Europe in 2007, and the Best Student Paper Award at the International Conference on Very Large Scale Integration Design in 2008. His paper was published in the book *The Most Influential Papers of 10 Years DATE*, and he received the Inventor Recognition Award from the Global Research Consortium in 2008.

Rob A. Rutenbar (S'77–M'84–SM'90–F'98) received the Ph.D. degree from the University of Michigan, Ann Arbor, in 1984.

From 1984 to 2009, he was a Faculty Member with Carnegie Mellon University, Pittsburgh, PA, where he held the Stephen J. Jatras (E'47) Chair in Electrical and Computer Engineering. In 2010, he joined the University of Illinois at Urbana-Champaign, Urbana, where he is currently the Abel Bliss Professor and Head of Computer Science. He has worked on tools for custom circuit synthesis and optimization for over 25 years. He has published over 150 papers in his career, and his work has been featured in venues ranging from *EE Times* to the *Economist Magazine*.

Dr. Rutenbar has received many awards throughout his career. He has received several Best Paper Awards at the Design Automation Conferences 1987, 2002, and 2010, Design, Automation and Test in Europe 2007, and the International Conference on VLSI Design 2008. He was the 2001 winner of the Semiconductor Research Corporation Aristotle Award for Excellence in Education and the 2007 winner of the IEEE Circuits and Systems Industrial Pioneer Award for his work in making analog synthesis a commercial technology. He is a Fellow of the ACM.



Benton H. Calhoun (M'02) received the B.S. degree in electrical engineering from the University of Virginia, Charlottesville, in 2000, and the M.S. and Ph.D. degrees in electrical engineering from the Massachusetts Institute of Technology, Cambridge, in 2002 and 2006, respectively.

In January 2006, he joined the faculty at the University of Virginia as an Assistant Professor with the Department of Electrical and Computer Engineering. He has published over 50 refereed papers and is a co-author of *Sub-Threshold Design for Ultra Low-Power Systems* (Berlin, Germany: Springer, 2006). His current research interests include low power digital circuit design, sub-threshold digital circuits, SRAM design for end-of-the-roadmap silicon, variation tolerant circuit design methodologies, low power configurable logic, and low energy electronics for medical applications.

Dr. Calhoun is the recipient of a Best Paper Award at the International Conference on VLSI Design 2008.