SRAM Dynamic Stability: Theory, Variability and Analysis

Wei Dong
Department of ECE
Texas A&M University
College Station, TX 77843
weidong@neo.tamu.edu

Peng Li
Department of ECE
Texas A&M University
College Station, TX 77843
pli@neo.tamu.edu

Garng M. Huang
Department of ECE
Texas A&M University
College Station, TX 77843
ghuang@neo.tamu.edu

ABSTRACT

Technology scaling in sub-100nm regime has significantly shrunk the SRAM stability margins in data retention, read and write operations. Conventional static noise margins (SNMs) are unable to capture nonlinear cell dynamics and become inappropriate for state-of-the-art SRAMs with shrinking access time and/or advanced dynamic read-write-assist circuits. Using the insights gained from rigorous nonlinear system theory, we define the much needed SRAM dynamic noise margins (DNMs). The newly defined DNMs not only capture key SRAM nonlinear dynamical characteristics but also provide valuable design insights. Furthermore, we show how system theory can be exploited to develop CAD algorithms that can analyze SRAM dynamic stability characteristics three orders of magnitude faster than a brute-force approach while maintaining SPICE-level accuracy. We also demonstrate a parametric dynamic stability analysis approach suitable for low-probability cell failures, leading to three orders of magnitude runtime speedup for yield analysis under high-sigma parameter variations.

1. INTRODUCTION

Static random access memories (SRAMs) provide indispensable on-chip data storage and continue to dominate the silicon area in many applications. It is projected that more than 90% of silicon real estate will be occupied by SRAM in the future [1]. However, technology scaling in sub-100nm regime has significantly shrunk SRAM stability margins in data retention (standby mode), read and write [1–3]. At the same time, the susceptibility to single event upsets (SEU) induced soft errors continues to cause concerns [4].

The SRAM performance and its variability have been extensively studied in the past [5–9]. In [2, 10], either simulation based first-order models or closed-form performance models using simple transistor models are employed to derive SRAM statistical performance distributions. Mixture important sampling is applied to speedup Monte-Carlo simulation to more efficiently capture rare failure events [11]. The same objective is approached by combining extreme value statistics theory and data filtering in [12]. Euler-Newton curve tracing is proposed to find the boundary between the success and failure regions for read access time yield analysis in [13]. A semi-analytical SRAM dynamical stability model is proposed in [14], where approximated circuit equations based on simple device models are solved in time domain. The use of piecewise linearization of circuit equations, however, can lead to inaccuracy and the challenging issue of process variations is not addressed.

It is important to note that stability occupies a central role in SRAM operations. While the stability in standby or hold implies proper data retention under SEUs and noise injection, stabilities in read and write correspond to non-destructive reads and successful writes, respectively. However, the widely used static noise margins (SNMs) [5] cannot capture the fundamental nonlinear dynamics upon which SRAM cells operate. Although simple to obtain, SNMs assume the DC operation condition and are used with the assumption that timing events have infinite time duration. Due to the intrinsic complexity, dynamic noise margins are researched to a much less extent. In many ways, they may not have been defined rigorously. However, dynamic stability metrics are strongly desirable because of the following reasons. SEU and noise induced soft error analysis requires an understanding of the duration, amplitude, and charge of the injected noise and their interactions with the nonlinear SRAM cell dynamics. Practically, reads and writes behave in an increasingly dynamic fashion in state-of-the-art SRAM designs with shrinking access cycle time and/or read-write-assist circuitry [3, 15]. In the latter case, well timed wordline pulses and write schemes are employed to enhance the read and write margins. Successful reads and writes depend on precise timing control where the nonlinear dynamics of SRAM cells plays a critical role. In design, balance must be made between conflicting static and dynamic stability margins in hold, read and write while considering their variability.

In this work, we start by developing an understanding on the basic nonlinear dynamics of SRAM cells using rigorous nonlinear system theory. In particular, we employ the notion of stability boundary, or separatrix [16,17], and show its central role in determining SRAM dynamic stability. Using separatrix, new dynamic noise margins (DNMs), in a way relevant to basic SRAM operations, are defined. The new DNMs not only characterize the fundamental system characteristics behind SRAM operations, but also provide valuable design insights by connecting dynamic stability with key design parameters. Interestingly, the conventional SNMs are *special* cases of our more general DNMs.

To embody our system concepts and DNM metrics into a practical CAD tool with SPICE-level accuracy, we show how efficient system-theoretically motivated CAD algorithms can be developed. By exploiting nonlinear system theory, a fast separatrix tracing algorithm for SRAM cells under device mismatch is developed. The entire separatrix can be efficiently computed by running two *special* transistor-level transient simulations, achieving three orders of magnitude speedup over the brute-force approach. With this as a basis, we further present a parametric DNM analysis approach, which for a given DNM performance target identifies the acceptance regions in the parameter space. The approach employs Newton method based acceleration and allows for efficient dynamic stability yield analysis over wide range of parametric variations, speeding up expensive high-sigma Monte-Carlo simulation by three orders of magnitude.

2. BACKGROUND

Conventional SNMs in hold and read are shown in Fig. 1. In hold, the conventional static noise margin (SNM) is determined as the side of the largest square that can be inscribed between the mirrored DC voltage transfer curves (VTCs) of the cross-coupled inverters [5]. This measure corresponds to the largest differential voltage noise that can be tolerated at the two storage nodes. SNM in read can be defined similarly by including the two access transistors as part of the inverter pair VTCs. SNM in read represents the largest DC voltage perturbation that can be tolerated without a state flip, or a destructive read. During write, one bit line is discharged and the other is pre-charged. The two inverter VTCs do not form any enclosed region. SNM in write is found by inscribing the largest square in between the two VTCs. To underscore the importance of dynamic stability, we examine the correlations between the SNM and our new DNM in hold as outlined in Section 4.3. The results are shown in Fig. 2. Totally 600 random samples of a 6T SRAM cell are taken, where independent Gaussian transistor threshold voltage variations are assumed. No strong correlation between the SNM and DNM is observed. Hence, in practice, it is difficult to determine whether a better SNM always corresponds to an improved DNM. The SNM and DNM must be both analyzed to provide design guidance.

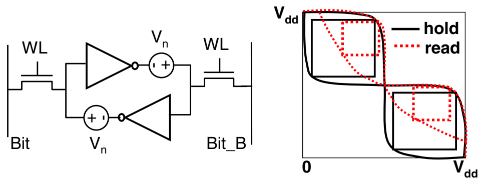


Figure 1: Static noise margins in hold and read.

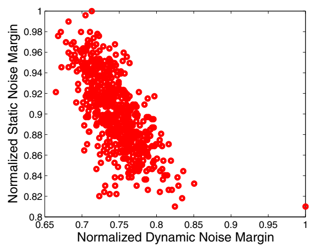


Figure 2: Correlation between SNM and DNM.

3. STABILITY BOUNDARY: SEPARATRIX

An SRAM cell can be described using the MNA equations

$$f(x(t)) + \frac{d}{dt}q(x(t)) + u(t) = 0, \tag{1}$$

where $x(t) \in R^N$ is the vector of nodal voltages and branch currents, $u(t) \in R^N$ is the input, $f(\cdot)$ and $q(\cdot)$ are nonlinear functions describing static and dynamic nonlinearities. Without loss of generality, the simple 2D case ($N = 2$) is focused to simplify the visual presentation. As a nonlinear dynamical system, a cell has three equilibria, which satisfy

$$f(x_e) = 0. \tag{2}$$

Two of them are stable and denoted as x_e^s, corresponding to the *zero* and *one* states of the cell. The third one, denoted as x_e^u, is unstable and also referred to as the saddle. The stable manifold of an equilibrium x_e is defined as [16]

$$W^s(x_e) = \{x \in R^N | \lim_{t\to\infty} \phi(t,x) = x_e\}, \tag{3}$$

where $\phi(t, x)$ is the state trajectory that starts from x and converges to x_e.

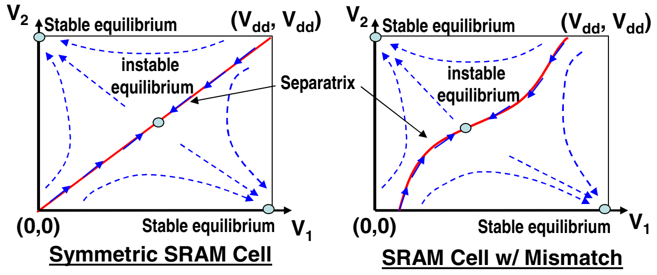


Figure 3: The separatrix of an SRAM cell: symmetric cell vs. cell with mismatch.

To understand SRAM dynamic stability, we consider how the SRAM state transits from one stable equilibrium to the other (i.e. a state flip) under an SEU, read or write event. The *stability region*, or *region of attraction*, of a stable equilibrium is defined to be its stable manifold. There exists a stability region for each stable equilibrium in the state space of a cell. Starting from any initial state in the stability region, the transient SRAM state trajectory will be attracted towards to the corresponding stable equilibrium. The boundary between two stability regions, or the *separatrix*, splits the entire state space. The importance of the separatrix lies in that a state flip will be generated if and only if the amplitude and duration of the disturbance to the cell are high and long enough so that the state is pushed away from the initial stable equilibrium to cross the separatrix. In Fig. 3, the separatrix of a symmetric cell is contrasted with that of an asymmetric cell with device mismatch across the two cross coupled inverters. The vector field (time derivatives of the state variables) is shown by arrowed lines. While the separatrix of the former is along the well expected 45° line, the one of the latter is distorted by mismatch.

4. DYNAMIC NOISE MARGINS

Using the concept of stability boundary, new dynamic noise margins (DNMs) are defined for read, write and hold.

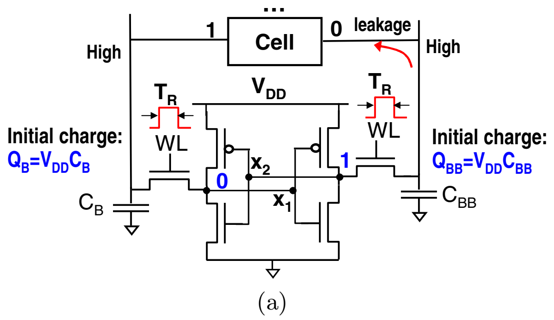
4.1 Dynamic Noise Margin in Read

The read DNM is defined for a given wordline pulse width T_R. To obtain the read DNM, the circuit setup corresponding to the read operation in Fig. 4 (a) is analyzed. Before the read starts, the bit and bit-bar lines are modeled as two fully charged line capacitances. Beside the cell being read, other unselected cells in the same column may draw leakage currents from either the bit or bit-bar lines depending on the stored value. These leakage contributions can be properly modeled and included to capture the inter-cell interaction.

First, the separatrix of the cell in hold, i.e. when the two access transistors are off, is analyzed (e.g. using the algorithm described in Section 5). Then, starting from an initial *zero* or *one* state, the transient state trajectory of the cell with the access transistors turned on (Fig. 4 (a)) is simulated. The time it takes for the trajectory to reach the separatrix is denoted as T_{across}. As shown in Fig. 4 (b), the read DNM is defined as

$$T_{DNM,R} = T_{across} - T_R. \tag{4}$$

This is an appropriate dynamic stability metric as follows. As the read process proceeds (the access transistors remain on), the cell state may be pushed away from its initial state towards the separatrix of the cell with the access transistors off. Note that after the wordline goes off, the read operation ends and the cell returns to hold. Hence, if the trajectory does indeed go across the separatrix, a state flip will be generated after the access transistors are turned off in hold.



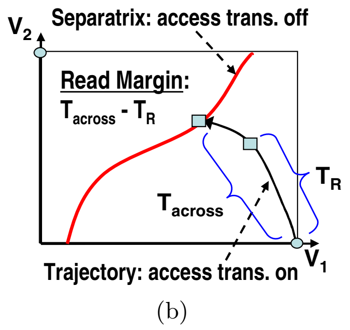


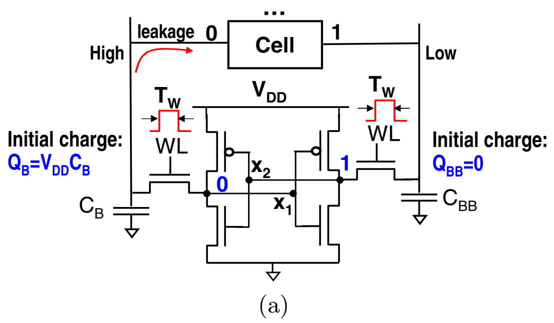
Figure 4: Read dynamic noise margin: a) circuit setup, and b) definition of read DNM.

The defined read DNM specifies the amount of read operation time margin before read instability takes place. That is, when $T_{across} > T_R$, there exists a positive margin; when $T_{across} = T_R$, the cell is on the verge of read instability; when $T_{across} < T_R$, state-flip happens and the cell loses read stability. As can be seen, $T_{DNM,R}$ decreases as T_R, the pulse width of the wordline signal, increases. It is easy to see that when $T_R = \infty$, our read DNM will correctly predict the static read stability, i.e. whether or not the state will be flipped if the duration of the read is infinite. On the other hand, if the conventional SNM is used to evaluate the cell read stability when $T_R < \infty$, which corresponds to the normal read operation, a *pessimistic* estimate may be produced. That is, even if the SNM predicts a state flip, but in reality, it may not happen.

4.2 Dynamic Noise Margin in Write

The write DNM can be defined in a way analogous to that of the read DNM, but by noting that a successful write overwrites the SRAM state, hence producing a state flip. The cell in write mode is analyzed using the circuit setup in Fig. 5 (a). One of the bit and bit-bar lines are discharged before the write starts. Hence, the initial voltage of one bit/bit-bar line capacitor is set to zero. Similar to the previous case, transient analysis is performed to compute the time, T_{across}, at which the state trajectory crosses the separatrix of the hold mode. For a given wordline pulse width, T_W, the write DNM is defined

$$T_{DNM,W} = T_W - T_{across}. \tag{5}$$



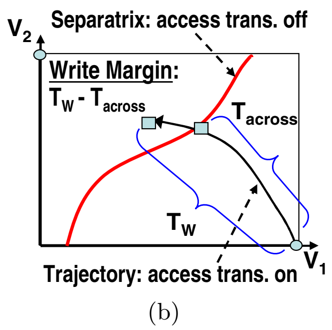


Figure 5: Write dynamic noise margin: a) circuit setup, and b) definition of write DNM.

When T_W is set to ∞, our DNM can correctly predict the static write-ability. On the other hand, when $T_W < \infty$, which corresponds to the normal write operation, however, the SNM *may* provide an *optimistic* estimate for dynamic write-ability. That is, even if the SNM predicts a successful write, in the reality, the write can actually fail. For the state-of-the-art SRAM designs with short access cycles and advanced read/write timing control circuitry, the distinctions between the SNMs and DNMs in read and write reveal the important role of cell nonlinear dynamics in determining dynamic SRAM stability.

4.3 Dynamic Noise Margin in Hold

The DNM in hold characterizes the data retention property of the cell under SEUs and noisy operating condition. Due to the scope limitation, the hold DNM is only briefly introduced. As in Fig. 6, the DNM may be examined by injecting a current disturbance into the cell. Compared with the use of noise voltage disturbance in the SNM [5], modeling the disturbance as an injected current more physically reflects the nonlinear dynamic nature of the cell. The DNM shall be evaluated by considering both the amplitude and duration of the current disturbance. Depending on these two factors, the state trajectory in hold may cross the separatrix, leading to instability. As in read, the conventional

SNM may provide a *pessimistic* stability estimate in hold when blindly applied under dynamic noise injection.

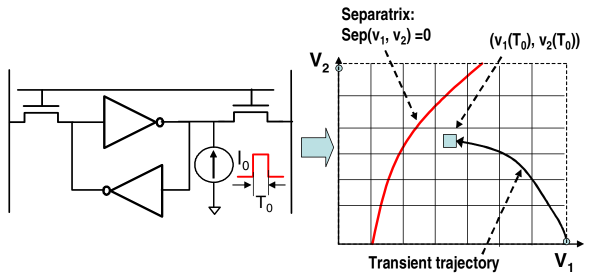


Figure 6: Hold dynamic noise margin.

5. TRACING SEPARATRIX

Consider a brute-force approach for finding the separatrix in hold as shown in Fig. 7 (a). The entire state space of the SRAM cell is sampled using a dense grid. Then, each grid point is used as an initial condition in transistor-level transient simulation. Starting from each initial condition, the SRAM state trajectory may be attracted eventually to one of the stable equilibria, i.e., the sampled point (initial condition) in the state space falls into the stability region of the corresponding equilibrium. If a large number of samples are taken, the separation between points falling into the two stability regions forms the separatrix. The number of transient runs required in this approach, however, makes it time consuming. The high cost of the brute-force approach makes the statistical dynamic stability analysis, where parameter variations are considered, even more difficult.

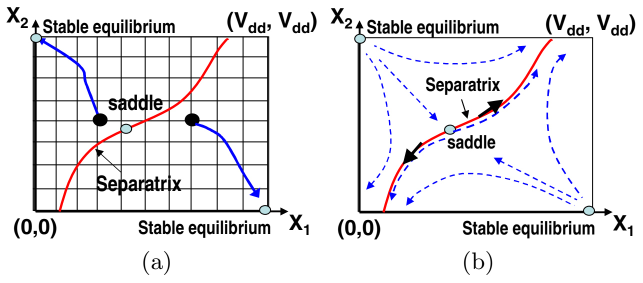


Figure 7: Finding the separatrix: a) brute-force dense sampling, and b) fast stable-manifold tracing.

To derive a much more efficient system-theoretical alternative, denote the stability boundary of the two stability regions as ∂A. For SRAM cells, the following stability boundary theorem exists [16,17]:

THEOREM 1. *The separatrix (stability boundary) of the two stable equilibria is the stable manifold of the unstable equilibrium (saddle point), i.e.,*

$$\partial A = W^s(x_e^u), \tag{6}$$

where x_e^u is the unstable equilibrium on the boundary of ∂A.

Importantly, *Theorem* 1 implies that to find the stability boundary, one shall identify the saddle on the stability boundary and then find its stable manifold. From the view point of Stable Manifold Theory [18], the stable eigenvector of the linearized system at an equilibrium is tangent to its corresponding stable manifold. This implies that by starting in a neighborhood of x_e^u, the desired stable manifold can be found by integrating the system equations along the direction of the stable eigenvector backward in time. Integration backward in time prevents the state trajectory to converge to any of the equilibrium points and enforces the trajectory to stay on the stability boundary.

The above theoretical results provide a foundation for the following highly efficient separatrix tracing algorithm.

Algorithm 1 Fast Stable-Manifold Separatrix Tracing

```
1: Compute the equilibria ($x_e^s$ and $x_e^u$) for the cell;
2: Find the stable eigenvector, $u_s$, for the saddle $x_e^u$: the stable
   eigenvector corresponds to the stable eigenvalue at the saddle;
3: Choose two initial conditions:
   $x^0_{\{1,2\}} = x_e^u \pm \varepsilon u_s$, $\varepsilon$ is small;
4: Perform two transient simulations from initial conditions
   $x^0_{\{1,2\}}$ for the modified dynamical system:
       $f(x(t)) - \frac{d}{dt}q(x(t)) = 0;$   (7)
5: Output the separatrix as:
       $\partial A = \{x | x = \phi_M(t, x_1^0)$ or $\phi_M(t, x_2^0), t = [0, T_{max}]\},$   (8)
   where $\phi_M$ is the state trajectory of the modified system, and
   $T_{max}$ is the maximum duration of the two transient runs.
```

The tracing algorithm is illustrated in Fig. 7 (b). In contrast to the original system in (1), integration backward in time is reflected by negating the differential term in the modified system in (7). Intuitively, negating the differential term reverses the vector field, as shown in the dashed arrow lines in Fig. 7 (b) (compare with the original field in Fig. 3). Note that along the separatrix, the direction of the vector field points away from x_e^u. Elsewhere in the state space, the vector field points towards the separatrix. Since the two transient simulations do not start from equilibrium x_e^u, but from a neighborhood of it, the state trajectories will move. Due to the new vector field, they do not converge to the saddle nor the two stable equilibrium points, but are actually forced to stay on the separatrix, making it possible to efficiently find the entire separatrix by using only two transient runs. In practice, step 2 of the algorithm can be bypassed as long as the two initial conditions are perturbed from the saddle along opposite directions. Here, the exact stable eigenvector u_s is not required since the unstable components dissipate fast as we integrate backward in time [16,18]. Another practical issue is to find the saddle. Usually the saddle can be found directly in one DC analysis with suitable choice of the initial guess and convergence control. We have also found that the saddle could be found rather reliably through one transient simulation of the modified dynamic system (7) starting from an initial condition.

6. DYNAMIC STABILITY YIELD

The proposed separatrix tracing algorithm enables DNM analysis through efficient computation of the separatrix, which is central to dynamic stability. Nevertheless, statistical SRAM analysis via direct Monte-Carlo simulation may be still prohibitively expensive. Large SRAM memories may have tens of millions of or even more cells. To achieve a good yield for the entire SRAM system, each SRAM cell must be designed to tolerate wide range of parametric variations such as random dopant fluctuation induced threshold voltage variation. Cell-level failures are rare events with very low probability and accurate yield estimate requires a very large number of Monte-Carlo samples. To further improve the efficiency of dynamic stability yield analysis, we proposed a Newton method accelerated parametric analysis technique.

6.1 Problem Formulation

Without loss of generality, the proposed yield analysis is described using the read DNM as an example. For a given wordline pulse width T_R and a target read DNM T_{target}, using (4), the read DNM yield is given as the following probability

$$\begin{aligned} Y_{DNM,R} &= Pr\{T_{DNM,R}(\vec{p}) \geq T_{target}\} \\ &= Pr\{T_{across}(\vec{p}) \geq T_{target} + T_R\} \\ &= Pr\{T_{across}(\vec{p}) \geq T_{across,0}\}, \end{aligned} \tag{9}$$

where $\vec{p} \in R^M$ is a set of M parameter variations with $\vec{p} = 0$ corresponding to the nominal condition, and $T_{across,0} = T_{target} + T_R$ is a constant. Instead of performing Monte-Carlo simulation, DNM yield can be computed more efficiently by identifying the acceptance region in the parameter space, Ω_{accept}, i.e., the parameter subspace that corresponds to SRAM cells meeting the DNM specification T_{target}. $Y_{DNM,R}$ can be then simply computed as the probabilistically weighted volume or area of the acceptance region

$$Y_{DNM,R} = \int \cdots \int_{\vec{p}} g(\vec{p}) f(\vec{p}) d\vec{p},\ g(\vec{p}) = \begin{cases} 1 & \text{if } \vec{p} \in \Omega_{accept} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where $f(\cdot)$ is the PDF of $\vec{p}$.

For ease of presentation, the identification of the acceptance region in a two-dimensional parameter space ($M = 2$) is considered, which is shown in Fig. 8 (a). From the nominal design at the center of the acceptance region, four initial directions are selected to search for the boundary of the acceptance region, $\partial\Omega_{accept}$, in each quadrant

$$\partial\Omega_{accept} = \{\vec{p} \mid T_{across}(\vec{p}) = T_{across,0}\}. \tag{11}$$

Possible initial directions can be along the 45°, 135°, 225° and 315° lines. It shall be noted that, in principle, it is possible that the search along a direction does not reach any boundary, indicating the acceptance region is not closed.

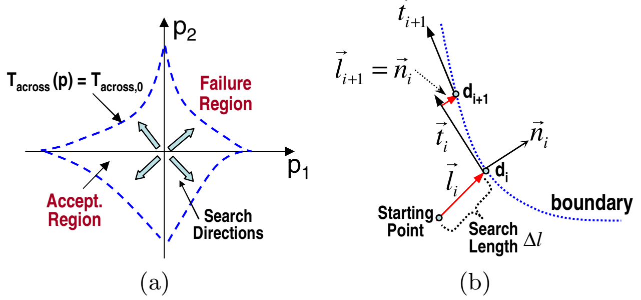


Figure 8: DNM yield analysis: a) find the acceptance region boundary, b) search along a direction.

6.2 Acceleration via Newton Method

The search for $\partial\Omega_{accept}$ is detailed in Fig. 8 (b), where the search direction is adapted. A complete search cycle is shown in the figure. Starting from an initial point in the parameter space, a search along a unit-length vector $\vec{l_i}$ is initiated to find a point d_i at $\partial\Omega_{accept}$. Then, the tangent direction, $\vec{t_i}$, and normal direction, $\vec{n_i}$, of the acceptance region boundary are evaluated at d_i. The search moves over a short distance along $\vec{t_i}$. To move back to $\partial\Omega_{accept}$, the search continues along the new search direct $\vec{l}_{i+1}$, which is set to be $\vec{n_i}$. The procedure repeats till a set of points on $\partial\Omega_{accept}$ are found. This general principle of acceptance region boundary tracing is similar to the idea in [13]. In our case, however, the search direction adaption as described above is used to ensure a reliable and fast tracing of $\partial\Omega_{accept}$. More importantly, complications arisen specifically from the dynamic stability analysis must be properly handled as below.

The key step in the above procedure is to find a point on the boundary along search direction $\vec{l}$ starting from an initial point $\vec{p_0}$. Such $\vec{p}$ satisfies the line equation $\vec{p} = \vec{p_0} + \Delta l \cdot \vec{l}$, where Δl is the search length. For $\vec{p} \in \partial\Omega_{accept}$, the nonlinear scalar equation in Δl must be satisfied

$$\hbar(\Delta l) \equiv T_{across}(\vec{p_0} + \Delta l \cdot \vec{l}) - T_{across,0} = 0. \tag{12}$$

To speedup the solution of the above equation using Newton method, the key task is to compute the derivative $\partial\hbar(\Delta l)/\partial\Delta l$, which is the focus of the subsequent discussion.

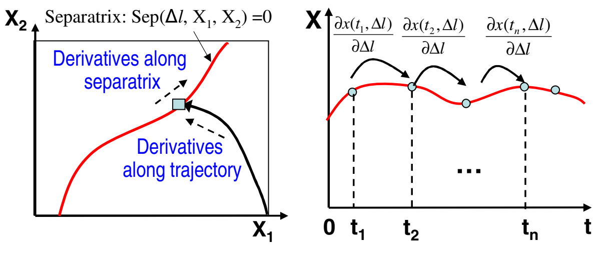
6.2.1 Derivation of Newton Derivatives

The separatrix in hold may be specified as a general nonlinear separatrix line equation in state variables $x = [x_1, x_2]^T$ and the parameter variation Δl

$$Sep(\Delta l, x_1, x_2) = 0. \tag{13}$$

In practice, after the separatrix is computed through the proposed tracing algorithm, it can be represented as a piecewise linear function connecting the points along the tracing trajectories. For any given guess for Δl, T_{across} can be computed by performing a transient simulation of the cell in read from an stable equilibrium initial condition to check whether (12) is satisfied or not. This is shown in Fig. 9 (a). At T_{across}, the transient trajectory satisfies (13), specifically

$$Sep\left(\Delta l, x_1(T_{across}(\Delta l), \Delta l), x_2(T_{across}(\Delta l), \Delta l)\right) = 0. \tag{14}$$



X2
Separatrix: Sep(Δl, X1, X2) =0
Derivatives along separatrix
Derivatives along trajectory
X1
(a)
X
$\frac{\partial x(t_1,\Delta l)}{\partial \Delta l}$
$\frac{\partial x(t_2,\Delta l)}{\partial \Delta l}$
$\frac{\partial x(t_n,\Delta l)}{\partial \Delta l}$
...
0
t1
t2
tn
t
(b)

Figure 9: Derivative computation: a) key derivatives for T_{across}, b) parametric transient sensitivities.

Here, both the state variables and T_{across} depend on Δl. If (12) is not satisfied, $\partial\hbar(\Delta l)/\partial\Delta l = \partial T_{across}/\partial\Delta l$ shall be computed such that Newton method can be used to correct Δl. Differentiating both sides of (14) w.r.t Δl gives

$$\begin{aligned} &\frac{\partial Sep}{\partial \Delta l} + \left(\frac{\partial Sep}{\partial x_1} \frac{\partial x_1}{\partial T_{across}} \frac{\partial T_{across}}{\partial \Delta l} + \frac{\partial Sep}{\partial x_1} \frac{\partial x_1}{\partial \Delta l} \right) + \\ &\left(\frac{\partial Sep}{\partial x_2} \frac{\partial x_2}{\partial T_{across}} \frac{\partial T_{across}}{\partial \Delta l} + \frac{\partial Sep}{\partial x_2} \frac{\partial x_2}{\partial \Delta l} \right) = 0, \end{aligned} \tag{15}$$

or equivalently

$$\begin{aligned}\frac{\partial \hbar(\Delta l)}{\partial \Delta l} &= \frac{\partial T_{across}}{\partial \Delta l} \\ &= -\frac{\frac{\partial Sep}{\partial \Delta l} + \frac{\partial Sep}{\partial x_1}\frac{\partial x_1}{\partial \Delta l} + \frac{\partial Sep}{\partial x_2}\frac{\partial x_2}{\partial \Delta l}}{\frac{\partial Sep}{\partial x_1}\frac{\partial x_1}{\partial T_{across}} + \frac{\partial Sep}{\partial x_2}\frac{\partial x_2}{\partial T_{across}}}. \quad (16)\end{aligned}$$

6.2.2 *Computation of Newton Derivatives*

In (16), $\frac{\partial Sep}{\partial x_1}$ and $\frac{\partial Sep}{\partial x_2}$ are the sensitivities of the separatrix line equation, where the transient trajectory intersects the separatrix. They can be readily computed once the separatrix is traced out using *Algorithm* 1. $\frac{\partial x_1}{\partial T_{across}}$ and $\frac{\partial x_2}{\partial T_{across}}$ are the time derivatives of the transient trajectory, again at the intersection between the transient trajectory and the separatrix (Fig. 9 (a)), hence they are readily available from the simulated transient waveforms.

The key remaining components in (16), $\frac{\partial Sep}{\partial \Delta l}$, $\frac{\partial x_1}{\partial \Delta l}$ and $\frac{\partial x_2}{\partial \Delta l}$, are the derivatives of the separatrix line equation and state trajectory (at T_{across}) w.r.t the parameter variation Δl evaluated at the intersection. Note that in this work the separatrix line equation $Sep(\cdot)$ is formed as a piecewise linear function connecting the points along the traced transient trajectories. Hence, it suffices to compute the sensitivities of the transient trajectory w.r.t Δl and then use the chain rule to finally get $\frac{\partial Sep}{\partial \Delta l}$. Therefore, for any of $\frac{\partial Sep}{\partial \Delta l}$, $\frac{\partial x_1}{\partial \Delta l}$ and $\frac{\partial x_2}{\partial \Delta l}$, what remains to be discussed is to compute transient response parametric sensitivities at certain time instance. It turns out that this can be achieved by accumulating the parametric sensitivities throughout the transient analysis, a technique employed in time-domain shooting method for steady-state RF simulation [13, 19], as shown in Fig. 9 (b).

Consider the parameter form of the MNA equations

$$f(x(t), \vec{p}) + \frac{d}{dt}q(x(t), \vec{p}) + u(t) = 0. \quad (17)$$

Using a standard numerical integration method, say Backward Euler, in transient analysis, (17) is discretized over a set of time points $[t_0, t_1, t_2, \cdots, t_K]$

$$f(x(t_i), \vec{p}) + \frac{q(x(t_i), \vec{p}) - q(x(t_{i-1}), \vec{p})}{t_i - t_{i-1}} + u(t_i) = 0. \quad (18)$$

Differentiating (18) w.r.t a parameter $p_j, p_j \in \vec{p}$, leads to

$$\begin{aligned}\frac{\left[\frac{\partial q}{\partial x}\cdot\frac{\partial x}{\partial p_j} + \frac{\partial q}{\partial p_j}\right]_{t_i} - \left[\frac{\partial q}{\partial x}\cdot\frac{\partial x}{\partial p_j} + \frac{\partial q}{\partial p_j}\right]_{t_{i-1}}}{t_i - t_{i-1}} &+ \\ \left[\frac{\partial f}{\partial x}\cdot\frac{\partial x}{\partial p_j} + \frac{\partial f}{\partial p_j}\right]_{t_i} &= 0. \quad (19)\end{aligned}$$

Define: $G_i = \frac{\partial f}{\partial x}|_{t_i}$, $C_i = \frac{\partial q}{\partial x}|_{t_i}$, $d_i = \frac{\partial f}{\partial p_j}|_{t_i}$, $e_i = \frac{\partial q}{\partial p_j}|_{t_i}$, $s_i = \frac{\partial x}{\partial p_j}|_{t_i}$, and $\Delta t = t_i - t_{i-1}$. (19) can be written as

$$s_i = \left(\frac{C_i}{\Delta t_i} + G_i\right)^{-1}\left(\frac{C_{i-1}s_{i-1}}{\Delta t_i} + \frac{e_{i-1} - e_i}{\Delta t_i} - d_i\right). \quad (20)$$

Note that d_i and e_i can be obtained by computing the sensitivities of device model evaluations. The matrix inversion in (20) can be facilitated by reusing the matrix factorization in the last Newton iteration of the transient simulation for each t_i. Hence, by accumulating the transient response sensitivities starting from time t_0 according to (20), sensitivities at any other time points can be rather efficiently computed as part of the transient analysis. The sensitivities w.r.t. other parameters in $\vec{p}$ can be computed individually in the same fashion. Denote the sensitivity vector of state x_1 (x_2) w.r.t all the parameters at time t_i as $\vec{s}_{1,i}$ ($\vec{s}_{2,i}$), the scaler sensitivity along the search direction $\vec{l}$ is simply $s_{1,i,\vec{l}} = \vec{l}^T \cdot \vec{s}_{1,i}$ ($s_{2,i,\vec{l}} = \vec{l}^T \cdot \vec{s}_{2,i}$).

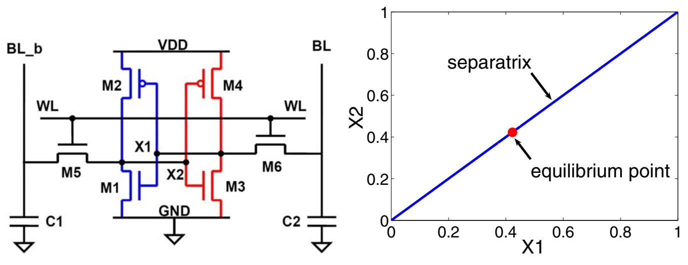
The aforementioned parametric dynamic stability analysis can be extended to include more transistor parameter variations, which amounts to search the acceptance boundaries in a higher dimensional parameter space. The same Newton method acceleration can be applied with a more involved search direction control.

7. EXPERIMENTAL RESULTS

The proposed techniques are implemented using C/C++ as part of a SPICE-like simulation environment on a Linux server with 3.0GHZ clock frequency and 2GB memory. We consider a 6-T SRAM cell structure with 1V supply voltage, as shown in Fig. 10(a), under various parameter settings.

7.1 Separatrix Tracing

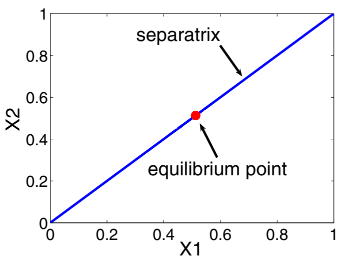
First, consider the case where the cell is fully symmetric. As in Fig. 10(b), the saddle is found to be at (0.4229363V, 0.4229363V). The separatrix is obtained using the proposed separatrix tracing method via two transient runs. As expected, the separatrix is along the 45° line, which verifies the correctness of our algorithm. Next, we consider two cells with transistor parameter variations across the two inverters and show their separatrice in Fig. 11(a) and Fig. 11(b). In the former case, the threshold voltages of N-type transistors $M1$ and $M3$ are both increased by 20% and the threshold voltages of P-type transistors $M2$ and $M4$ are decreased by 20%. The separatrix is still along the 45° line, however, the saddle moves to (0.5123445V, 0.5123445V). For the second case in Fig. 11(b), the effective channel lengths and the threshold voltages of $M1$ and $M2$ are simultaneously decreased by 30% and those of $M3$ and $M4$ are increased by 30%. Due to the mismatch effects, the separatrix is no longer a straight line and the saddle moves to (0.3612088V, 0.4596336V).



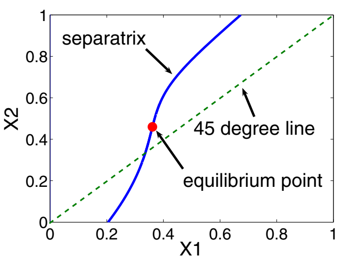
(a) A 6-T SRAM cell (b) separatrix

Figure 10: Separatrix of a symmetric cell.

Since the main cost of our tracing algorithm comes from the two transient runs, the separatrix tracing method is very efficient compared with the brute-force method. The separatrix tracing time is less than 1 minute. But if we run 100x100 brute-force transient simulations to sample the state space to obtain the separatrix at 1% accuracy, the total runtime is about 38 hours. Hence, the proposed method provides more than 2000X speedup. To verify the accuracy of the



(a) Case 1 : V_T variations



(b) Case 2 : V_T and L_{eff} variations

Figure 11: Separatrix under device variations.

tracing algorithm, we randomly select 20 points close to the separatrix in the state space. We run transient simulations by taking these points as initial conditions. Each transient state trajectory ends up at the correct stable equilibrium points without crossing the separatrix.

7.2 Dynamic noise margin analysis

Several DNM analyses for read, write and hold are conducted, where the initial SRAM state is at $x_1 = 1.0V$ and $x_2 = 0V$ in Fig. 10(a).

7.2.1 *Read DNM*

We consider an asymmetric 6-T SRAM cell for read DNM analysis. We first trace the separatrix in hold. In the read mode, with the access transistors being turned on, starting from a stable equilibrium, a transient run is used to compute the time at which the state trajectory crosses the separatrix, or T_{across}. It is found to be $8.71ns$. Then, consider four worldline turn-on times $T_{R1} = 8.72ns$, $T_{R2} = 8.70ns$, $T_{R3} = 8.20ns$, $T_{R4} = 5.00ns$. According to our read DNM definition, the read DNMs for these four cases are $-0.01ns$, $0.01ns$, $0.51ns$ and $3.71ns$, respectively. Next, we use the transient simulations to verify our read DNM results. The simulation trajectories under the four worldline turn-on times are shown in Fig. 12. As expected, in the first case a state flip is produced, indicated by a negative read DNM. In each of the other three cases, there is no read instability (the state trajectory moves back to the initial stable equilibrium after the read operation ends).

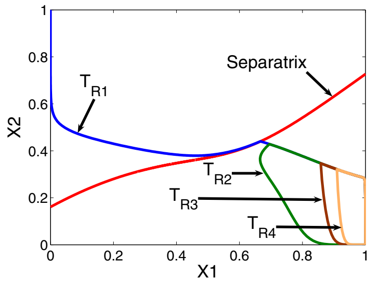


Figure 12: Verification of read DNMs.

7.2.2 *Write DNM*

For the same cell, we further analyze write DNMs when the wordline pulse width is set to be $T_{W1} = 0.038ns$, $T_{W2} = 0.042ns$, $T_{W3} = 0.053ns$, $T_{W4} = 0.065ns$, respectively. For the write, the time for the trajectory to reach the separatrix, T_{across}, is found to be $0.040ns$. Therefore, the write DNMs are $-0.002ns$, $0.002ns$, $0.013ns$ and $0.025ns$, respectively. Again, we use brute-force transient simulations to verify our results, as shown in Fig. 13. It can be seen that in the first case, no state flip is produced, indicating a write failure, which is correctly predicted by our negative DNM margin. In all other three cases, the cell is successfully written, consistent to the computed positive write DNMs.

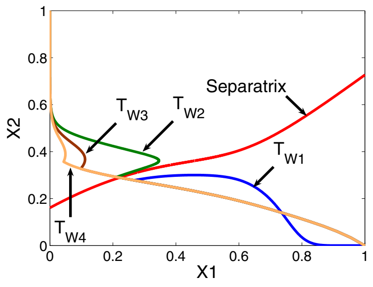


Figure 13: Verification of write DNMs.

7.2.3 *Hold DNM*

To understand the cell stability in hold, a noise current with an amplitude of $135\mu A$ is injected into one of the storage nodes. The duration is varied as: $T_{H1} = 1.685ns$, $T_{H2} = 1.683ns$, $T_{H3} = 1.654ns$, $T_{H4} = 1.460ns$. To characterize the hold DNM, again the separatrix in hold (without noise injection) is traced. Then, with the noise injection, a transient simulation is performed to find the separatrix crossing time T_{across} to be $1.684ns$. The hold DNM may be defined as the difference between T_{across} and each T_H. Given this, for T_{H1}, the hold DNM is negative, suggesting a state flip, which is confirmed by the transient simulation shown in Fig. 14. For each of the other three cases, the hold DNM is positive and hence no state flip happens, as confirmed in the same figure.

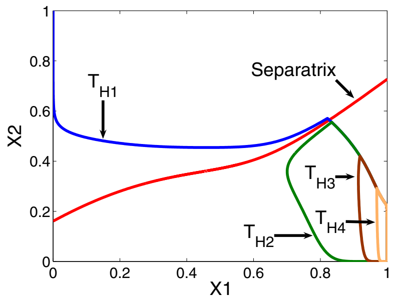


Figure 14: Verification of hold DNMs.

7.3 Parametric DNM analysis

The read and write DNM variations under independent Gaussian random transistor threshold voltage variations are considered. The nominal threshold voltages for NMOS and PMOS transistors are 0.3V and -0.28V, respectively. The initial SRAM state is also at $x_1 = 1.0V$ and $x_2 = 0V$ (Fig. 10(a)).

In the first example, we consider the V_T variations of transistors $M1$ and $M5$ in Fig. 10(a). $T_{target} + T_R$ in (9) is set to be 0.3ns and accordingly the acceptance region boundary is:

$$\partial\Omega_{accept} = \{(V_{th1}, V_{th5}) \mid T_{across}(V_{th1}, V_{th5}) = 0.3ns\}, \tag{21}$$

where V_{th1} and V_{th5} represent the threshold voltages of transistors M1 and M5. The traced boundary is shown in Fig. 15.

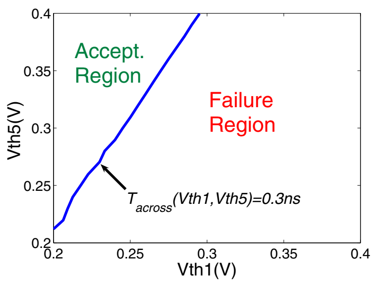


Figure 15: Read DNM acceptance region - case 1.

Next, we consider the variations of the threshold voltages of transistors M3 and M4. The traced boundary is plotted in Fig. 16. To verify the correctness of the traced boundaries,

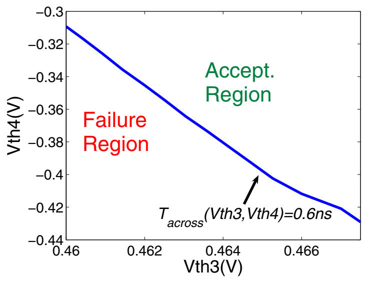


Figure 16: Read DNM acceptance region - case 2.

we select 50 points close to the boundary from each side in the parameter space shown in Fig.15. For each parameter corner, we simulate the SRAM cell to obtain time T_{across}. The simulated T_{across} values of the points left to the boundary are close to $0.3ns$ and larger. The T_{across} values of the points right to the boundary are also close to $0.3ns$, but smaller.

Since we have obtained the boundary of acceptance region, the yield of read DNM can be efficiently evaluated. However, if we use Monte Carlo simulation to evaluate the read DNM yield, to cover the wide range of variations (e.g. 6σ or beyond), the number of samples required can be huge. One Monte-Carlo run on average takes 10 seconds. Assuming one million samples are needed, then a total of 10^7seconds will be required. For the above two cases, the proposed method takes 63 and 57 minutes, respectively, hence providing a runtime speedup of three orders of magnitude.

For statistical write DNM analysis, the threshold voltage variations of M3 and M4 are considered. The specified write DNM is setup so that T_{across} should be at most 0.2ns. The traced boundary, as shown in Fig. 17, is computed with a similar efficiency, by using 71 minutes.

8. CONCLUSIONS

In this work, nonlinear system theory is adopted to provide a basis for rigorous understanding of SRAM dynamic stability. Using the concept of stability boundary, new DNM metrics are defined for read, write and hold. Nonlinear system theory is further exploited to develop fast SPICE-level CAD algorithm for tracing the separatrix. A fast Newton-based DNM yield analysis is also presented.

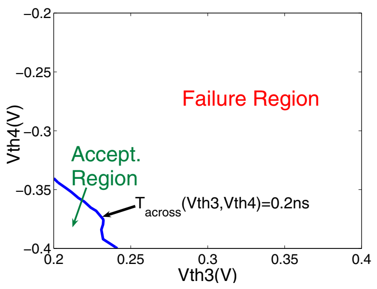


Figure 17: Write DNM acceptance region.

9. REFERENCES

[1] K. Chakraborty and P. Mazumder. *Fault-Tolerance and Reliability Techniques for High-Density Random-Access Memories*. Prentice Hall, 2002.

[2] K. Roy S. Mukhopadhyay, H. Mahmoodi. Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled cmos. *IEEE Trans. CAD*, 24(12):1859–1880, Dec. 2005.

[3] M. Khellah, Y. Ye, N. S. Kim, D. Somasekhar, G. Pandya, A. Farhang, K. Zhang, C. Webb, and V. De. Wordline & bitline pulsing schemes for improving SRAM cell stability in low-vcc 65nm CMOS designs. In *Symp. on VLSI Circuits*, June 2006.

[4] P. C. Murley and G. R. Srinivasan. Soft-error monte carlo modeling, program, SEMM. *IBM J. Res. Develop.*, 40(1):109–118, Jan. 1996.

[5] J. Lohstroh, E. Seevinck, and J. D. Groot. Worst-case static noise margin criteria for logic circuits and their mathematical equivalence. *IEEE J. of Solid-State Circuits*, sc-18(6):803–806, Dec. 1983.

[6] E. Seevinck, F. J. List, and J. Lohstroh. Static-noise margin analysis of MOS SRAM cells. *IEEE J. of Solid-State Circuits*, sc-22(5):748–754, Oct. 1987.

[7] A. J. Bhavnagarwala, X. Tang, and J. D. Meindl. The impact of intrinsic device fluctuations on CMOS SRAM cell stability. *IEEE J. of Solid-State Circuits*, 36(4):658–665, Apr. 2001.

[8] E. Grossar, M. Stucchi, K. Maex, and W. Dehaene. Read stability and write-ability analysis of SRAM cells of nanometer technologies. *IEEE J. of Solid-State Circuits*, 41(11):2577–2588, Nov. 2006.

[9] R. V. Joshi, S. Mukhopadhyay, D. W. Plass, Y. H. Chan, C. Chuang, and A. Devgan. Variability analysis for sub-100 nm PD/SOI CMOS SRAM cell. In *European Solid-State Circuits Conf.*, Sept. 2004.

[10] K. Agarwal and S. Nassif. Statistical analysis of SRAM cell stability. In *IEEE/ACM Design Automation Conf.*, July 2006.

[11] R. Kanj, R. Joshi, and S. Nassif. Mixture importance sampling and its application to the analysis of SRAM designs in the presence of rare failure events. In *IEEE/ACM Design Automation Conf.*, July 2006.

[12] A. Singhee and R. A. Rutenbar. Statistical blockade: A novel method for very fast Monte Carlo simulation for rare circuit events, and its application. In *IEEE/ACM Design, Automation and Test in Europe Conf.*, Apr. 2007.

[13] S. Srivastava and J. Roychowdhury. Rapid estimation of the probability of SRAM failure due to MOS threshold variations. In *IEEE Custom Integrated Circuits Conf.*, Sept. 2007.

[14] B. Zhang, A. Arapostathis, S. Nassif, and M. Orshansky. Analytical modeling of SRAM dynamic stability. In *IEEE/ACM Int. Conf. on Computer-Aided Design*, Nov. 2006.

[15] H. Pilo et al. An SRAM design in 65nm and 45nm technology nodes featuring read and write-assist circuits to expand operating voltage. In *Symp. on VLSI Circuits*, June 2006.

[16] J. Zaborszky, G. M. Huang, B. Zheng, and T. C. Leung. On the phase portrait of a class of large nonlinear dynamic systems such as the power system. *IEEE Trans. Automatic Control*, pages 4–15, Jan. 1988.

[17] G. M. Huang, W. Dong, Y. Ho, and P. Li. Tracing SRAM separatrix for dynamic noise margin analysis under device mismatch. In *IEEE Int. Behavioral Modeling and Simulation Conf.*, 2007.

[18] H. Khalil. *Nonlinear Systems, 3rd Edition*. Prentice Hall, 2002.

[19] K. Kundert, J. White, and A. Sangiovanni-Vincentelli. *Steady-state methods for simulating analog and microwave circuits*. Kluwer Academic Publisher, Boston, 1990.