# An Analytical Approach for Soft Error Rate Estimation in Digital Circuits

Ghazanfar Asadi Mehdi B. Tahoori

Dept. of Electrical & Computer Engineering, Northeastern University
Boston, MA 02115
*{gasadi, mtahoori}@ece.neu.edu*

***Abstract*—Soft errors due to cosmic rays cause reliability problems during lifetime operation of digital systems, which increase exponentially with Moore's law. The first step in developing efficient soft error tolerant schemes is to analyze the effect of soft errors at the system level. In this work, we develop a systematic approach for soft error rate estimation. Experiments on benchmark circuits and comparison of the results with random fault injection (previous work) show that our proposed method is on average 95% accurate while 4-5 orders of magnitude faster.**

## 1. INTRODUCTION

Soft errors, also called *transient* errors, are intermittent malfunctions of the hardware that are not reproducible [1]. Soft errors arise from *Single Event Upsets* (SEU), which are caused by energetic particles (neutrons and alpha particles). *Soft Error Rate* (SER) for a device is defined as the error rate due to SEUs, which depends on both the particle flux and circuit characteristics. Device circuit parameters that influence the error rate include the amount of charge stored, the vulnerable cross-sectional area, and the charge collection efficiency.

Device scaling significantly affects the susceptibility of integrated circuits to soft errors [2]. As the feature size shrinks, the amount of charge per device decreases enabling a particle strike to be much more likely to cause an error. As a result, particles of lower energy, which are far more plentiful, can generate sufficient charge to cause a soft error. Hence, in the absence of error correction schemes, the error rate of vulnerable parts will grow in direct proportion to the number of bits on the chip [3].

So far, memory elements have been more susceptible to soft errors than the combinational logic. However, analytical models predict that the soft error rate in the combinational logic will be comparable to that of memory elements by 2011 [2]. Soft error avoidance techniques such as shielding, Silicon-On-Insulator (SOI), and radiation-hardened can only reduce the effect of soft error while introducing significant amount of area and performance penalty.

The first step in developing soft error tolerant scheme with low cost and performance penalties is to estimate the system failure rate due to soft errors and the contribution of each component to the overall system failure rate. Previous work on SER estimation is based on fault injection using random simulations and hence inaccurate and very time-consuming [1][2][3].

In this paper, we present a new *error propagation probability* (EPP) computation approach for the estimation of system failure due to soft errors at the logic level. The presented approach uses the signal probabilities of all nodes in the combinational part and then computes EPPs based on the topological structure of the circuit. Signal probability calculation is widely used for accurate estimation of signal activity and power dissipation of circuits. By reusing these results from previous design steps, the complexity of our approach will not increase. Experiments on benchmark circuits and comparison of the results with fault injection method based on random simulation show the effectiveness and the accuracy of the presented approach.

The rest of this paper is organized as follows. In Sec. 2, the analytical SER estimation method is described. In Sec. 3, the experimental results are presented. Finally, Sec. 4 concludes the paper.

## 2. ANALYTICAL SER ESTIMATION METHOD

A typical synchronous circuit consists of combinational logic and flip-flops (Fig. 1). Primary Inputs (PIs) and the outputs of flip-flops ($PI_{FF}$) are inputs of combinational logic (CL). Also, Primary Outputs (POs) and the inputs of the FFs ($PO_{FF}$) are outputs of CL.

To compute the error rate of a node in a circuit, three probability factors are required to be computed:

$$R_{SEU}(n_i) \times P_{latched}(n_i) \times P_{sensitized}(n_i)$$

These parameters are defined as follows:

- $R_{SEU}(n_i)$ is the occurrence rate of SEUs at node $n_i$ to cause a glitch at the output of the gate. This parameter depends on the energy of the particle, type and the size of the gate, and device characteristics.
- $P_{latched}(n_i)$ is the probability that an erroneous value reaching the flip-flop inputs is latched.
- $P_{sensitized}(n_i)$ is the probability that node $n_i$ is functionally sensitized by the input vectors to propagate the erroneous value from the error site to POs/FFs.

$R_{SEU}(n_i)$ can be obtained from layout information of library cells, technology parameters, and particle energy [1][4][5]. $P_{latched}(n_i)$ estimation consists of logic and timing derating. *Logic derating* is the probability that an erroneous value is propagated to the input of a flip-flop. *Timing derating* is the probability that there is an overlap between the width of an error glitch and the latching window of a reachable flip-flop. In this work we focus on the estimation of the logic part of $P_{latched}(n_i)$ and $P_{sensitized}(n_i)$. This is based on the fact that the error propagation probability computation is the most time-consuming part of SER estimation. The error sites considered in this paper are all circuit nodes (inputs and output of all gates and FFs).

In the proposed approach, we first extract the structural paths from the error sites to all reachable outputs and then traverse these paths to compute the propagation probability of the erroneous value to the reachable primary outputs or to the reachable flip-flops. Based on the error site, we categorize nets and gates in the circuit as follows. An *on-path* signal is a net on a path from the error site to a reachable output. Also, an *on-path gate* is defined as the gate with at least one on-path input. Finally, an *off-path* signal is a net that is not on-path and is an input of an on-path gate. These three are also shown in Fig. 2.

For error propagation probability calculation, as we traverse the paths, we use signal probability for off-path signals and use our propagation rules for on-path signals. The signal probability (SP) of a line *l* indicates the probability of *l* having logic value 1 [6]. SP

0-7803-8834-8/05/$20.00 ©2005 IEEE.

techniques have been presented in [7] [8]. The problem statement can be described as follows:

*Given the SEU probability in node $n_i$ calculate the probability of the propagation of this error to POs/FFs (i.e., system failure).*

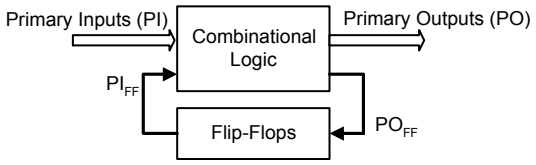


Fig. 1: A typical block diagram of synchronous sequential circuits

Errors can be directly propagated to a primary output and cause a system failure at the same clock cycle, or they can be propagated to flip-flops repeatedly, and finally manifest as errors at a primary output several clock cycles later.

First, consider a simple case when there is only one path from the error site to an output. As we traverse this path gate by gate, the error propagation probability from an on-path input of a gate to its output depends on the type of the gate and the signal probability of other off-path signals. In the example shown in Fig. 3, the error propagation probability to the output of the gate C (AND gate) is the product of the probability of the output of gate A being 1 and the error probability at the PI (1×0.2=0.2). Similarly, EPP at the output of the gate D (OR gate) is calculated as $0.2\times(1\text{-}SP_B) = 0.2\times0.7=0.14$.

In the general case in which reconvergent paths might exist, the propagation probability from the error site to the output of the reconvergent gate depends on not only the type of the gate and the signal probabilities of the off-path signals, but also the polarities of the propagated error on the on-path signals. To address this issue, we need propagation rules for reconvergent gates. First, we define the $P_a(U_i)$, $P_{\bar{a}}(U_i)$, $P_1(U_i)$, and $P_0(U_i)$ as follow:

- $P_a(U_i)$ and $P_{\bar{a}}(U_i)$ are defined as the probability of the output of node $U_i$ being $a$ and $\bar{a}$, respectively, where $\bar{a}$ is inverted of $a$. In other words, $P_a(U_i)$ is the probability that the erroneous value is propagated from the error site to $U_i$ with an even number of inversions, whereas $P_{\bar{a}}(U_i)$ is the similar propagation probability with an odd number of inversions.
- $P_1(U_i)$ and $P_0(U_i)$ are defined as the probability of the output of node $U_i$ being 1 and 0, respectively. In these cases, the error is blocked and not propagated.

Note that $P_a(U_i) + P_{\bar{a}}(U_i) + P_1(U_i) + P_0(U_i) = 1$. Since we have considered the polarity of error effect propagation, this will take care of reconvergent points. The error propagation calculation rules for elementary gates are shown in Fig. 4. To illustrate how to employ the propagation rules for reconvergent paths, consider the example shown in Fig. 5. In this example, the error propagation probability from the output of gate A to PO is calculated. Here, we assume that an SEU with sufficient energy hits the gate A. After computing P(E) = 1($\bar{a}$), P(G) = 0.7($\bar{a}$)+0.3(0), and P(D) = 0.2(a) + 0.8(0)[1], we do the following steps to compute the error propagation probability of the erroneous value to the output.

**$P_0$(H)** = $P_0$(C)×$P_0$(D)×$P_0$(G) = 0.7×0.8×0.3 = 0.168
**$P_a$(H)** = ($P_0$(C) + $P_a$(C)) × ($P_0$(D) + $P_a$(D)) × ($P_0$(G) + $P_a$(G)) − $P_0$(H) = (0.7)×(0.2+0.8)×(0.3)−0.168 = 0.042
**$P_{\bar{a}}$ (H)** = ($P_0$(C) + $P_{\bar{a}}$(C)) × ($P_0$(D) + $P_{\bar{a}}$(D)) × ($P_0$(G) + $P_{\bar{a}}$(G)) − $P_0$(H) = (0.7)×(0.8)×(0.7+0.3)−0.168 = 0.392
**$P_1$(H)** = 1 − (0.168 + 0.042 + 0.392) = 0.398
➔**P(H)** = 0.042(a) + 0.392($\bar{a}$) + 0.168(0) + 0.398(1)

[1] This means: $P_{\bar{a}}$(E)=1, $P_{\bar{a}}$(G)=0.7, $P_0$(G)=0.3, $P_a$(D)=0.2, $P_0$(D)=0.8.

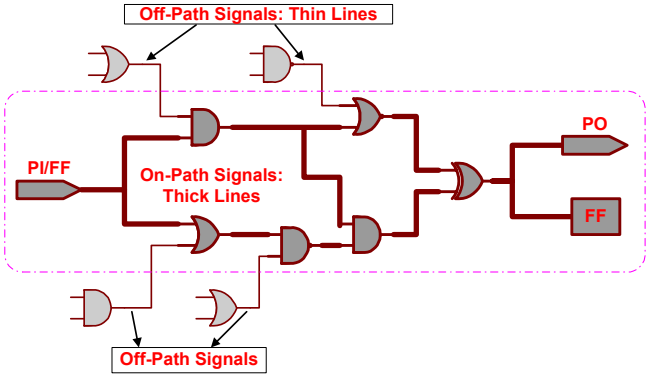


Fig. 2: A typical path between an erroneous input/flip-flop to a primary output/flip-flop

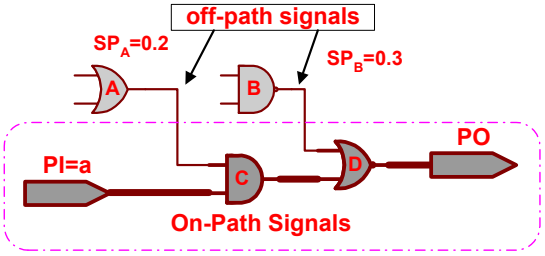


Fig. 3: A simple path between an erroneous input to a primary output

The system failure due to SEUs hitting the gate A can be computed as ($r_A$ is the error rate of gate A):

P(System failure due to gate A) = $[P_a(H)+P_{\bar{a}}(H)]\times r_A$
= (0.042 + 0.392) × $r_A$ = 0.434 × $r_A$

The following algorithm shows how we can extract and then traverse all paths from a given error site to all reachable outputs and how we apply the propagation probability rules as we traverse the paths.

| | |
|---|---|
| **AND** | $P_1(out) = \prod_{i=1}^{n} P_1(X_i)$<br>$P_a(out) = \prod_{i=1}^{n} [P_1(X_i) + P_a(X_i)] - P_1(out)$<br>$P_{\bar{a}}(out) = \prod_{i=1}^{n} [P_1(X_i) + P_{\bar{a}}(X_i)] - P_1(out)$<br>$P_0(out) = 1 - [P_1(out) + P_a(out) + P_{\bar{a}}(out)]$ |
| **OR** | $P_0(out) = \prod_{i=1}^{n} P_0(X_i)$<br>$P_a(out) = \prod_{i=1}^{n} [P_0(X_i) + P_a(X_i)] - P_0(out)$<br>$P_{\bar{a}}(out) = \prod_{i=1}^{n} [P_0(X_i) + P_{\bar{a}}(X_i)] - P_0(out)$<br>$P_1(out) = 1 - [P_0(out) + P_a(out) + P_{\bar{a}}(out)]$ |
| **NOT** | $P_0(out) = P_1(in)$<br>$P_a(out) = P_{\bar{a}}(in)$<br>$P_{\bar{a}}(out) = P_a(in)$<br>$P_1(out) = P_0(in)$ |

Fig. 4: Computing probability at the output of a gate in terms of its inputs

### 2.1. The Main Algorithm

For any flip-flop, $ff_i$:

1. *Path Construction*: Extract all on-path signals (and gates) from $n_i$ to any reachable primary output $PO_j$ and/or flip-flop $ff_j$ . This is achieved using the forward and backward Depth-First Search (DFS) algorithms [9].

2. *Ordering*: Levelize signals on these paths using the *topological sorting* algorithm [9]. Topological sort of a directed acyclic graph is an ordered list of the vertices such that if there is an edge $(u,v)$ in the graph, then $u$ appears before $v$ in the list.

3. *Propagation Probabilities Computation*: Traverse the paths in order and apply propagation rules to compute probability for each on-path node based on propagation probability rules.

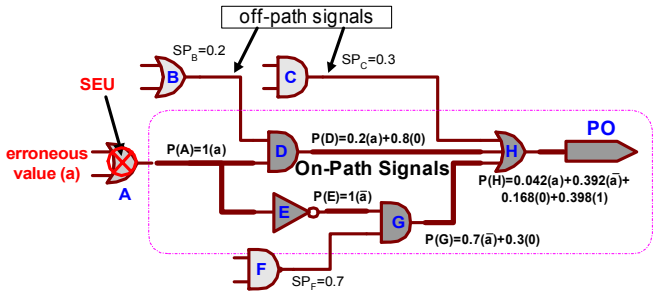


Fig. 5: Applying error propagation rules for a reconverging path

### 2.2. Path Construction Algorithm

1. Construct the directed graph $G(V,E)$ corresponds to the combinational part of the circuit.
2. Perform DFS algorithm to find all reachable nodes of the circuit from the erroneous node, $n_i$. This subset of nodes is denoted by $V1$.
3. The Graph $G(V1,E1)$ is defined as follow: E1 is the list of all edges (u, v) of G(V,E), where both vertices $u$ and $v$ are in V1, i.e., E1 = {(u, v)| | $(u,v) \in$ E, $u,v \in$ V1}
4. Apply *topological sorting* algorithm on graph $G(V1,E1)$. $V1_{sort}$ is the ordered list of all nodes of this graph with respect to the topological sorting. $V1_{sort} = \{U_1, U_2, U_3, \ldots, U_n\}$.

### 2.3. Propagation Computation Rules

We start at node $U_1$, assuming that one of its inputs has an erroneous value $a$, and compute propagation probability of erroneous value at node $U_1$ and then move to the next node $U_2$ in the sorted list. For every node $U_i$, which belongs to $V1_{sort}$, we apply the propagation rules according to Fig. 4.

For every input $X_i$, If $X_i \in V1_{sort}$, we have already computed its error probability using the propagation rules. In this case, $X_i$ is an *on-path* signal. If $X_i \notin V1_{sort}$, i.e. $X_i$ is an *off-path* signal, we use its signal probability value. Since the node $X_i$ is not reachable from erroneous node $n_i$, the values of $P_a(X_i)$ and $P_{\bar{a}}(X_i)$ will be 0 and $P_0(X_i)$ = 1 – $P_1(X_i)$. In a linear time proportional to the number of nodes of $V1_{sort}$, we will reach to the end of list $V1_{sort}$.

Note that the construction of the graph G(V,E) is done in $O(|V|+|E|)$. Also, both DFS and topological sorting algorithms are in the order of $O(|V|+|E|)$. So, the path construction and ordering can be done in $O(|V|+|E|)$. Also, traversing of the paths and applying the propagation rules can be done in $O(|V|+|E|)$. Therefore, the overall complexity of our system failure probability computation algorithm is $O(|V|+|E|)$.

### 2.4. System Failure Probability

After completing the three steps (path construction, ordering, and propagation computation) for every $PO_j$ reachable from $n_i$, we have $P_a(PO_j)$ and $P_{\bar{a}}(PO_j)$ computed. The set of all reachable outputs from node $n_i$ is called $PO_{reachable\ from\ ni}$. If $PO_j$ is a primary output, the error propagation probability to $PO_j$ is equal to $P_{erroneous}(PO_j)=P_a(PO_j)+P_{\bar{a}}(PO_j)$. If $PO_j$ is a flip-flop input, the error propagation probability is equal to $P_{erroneous}(PO_j) \times P_{latched}(n_i,j)$. Note $P_{latched}(n_i,j)$ is the probability that an erroneous value propagated from node $n_i$ is captured in $FF_j$.

Using the above steps, we have already computed the propagation probability of an erroneous value from node $n_i$ to all reachable outputs. Since *system failure* (SF($n_i$)) due to bit-flip at $n_i$ occurs if an erroneous value is propagated to at least one output, SF($n_i$) is calculated as follow:

$$SF(n_i) = R_{SEU}(n_i) \times P_{latched}(n_i) \times P_{sensitized}(n_i)$$

$$SF(n_i) = R_{SEU}(n_i) \times \left(1 - \prod_{j=1}^{k} [1 - P_{erroneous}(PO_j) P_{latched}(n_i, j)]\right)$$

where $k$ is the number of outputs belonging to the $PO_{reachable\_from\_ni}$. Note that $P_{latched}(n_i,j) = 1$ if $PO_j$ is a primary output.

### 2.5. Output Dependency

Although the propagation probabilities ($P_{erroneous}(PO_j)$) are precisely computed for each reachable output (the accuracy is more than 97% for ISCAS'89 benchmark circuits), the computed system failure probability (SF($n_i$)) may not be accurate for all circuit nodes. This happens when there is a dependency between two or more reachable outputs.

Consider an example shown in Fig. 6 (a). In this example, the exact propagation probabilities to $PO_j$ ($PO_j$ = 0.25($\bar{a}$) + 0.75(0)) and $PO_k$ ($PO_k$ = 0.25(a) + 0.75(0)) are calculated. The computed system failure probability using the above equation is SF(A) = $R_{SEU}$(A)×(1−0.75×0.75) = 0.4375×$R_{SEU}$(A). However, the actual SF(A) is 0.25 × $R_{SEU}$(A). In this example, an SEU is propagated to either both or none of the two outputs. So, there is a dependency between $PO_j$ and $PO_k$.

We have shown examples of different types of output dependency in Fig. 6. The output dependencies shown in Fig. 6(a) and (b) can be resolved by forwarding the signal probability of $PO_k$ to the other output ($PO_j$). In other words, instead of EPP of $PO_k$, SP of $PO_k$ is forwarded to the next stages. Note that the exact solution to the provisional probability calculation, such as the example shown in Fig. 6 (c), requires logic implication computation and is computationally intractable.

## 3. EXPERIMENTAL RESULTS

The proposed approach was implemented and applied to ISCAS89 sequential benchmark circuits. All experiments have been performed on the DELL Precision 450 © system equipped with 2 GB main memory. Table I shows the results for our systematic approach as well as fault injection based on random simulation. In the random simulation method, at least 1% of all input combinations must be simulated to achieve a reasonable accuracy [4]. For large circuits, such simulations are intractable. So, at most 100,000 simulation vectors are randomly applied to the circuits. Due to this fact, the speedups reported in Table I are smaller than the actual values for larger circuits.

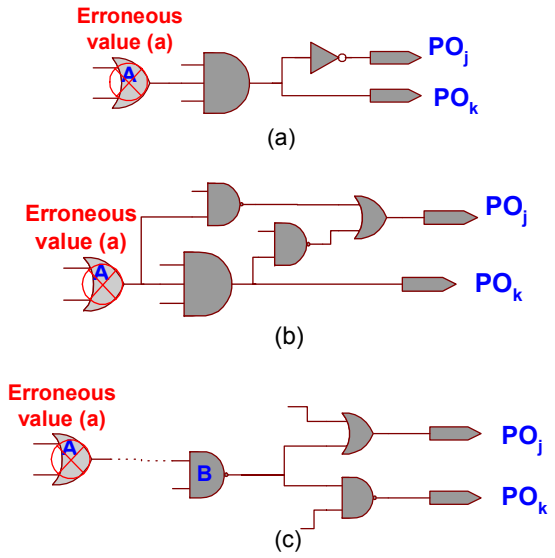

Fig. 6: Examples of output dependency

The run time of our systematic approach varies from less than one second for small circuits to at most 5 minutes for the largest circuits. For larger circuits, we have simulated a limited number of gates of the circuits due to extremely long simulation time of the random-simulation method. Our approach is 4-5 orders of magnitude faster than the random simulation method. As shown in this table, the difference between the results of our approach and random simulation is about 5%, on average.

## 4. CONCLUSIONS

Soft errors due to single event upsets are the main reliability threat of digital systems. In particular, vulnerability of digital systems grows in direct proportion to the Moore's law. In this paper, an accurate propagation probability computation technique has been developed, which significantly reduces the SER estimation time.

The proposed approach leverages the signal probability calculation, which is already used in other steps of the design flow, and computes the error propagation probability. Some efficient graph-based algorithms have been used for this computation. To improve the accuracy of our approach, we have considered the output dependencies. Experiments on benchmark circuits and comparison of the results with the random simulation technique show the effectiveness and the accuracy of the presented approach. Our approach, on average, is about 95% accurate compared to the random-simulation method while 4-5 orders of magnitude faster.

This analysis helps the designers to evaluate the vulnerability of gate-level designs to soft errors and the contribution of each gate to the overall SER. Gates which are contributing most to the overall system vulnerability can be protected by circuit-level radiation hardening techniques.

TABLE I. Comparison of our systematic approach to random simulation

| Circuit | m+n | #Gates | #Gates Sim | Sys | Sim | %Diff | SpeedUp |
|---|---|---|---|---|---|---|---|
| s298 | 17 | 119 | 119 | 0.01 | 1309 | 4.1 | 131230 |
| s344 | 24 | 160 | 160 | 0.04 | 1860 | 3.6 | 46410 |
| s349 | 24 | 161 | 161 | 0.04 | 1903 | 3.7 | 47630 |
| s382 | 24 | 158 | 158 | 0.03 | 2208 | 4.8 | 73490 |
| s386 | 13 | 159 | 159 | 0.04 | 151.82 | 1.7 | 3800 |
| s400 | 24 | 164 | 164 | 0.03 | 2705 | 5.0 | 90260 |
| s420 | 34 | 218 | 218 | 0.02 | 2807 | 3.0 | 141130 |
| s444 | 24 | 181 | 181 | 0.03 | 2952 | 5.0 | 98080 |
| s510 | 25 | 211 | 211 | 0.02 | 2595 | 3.6 | 133720 |
| s526 | 24 | 193 | 193 | 0.03 | 3663 | 3.5 | 135570 |
| s641 | 54 | 379 | 379 | 0.49 | 8914 | 9.0 | 19110 |
| s713 | 54 | 393 | 393 | 0.88 | 14807 | 9.8 | 17640 |
| s820 | 23 | 289 | 289 | 0.04 | 5604 | 3.9 | 142600 |
| s832 | 23 | 287 | 287 | 0.03 | 5578 | 3.9 | 188720 |
| s838 | 66 | 446 | 446 | 0.23 | 20919 | 3.0 | 97520 |
| s953 | 45 | 395 | 395 | 0.15 | 11178 | 4.3 | 79950 |
| s1196 | 32 | 529 | 529 | 0.41 | 28883 | 3.6 | 72800 |
| s1238 | 32 | 508 | 508 | 0.28 | 18784 | 3.4 | 69510 |
| s1423 | 91 | 657 | 657 | 1.63 | 34891 | 3.9 | 23810 |
| s1488 | 14 | 653 | 653 | 0.28 | 4778 | 4.4 | 17220 |
| s1494 | 14 | 647 | 647 | 0.46 | 7045 | 4.4 | 15480 |
| s9234 | 247 | 5597 | 52 | 54.41 | NA | 11.3 | 87230 |
| s15850 | 611 | 9772 | 136 | 352.2 | NA | 12.6 | 28440 |
| s35932 | 1763 | 16065 | 110 | 124.9 | NA | 4.5 | 271240 |
| s38584 | 1464 | 19253 | 77 | 286.6 | NA | 7.1 | 167180 |
| s38417 | 1664 | 22179 | 85 | 292.2 | NA | 6.0 | 170126 |
| **average** | - | - | - | **40.00** | **NA** | **5.1** | **93072** |

**m+n:** Number of flip-flops plus the number of primary inputs
**#Gates Sim**: Number of gates simulated
**Sys:** Run time for our systematic approach (in seconds)
**Sim:** Run time for random simulation method (in seconds)
**%Diff:** Percentage of difference between these two methods
**SpeedUp:** Speed up of our method over random simulation
**NA**: Not Available because of extremely large simulation time

## REFERENCES


[1] H. T. Nguyen and Y. Yagil, "A systematic approach to SER estimation and solutions," Proc. of the 41st Annual Intl. Reliability Physical Symp., pp. 60-70, Dallas, Texas, 2003.

[2] P. Shivakumar, M. Kistler, S. W. Keckler, D. Burger, L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic" Proc. of the Intl. Conf. on Dependable Systems and Networks (DSN), Washington, D.C., June 2002.

[3] S. S. Mukherjee, C. Weaver, J. Emer, S. K. Reinhardt, and T. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor," Proc. of the Intl. Symp. on Micro-architecture, San Diego, CA, pp. 29-40, Dec. 2003.

[4] A. Maheshwari, I. Koren, and W. Burleson, "Techniques for Transient Fault Sensitivity Analysis and Reduction in VLSI Circuits," Proc. of the IEEE Intl. Symp. on Defect and Fault-tolerance, pp. 597-604, 2003.

[5] K. Mohanram and N. A. Touba, "Cost-Effective Approach for Reducing Soft Error Failure Rate in Logic Circuits," Proc. of the International Test Conference (ITC), pp. 893-901, 2003.

[6] K. P. Parker and E. J. McCluskey, "Probabilistic treatment of general combinational networks," IEEE Trans. on Computers, Vol. c-24, No.6, pp. 668-670, June 1975.

[7] J. Savir, G. S. Ditlow, P. H. Bardell, "Random pattern testability," IEEE Trans. on Computers, Vol. c-33, No. 1, pp. 79-90, Jan. 1984.

[8] R. Kodavarti and D. E. Ross, "Signal probability calculation using partial functional manipulation," Proc. of the VLSI Test Sym., [Digest of Papers], 1993.

[9] T. H. Cormen, C. L. Leiserson, R. L. Rivest, C. Stein, "Introduction to algorithms," 2nd Edition, MIT Press & McGraw-Hill, pp. 549-551, 2001.