Soft Error Modeling and Remediation Techniques in ASIC Designs

Hossein Asadi
Department of Computer Engineering
Sharif University of Technology, Tehran, Iran
asadi@sharif.edu

Mehdi B. Tahoori
Chair of Dependable Nano Computing
Karlsruhe Institute of Technology, Germany
mehdi.tahoori@kit.edu

Abstract

Soft errors due to cosmic radiations are the main reliability threat during lifetime operation of digital systems. Fast and accurate estimation of soft error rate *(SER) is essential in obtaining the reliability parameters of a digital system in order to balance reliability, performance, and cost of the system. Previous techniques for SER estimation are mainly based on fault injection and random simulations. In this paper, we present an analytical SER modeling technique for ASIC designs that can significantly reduce SER estimation time while achieving very high accuracy. This technique can be used for both combinational and sequential circuits. We also present an approach to obtain uncertainty bounds on estimated* error propagation probability *(EPP) values used in our SER modeling framework. Comparison of this method with the Monte-Carlo fault injection and simulation approach confirms the accuracy and speed-up of the presented technique for both the computed EPP values and uncertainty bounds.*

Based on our SER estimation framework, we also present efficient soft error hardening techniques based on selective gate resizing to maximize soft error suppression for the entire logic-level design while minimizing area and delay penalties. Experimental results confirm that these techniques are able to significantly reduce soft error rate with modest area and delay overhead.

1 Introduction

Device scaling in *Complementary Metal-Oxide-Semiconductor* (CMOS) technology has made digital circuits extremely sensitive to soft errors. These errors are radiation-induced transient errors caused by neutrons from cosmic rays and alpha particles from packaging material [1]. In the past, soft errors were regarded as a major concern only for microelectronic devices used in space applications as well as *Dynamic Random Access Memory* (DRAM) and *Static Random Access Memory* (SRAM) used at the ground-level [2]. Recent studies, however, show that designs manufactured at advanced technology nodes, such as 65 nm and smaller, system-level soft errors are much more frequent than in the previous generations [3, 4]. Several big electronic companies such as Intel, IBM, Fujitsu, and Texas Instruments (TI) have dedicated a lot of investments and research study to investigate and mitigate the effect of soft errors on their products [5, 4]. The vulnerability of VLSI circuits to soft errors exponentially increases as an unwanted side effect of Moore's law [6].

When an energetic particle strikes a CMOS transistor, it induces a localized ionization capable to reverse (flip) the data state of a memory cell, logic gate, latch, or *flip-flop* causing a soft error [3]. These errors are called *soft* since the circuit itself is not permanently damaged by the radiation. If the system is reset and rerun, the hardware will perform correctly. High-energy particles that strike a sensitive region in a semiconductor device deposit a dense track of electron-hole pairs as they pass through a p-n junction. Some of the deposited charge will recombine to form a very short duration current pulse at the transistor that was struck by the particle. This can flip the value stored in the memory cell or the logic gate, resulting in a soft error. A charged particle striking an MOS transistor inducing trail of ionization and current pulse. The smallest charge that results in a soft error is called the *critical charge* [7]. Particles that generate less charge than critical charge are considered harmless. Soft errors are caused by external events such as particle hits on a transistor's diffusion area. In the past two decades, researchers have discovered three major radiation mechanisms that cause soft errors in semiconductor devices at terrestrial altitudes. These are a) alpha particles, b) high-energy neutrons, and c) low-energy neutrons interacted with the isotope boron-10 (^{10}B) [8].

Accurate estimation of soft error rate (SER), i.e., the probability of system failure due to soft errors, is a key factor in the design of cost-effective soft error resilient ASICs. To compute the SER of a circuit, it is required to compute the probability that the node is functionally sensitized by the input vectors to propagate the erroneous value from the error site to outputs (logical masking) [9]. Previous methods on SER estimation are based on *fault injection* (FI) using random vector or fault simulation approaches [10, 11, 9, 1, 12, 13, 14, 15, 16]. However, SER estimation of large circuits using fault injection becomes intractable since the number of required simulation steps grows exponentially with the size of the circuit. In contrast to simulation-based methods, our proposed *analytical* method estimates SER values much faster than previous approaches by reusing signal probabilities (SP) and topological traversal of the design netlist [17, 18].

Signal probability estimation methods have widely been used for power estimation and testability analysis [19]. The accuracy of the analytical SER estimation method is dependent on the accuracy of SP values. Depending on the size of the circuit and run-time constraints of SP estimation method, the accuracy of SP values can vary. Although it is possible to measure the accuracy of the analytical method with comparison versus exhaustive or simulation-based methods for small circuits, it is impossible to perform such analysis for large (industrial-size) circuits. Exhaustive method is intractable even for medium-sized blocks (more than a few hundred gates), and the accuracy of simulation-based approaches are questionable for larger circuits. Therefore, it is extremely important to come up with an approach to obtain the accuracy of SER values along with the SER values.

This work complements our previous work by obtaining the accuracy (uncertainty bound) of estimated error propagation probabilities [17, 18, 20]. In this paper, we first present an analytical soft error modeling technique for ASIC designs. The proposed approach accurately computes the contribution of each gate and path to the overall SER. In contrast to fault injection techniques, the proposed approach is analytical and has minimal dependency on vector simulations. We also experimentally examine the sensitivity of the accuracy of obtained EPP values to the accuracy of SP values. This analysis is important because obtaining SP values with more accuracy requires exponentially more run time. We also present a mixed analytical and fault simulation method to improve the accuracy of estimated SER values for very deep combinational paths in the circuit.

The next step after SER estimation of an ASIC design is soft error remediation. Both experiments and analytical models show that the major components used in ASIC designs such as latches, flip-flops, and combinational logic are now sensitive to cosmic rays at terrestrial levels [3, 4]. In this paper, we present efficient gate sizing techniques for SER reduction of the combinational logic core of ASIC designs in which the contribution of each logic gate to the system-level SER and its criticality in the overall performance are carefully considered. Fast and efficient soft error vulnerability minimization algorithms under different constraints, such as area, delay, or both, are also presented. Having a fast and accurate SER estimation technique along with a cost-effective soft-error remediation technique would help ASIC designers to deliver a soft-error resilient product without compromising other important parameters such as time-to-market, cost, and performance.

The rest of this paper is organized as follows. In Sec. 2, previous SER modeling and remediation techniques are reviewed. In Sec. 3, the analytical approach for SER estimation based on signal probability is presented. Computation of uncertainty bounds (variances) for the estimated error propagation probability values is discussed in Sec. 4. In Sec. 5, a soft error remediation technique for combinational logic is presented. Finally, Sec. 6 concludes the paper.

2 Related Work

2.1 Soft Error Modeling and SER Estimation

2.1.1 Single Event Upset Modeling Techniques

In CMOS circuits, radiation-induced errors can corrupt both sequential elements and combinational logic. Those that directly change the state of flip-flops and memory elements are often called *Single Event Upset* (SEU). Those errors that create a transient glitch in the combinational logic are called *Single Event Transient* (SET). In the rest of this paper, we refer to both SETs and SEUs as *Single Event Effects* (SEEs). SETs are modeled by injecting a current pulse at the output of a gate. This pulse has rapid rise time and gradual fall time. The current of injected pulse can typically be approximated using a double exponential current (shown in equation 1) [21, 22]. In this equation, Q is the amount of injected charge (positive or negative) that is deposited as a result of a particle strike. τ_α represents the collection time-constant of the junction, and τ_β accounts for the ion-track establishment time-constant. τ_α and τ_β are constants dependent on the CMOS technology process-related factors.

$$I_{SET}(t) = \frac{Q}{\tau_\alpha - \tau_\beta}\left(e^{(\frac{-t}{\tau_\alpha})} - e^{(\frac{-t}{\tau_\beta})}\right) \tag{1}$$

A numerical analysis technique to simulate the SEE effect in logic circuits is presented in [23]. The transient voltage response is computed by solving the non-linear Riccati differential equation using the Runge-Kutta method. The results of the proposed modeling are within 10% of the results observed using SPICE simulations.

A delay soft error modeling is presented in [24]. The delay soft error is a temporary delay in CMOS combinational circuits due to high energetic particle strike. The effects of the delay soft error in CMOS combinational circuits have been investigated in [24]. It has been demonstrated that as technology and operating voltages scale down, delay soft errors can increase the circuit SER.

Techniques to compute the raw FIT rates of different elements in combinational and sequential circuits including domino gates, latches, and static gates are presented in [25, 1]. It is concluded that there are five main factors affecting the raw FIT rates in CMOS circuits: a) diffusion area, b) capacitance charge, c) operating voltages, d) fabrication technology, and e) particle flux.

An approach for soft error analysis of combinational logic is presented in [26]. This approach can be applied to library cell-based designs. In the proposed approach, analytical equations are used to model the propagation of a transient pulse to the inputs of system bistables. This approach simplifies SPICE equations to expedite simulations. However, this approach has still a large run-time for medium-size circuits.

There are also several studies [27, 28] that measure flux of neutron and alpha particles at the ground level. These studies investigate impact of energetic particles on 90nm, 65nm, and 45nm technologies and present experimental data on characteristics of SETs and SEUs on CMOS technology. Discussion and investigation of these techniques is beyond the scope of this paper.

2.1.2 Gate-Level SER Techniques

A method to statically analyze the susceptibility of arbitrary combinational circuits to single event upsets is presented in [29, 30]. Accurate models are based on pre-characterization methods. Logical masking is computed using *Binary Decision Diagrams* (BDDs) with circuit partitioning. The proposed approach is a static SER analysis methodology since it relies on implicit enumeration of the input vector space. The transient pulses are encoded and propagated at the gate-level using BDDs. It is well known that the worst-case complexity of BDD encoding of logic functions is exponential to the number of variables [31]. To make fast manipulation of BDDs, a partitioning heuristic is exploited. Using the partitioning heuristic, this approach runs faster than fault injection techniques. However, the accuracy of estimated SER values becomes less than that of FI techniques.

A transient error-sensitivity estimation method based on random simulations is presented in [11]. In a circuit under study, 1% of total possible random vector simulations is performed to compute circuit SER. However, the number of experiments increases exponentially with the circuit size and the number of circuit inputs.

A method to compute the latching-window masking factor (also, called *timing vulnerability factor* (TVF)) of sequential elements is presented in [32]. Using SPICE simulations, it is concluded that the TVF varies between 50% down to almost 0%. TVF is computed based on the following parameters: a) the propagation delay though the combinational logic and the intrinsic delay within the sequential logic, b) the setup time, c) the clock rise and fall times and the clock jitter, d) the clock skew, and e) the clock cycle width.

A soft error rate analysis methodology for combinational and memory circuits is presented in [15]. An approach to compute the critical charge for logic and sequential elements is also presented to expedite SPICE simulations. In this method, however, logical masking factor is computed by random vector simulations which can be time-consuming for large circuits.

An approach for soft error analysis of combinational and sequential logic is presented in [33]. In the proposed approach, electrical masking, logical masking and latching-window masking are considered in order to compute the system SER. The logical masking parameter is computed by random vector simulations.

A general computational framework based on probabilistic transfer matrices (PTMs) to estimate the effects of soft errors on logic circuits is developed in [34]. Algebraic decision diagrams are used to implement and optimize the PTMs. Since the size of decision diagrams grows exponentially with the circuit size, this approach is not applicable for large circuits.

A framework to compute SER of a circuit at the gate-level is presented in [35]. The SER is calculated based on sum of the FIT rates of the underlying logic gates characterized for different energy levels. The proposed method does not provide accurate SER for interconnected circuits with reconvergent fanouts. A similar approach is also presented in [36] which uses fault injection to compute overall SER.

As stated in the introduction section, there are also numerous SER estimation techniques which work based on fault injection and Monte-Carlo fault simulation [10, 11, 9, 1, 12, 13, 14, 15, 16]. SER estimation of large circuits using these techniques is intractable since the number of simulation runs grows exponentially with the size of the circuit. However, using emulation-based approaches one can significantly reduce simulation time [37].

2.2 Soft Error Remediation Techniques in ASICs

2.2.1 Gate-Level Hardening Techniques

A soft error mitigation technique is presented in [9, 38]. The asymmetric soft error susceptibility of internal nodes in combinational logic is exploited to increase the reliability of the logic circuit. To achieve a cost-effective tradeoff between overhead and SER reduction, the gates with highest soft error susceptibility are targeted first. The logic gates are protected against SETs by a partial Triple Module Redundancy (TMR) scheme. The experimental results show that using the partial TMR scheme, the circuit SER can be reduced up to 88% with 50% area overhead.

A similar approach has been presented in [16]. Using the asymmetric soft error susceptibility of internal logic gates, the most susceptible gates are extracted and hardened using a transistor sizing method presented in [23]. The proposed algorithm uses a fault simulation-based technique to identify and rank the critical nodes that contribute significantly to the soft error failure rate of a combinational logic block. Then, these critical gates are sized in order to be hardened against SETs. The results show that the soft error rate of experimented circuits has been reduced by 90% with average area/power overhead of 18-23%.

A methodology for the synthesis of low-cost *Concurrent Error Detection* (CED) circuitry based on parity prediction for logic circuits is introduced in [39]. The basic idea is to construct a simple Boolean function of a selected subset of the inputs of the circuit and to disable CED in *don't-care* conditions. The proposed method can detect, on average, 68% of soft error occurrence on the circuit. This comes at the cost of 102% area overhead.

A time-redundancy technique is presented in [40], which exploits the inherent temporal redundancy (timing slack) of logic signals to increase soft error robustness. The vulnerable paths are identified and delay elements are inserted along these paths. Also, the CMOS flip-flops are structurally modified so that they can sample and latch signal value at different time instances within a clock cycle. The slave stage within the flip-flop contains a majority voter to vote among the different sampled values. Using the proposed approach, SER can be reduced by 70% with 12% area overhead. There is also an small performance degradation in flip-flops for the sampling process.

A built-in soft error resilience technique for detection and correction of soft errors in latches and flip-flops is presented in [41, 42]. In the proposed technique, on-chip design for testability and debug resources are reused to reduce the area overhead redundancy. These resources are generally idle during normal operation of the circuit and they are only used for product testing and also post-silicon debug activities. *Scanout* structures along with circuit-level voting elements are used for soft error protection. This approach can reduce the SER of flip-flops and latches by 20 times with less than 5% power overhead and 0.5% area overhead.

A technique for correcting soft errors in combinational logic is presented in [43]. This technique is based on logic duplication. A circuit-level voting element is used to detect any mismatch at the bistable outputs. Simulation results show that the combinational logic SER is reduced by more than an order of magnitude. Using this technique, soft errors affecting sequential elements are also automatically corrected.

Lastly, a time-redundancy mitigation technique to remediate effect of SETs on combinational logic has been discussed in [44, 45]. In the proposed technique, circuit bistables are duplicated. In each clock cycle, the duplicated bistables are used to latch the outputs of the combinational logic with a delay greater than the width of the largest possible SET. If contents of the original bistable and the duplicated bistable do not agree, it means that an SET has been occurred within the combinational logic and been propagated to the circuit bistables.

2.2.2 Circuit-Level Hardening Techniques

The objective of circuit-level hardening techniques is to protect the device against soft errors by reducing the vulnerability of the CMOS transistor to radiation events. An effective circuit-level hardening technique is transistor sizing [23]. The sizing factor directly depends on the charge of particles and technology process. A large transistor can dissipate (sink) the injected charge as quickly as it is deposited, so that the transient does not achieve sufficient magnitude and duration to propagate to gates in the fanout. However, this technique incurs significant overhead in terms of area, power and to some extent delay. The effect of transistor sizing on the soft error rate of CMOS logic gates has been investigated in [23]. The proposed technique calculates the minimum transistor size required to make a CMOS gate immune to SETs.

A gate sizing algorithm that trades off SER reduction and area overhead is presented in [46]. The gate sizing algorithm is applied to logic gates that are the largest contributors to the circuit SER. To further reduce the logic SER, an enhanced library of flip-flop variants is used to trade off reduced SET latching susceptibility with larger amounts of delay overhead. In this approach, the circuit slack information is used to select proper flip-flops from the library.

The effect of threshold voltage (V_t) on the soft error rate of both logic gates and flip-flops has been investigated in [47]. It is shown that increasing threshold voltage improves the SER of transmission-gate based flip-flops (TGFF). However, increasing threshold voltage can adversely affect the robustness of combinational logic due to the effect of higher threshold voltages on the attenuation of transient pulses. It has also been shown that higher V_t can improve the robustness of 6-transistor SRAMs. This technique, however, needs another mask process for the implementation of two threshold voltages. Also, high V_t can slow down the device.

A low power soft error suppression technique to reduce the impact of soft errors in dynamic logic is presented in [48]. In this approach, a complementary pass transistor (c-pass) logic and an additional weak keeper transistor at

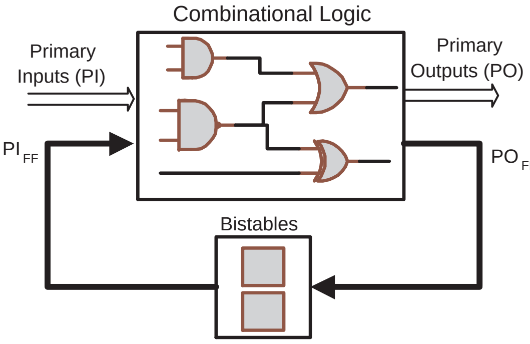

Figure 1. A typical block diagram of synchronous sequential circuits

the output of gates are used to selectively isolate the logic gates struck by SEEs. It is shown that the magnitude of a transient on the next stage of the combinational circuit can be substantially minimized when c-pass transistors are used to shield the two circuit stages. The results show that this technique achieves soft error suppression with no extra power consumption and modest area (2.6%) and delay (13.6%) overhead.

3 SER Modeling in Combination Logic

A typical synchronous circuit consists of combinational logic and bistables. A bistable is conventionally referred to as a flip-flop (FF) or a latch. Figure 1 shows the typical representation of a synchronous sequential circuit. *Primary Inputs (PIs)* and the outputs of bistables are inputs of the combinational logic (CL). Also, *Primary Outputs* (POs) and the inputs of the bistables are outputs of CL. In the remainder of this paper, the term "outputs" refers to both primary outputs and bistable inputs (POs/FFs). This section presents SER modeling in the combinational logic.

To compute the error rate of a node in a circuit, three probability factors are required to be computed: electrical masking (also, called *nominal FIT* or raw FIT, denoted by $P_{SEE}(n_i)$), logical masking (also, called *logic derating*, denoted by $P_{sensitized}(n_i)$), and latching-window masking (also, called *timing derating*, denoted by $P_{latched}(n_i)$). The soft error rate of node n_i can be computed as [38]:

$$SER(n_i) = P_{SEE}(n_i) \times P_{sensitized}(n_i) \times P_{latched}(n_i) \tag{2}$$

$P_{SEE}(n_i)$ can be easily obtained from layout information of library cells, technology parameters, and particle energy [11, 9, 25]. Latching-window masking probability is generally computed based on the logic derating and timing derating factors. The logic derating is the probability that an erroneous value at the output of a logic gate is propagated to a bistable input. The estimation of logic derating probability is included in $P_{sensitized}(n_i)$. The timing derating probability depends on the width of a glitch caused by a particle strike at the output of the gate, the latching windows of reachable bistables, and the propagation delay from the output of the struck gate to the inputs of reachable bistables. Here, we only concentrate on computing the logic derating, which is the most time-consuming part of SER modeling. In the proposed technique to model logic derating, all circuit nodes are considered as potential error sites. Circuit nodes are referred to as logic gates and bistables.

3.1 The Proposed SER Modeling Technique

In this section we present a framework to accurately estimate SER in gate-level digital circuits and obtain uncertainty bounds. In the proposed approach, the structural paths from the *error sites* to all reachable outputs and bistables are extracted first. Since SEEs can affect the active area of all transistors, (the outputs of) all logic gates are considered as potential error sites. Then, these structural paths are traversed to compute the propagation probability of the erroneous value to the reachable primary outputs or the reachable *bistables* (flip-flops or latches). Figure 2 shows an example of the paths from an erroneous node to primary-outputs/bistables. An *on-path* signal is defined as a net on a path from the error site to a reachable output. Also, an *on-path gate* is 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 shown in Figure 2.

In order to explain the main idea, 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 probability that the error would propagate from an on-path input of a gate to its output depends on the type of the gate and the *Signal Probability* (SP) of other off-path signals. The SP of a line l indicates the probability of l having logic value "1" [19]. SP estimation techniques have been presented in [49, 50, 51].

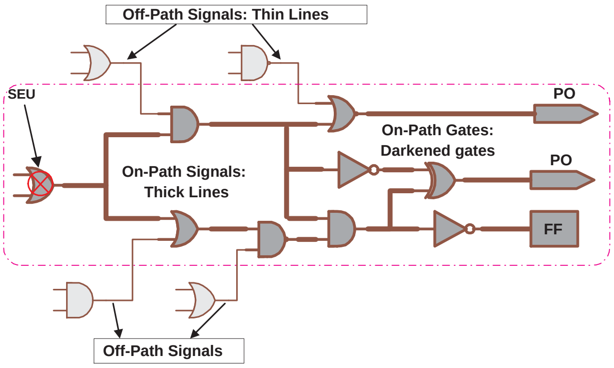

Figure 2. Paths from an error site to reachable outputs

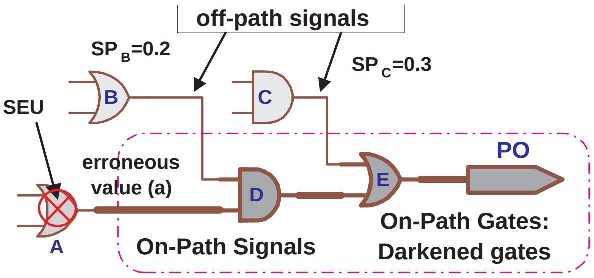

Figure 3. A simple path between an erroneous gate to a primary output

Since SP calculation is typically done in other steps of design, such as power and heat gradient estimation, we can reuse the SP values and reduce the overall complexity of the presented method.

As an example, consider a simple path shown in Figure 3. The probability that the erroneous value a appears at the output of the gate D (AND Gate) is the product of the probabilities of the output of gate B being 1 and the output of gate A being erroneous. So, the probability that the erroneous value appears at the output of gate D is calculated as $P(D_{erroneous}|A\ is\ erroneous) = 1 \times 0.2 = 0.2$. In the same manner, the probability of erroneous value at the output of the gate E (OR Gate) is $0.2 \times (1 - SP_C) = 0.2 \times 0.7 = 0.14$. Suppose that $P_{SEE}(A)$ is the nominal error rate of gate A[1]. In this case, the system failure due to an SEE hitting the gate A equals to : $P(System\ failure\ due\ to\ gate\ A) = 0.14 \times P_{SEE}(A)$

Now consider the general case in which reconvergent paths might exist. In this case, the propagation probability from the error site to the output of the reconvergent gate depends not only on the type of the gate and the signal probabilities of the off-path signals, but also on the polarities of the propagated erroneous values on the on-path signals. In the presence of errors, the status of each signal can be expressed with four values:

- *0:* no error is propagated to this signal line and the signal has an error-free value of 0.
- *1:* no error is propagated to this signal line and it has the logic value of 1.
- *a:* the signal has an erroneous value with the same polarity as the original erroneous value at the error site (denoted by a).
- *$\bar{a}$:* the signal has an erroneous value, but the erroneous value has an opposite polarity compared to the erroneous value at the error site (denoted by $\bar{a}$).

Based on this four-value logic, we can define error propagation rules for each logic gate. These probabilities, denoted by $P_a(U_i)$, $P_{\bar{a}}(Ui)$, $P_1(U_i)$, and $P_0(U_i)$, are defined as follows:

- $P_a(U_i)$ and $P_{\bar{a}}(Ui)$ are defined as the probabilities of the output of node U_i being a and $\bar{a}$, respectively. 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 $P_{\bar{a}}(Ui)$ is the similar propagation probability but with an odd number of inversions.
- $P_1(U_i)$ and $P_0(U_i)$ are the probabilities of the output of node U_i being 1 and 0, respectively. In these cases, the error is masked and not propagated.

[1] We assume that the SEE has sufficient energy to reach PO.

GATE	RULE
AND	$P_1(out) = \prod_{i=1}^{n} P_1(X_i)$
	$P_a(out) = \prod_{i=1}^{n} [P_1(X_i) + P_a(X_i)] - P_1(out)$
	$P_{\bar{a}}(out) = \prod_{i=1}^{n} [P_1(X_i) + P_{\bar{a}}(X_i)] - P_1(out)$
	$P_0(out) = 1 - [P_1(out) + P_a(out) + P_{\bar{a}}(out)]$
OR	$P_0(out) = \prod_{i=1}^{n} P_0(X_i)$
	$P_a(out) = \prod_{i=1}^{n} [P_0(X_i) + P_a(X_i)] - P_0(out)$
	$P_{\bar{a}}(out) = \prod_{i=1}^{n} [P_0(X_i) + P_{\bar{a}}(X_i)] - P_0(out)$
	$P_1(out) = 1 - [P_0(out) + P_a(out) + P_{\bar{a}}(out)]$
3-State BUF	$P_1(out) = P_1(input) \times P_1(enable)$
	$P_a(out) = [P_1(input) + P_a(input)] \times [P_1(enable) + P_a(enable)] - P_1(out)$
	$P_{\bar{a}}(out) = [P_1(input) + P_{\bar{a}}(input)] \times [P_1(enable) + P_{\bar{a}}(enable)] - P_1(out)$
	$P_0(out) = 1 - [P_1(out) + P_a(out) + P_{\bar{a}}(out)]$
NOT	$P_1(out) = P_0(input)$, $P_0(out) = P_1(input)$
	$P_a(out) = P_{\bar{a}}(input)$, $P_{\bar{a}}(out) = P_a(input)$

Table 1. Computing error propagation probability at the output of a gate in terms of its inputs

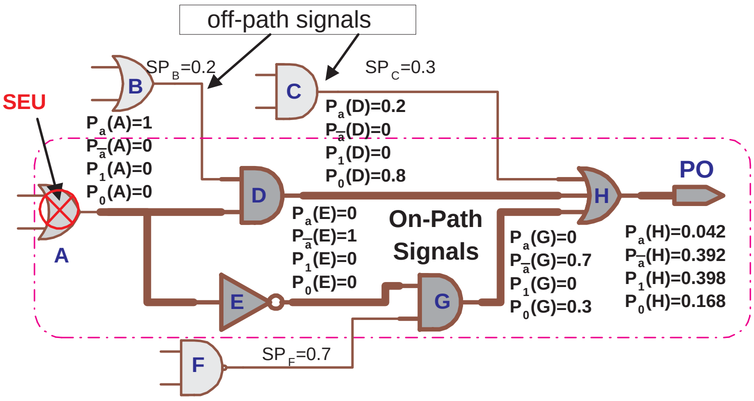

Figure 4. An example: error propagation on reconvergent paths

Note that for on-path signals, $P_a(U_i)+P_{\bar{a}}(Ui)+P_1(U_i)+P_0(U_i) = 1$ and for off-path signals, $P_1(U_i)+P_0(U_i) = 1$. As we traverse the on-path gates, we use signal probability for off-path signals and use the error propagation probability rules for on-path signals. Since the polarities of propagated errors are considered, propagation probabilities at the output of reconvergent gates are correctly calculated. The propagation computation rules for elementary gates and commonly used library gates (AND, OR, NOT, and 3-state buffer) are shown in Table 1.

To illustrate how to employ the propagation rules for reconvergent paths, consider the example shown in Figure 4. Assume that an SEE with sufficient energy hits the gate A. After computing $P(E) = 1(\bar{a})$, $P(G) = 0.7(\bar{a}) + 0.3(0)$, and $P(D) = 0.2(a) + 0.8(0)$ [2], the following steps are performed to compute the error propagation probability of the erroneous value to the output.

$\mathbf{P_0(H)} = P_0(C) \times P_0(D) \times P_0(G) = 0.7 \times 0.8 \times 0.3 = 0.168$
$\mathbf{P_a(H)} = (P_0(C) + P_a(C)) \times (P_0(D) + P_a(D)) \times (P_0(G) + P_a(G)) - P_0(H) =$
$(0.7) \times (0.2 + 0.8) \times (0.3) - 0.168 = 0.042$
$\mathbf{P_{\bar{a}}(H)} = (P_0(C) + P_{\bar{a}}(C)) \times (P_0(D) + P_{\bar{a}}(D)) \times (P_0(G) + P_{\bar{a}}(G)) - P_0(H) =$
$(0.7) \times (0.8) \times (0.7 + 0.3) - 0.168 = 0.392$
$\mathbf{P_1(H)} = 1 - (0.168 + 0.042 + 0.392) = 0.398$
$\rightarrow \mathbf{P(H)} = 0.042(a) + 0.392(\bar{a}) + 0.168(0) + 0.398(1)$

In this case, SER due to an SEE hitting the gate A can be computed as ($P_{SEE}(A)$ is the error rate of gate A):

$P(erroneous\ output|SEE\ at\ gate\ A) = [P_a(H) + P_{\bar{a}}(H)] \times P_{SEE}(A) = (0.042 + 0.392) \times P_{SEE}(A) = 0.434 \times P_{SEE}(A)$

[2] This means: $P_a(D) = 0.2$, and $P_0(D) = 0.8$

3.2 Combinational Logic SER Modeling Algorithm

The steps described above in Sec. 3.1 can be formalized by an algorithm. The following algorithm shows the required steps to compute the overall SER, i.e. it describes how all paths can be extracted and then traversed from a given error site to all reachable outputs and how the propagation probability rules are applied.

For every node, n_i, do:

1. *Path Construction*: Extract all on-path signals (and gates) from n_i to every reachable primary output PO_j and/or bistable FF_k . This is achieved using the forward *Depth-First Search* (DFS) algorithm [52].
2. *Ordering*: Prioritize signals on these paths based on their distance level using the *topological sorting* algorithm [52]. 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 the topological order and apply propagation rules to compute the probability for each on-path node based on propagation probability rules (Table 1).

3.2.1 Path Construction Algorithm

Inputs: Netlist (combinational logic core), error site (n_i)
Output: Topologically sorted list of reachable nodes from n_i

1. Construct the directed graph $G(V, E)$ corresponding to the combinational part of the circuit.
2. Perform the 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. Construct the graph $G(V1, E1)$ as defined as follows: E1 is the list of all edges (u, v) of $G(V, E)$, where both vertices u and v are in V1, i.e., $E1 = \{(u, v)|u, v \in V1\}$
4. Apply the 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. $V1_{sort} = \{U_1,\ U_2,\ U_3, ...,\ U_n\}$.

3.2.2 Error Propagation Computation

Input: Topologically sorted list of reachable nodes from error site, $V1_{sort} = \{U_1,\ U_2,\ U_3, ...,\ U_n\}$
Outputs: Propagation probabilities for reachable outputs

1. Start at node U_1, assuming that one of its inputs has an erroneous value a.
2. Traverse all nodes $U_i \in V1_{sort}$ from U_1 to U_n.
3. For every input signal (X_j) of node U_i, do:
   - If $X_j \in V1_{sort}$, X_j is an on-path signal and $P_1(X_j)$, $P_0(X_j)$, $P_a(X_j)$, and $P_{\bar{a}}(X_j)$ have already computed.
   - If $X_j \notin V1_{sort}$, X_j is an off-path signal, i.e., X_j is not reachable from the erroneous node n_i. In this case, $P_a(X_j) = 0$, $P_{\bar{a}}(X_j) = 0$, $P_1(X_j) = SP_{X_j}$ and $P_0(X_j) = 1 - SP_{X_j}$.
4. Compute $P_1(U_i)$, $P_0(U_i)$, $P_a(U_i)$, and $P_{\bar{a}}(U_i)$ according to Table 1.

$V1_{sort}$ is traversed and error propagation probabilities are computed in a linear time proportional to the number of nodes in $V1_{sort}$. Note that the construction of the graph $G(V, E)$ is done in $O(|V| + |E|)$. Also, both DFS and topological sorting algorithms are $O(|V| + |E|)$. So, the path construction and the 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 this failure probability computation algorithm is $O(|V| + |E|)$.

3.2.3 Overall EPP Estimation and System Failure Probability

After completing the three steps (path construction, ordering, and error propagation computation), $P_a(O_j)$ and $P_{\bar{a}}(O_j)$ are computed for every output O_j reachable from n_i. The set of all reachable outputs from node n_i is called $RO(n_i)$. If O_j is a primary output, the *Error Propagation Probability* (EPP) from node n_i to O_j is calculated as $EPP_{n_i \to O_j} = P_a(O_j) + P_{\bar{a}}(O_j)$. If O_j is a bistable input, then $EPP_{n_i \to O_j} = [P_a(O_j) + P_{\bar{a}}(O_j)] \times P_{latched}(n_i, j)$,

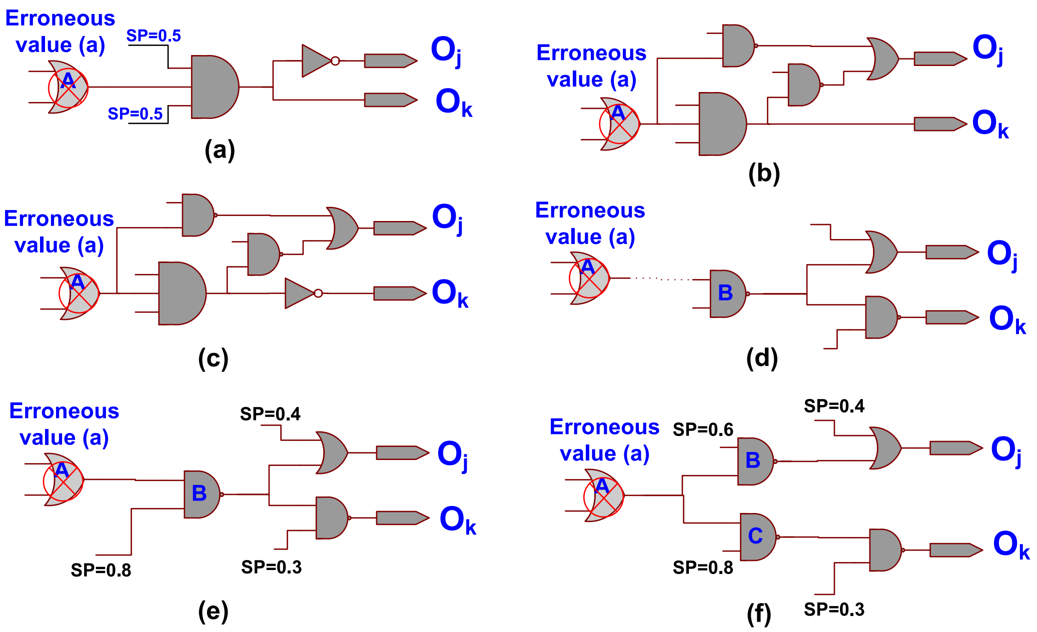

Figure 5. Examples of output dependency

where $P_{latched}(n_i, j)$ is the probability that an erroneous value of node n_i reaching the input of FF_j is captured in FF_j.

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

$$EPP_{n_i} = 1 - \prod_{j=1}^{k} (1 - EPP_{n_i \to O_j}) \tag{3}$$

$$SF(n_i) = P_{SEE}(n_i) \times \left(1 - \prod_{j=1}^{k} 1 - EPP_{n_i \to O_j}\right) \tag{4}$$

, where k is the number of outputs belonging to the $RO(n_i)$.

3.3 Resolving Output Dependencies

Although the error propagation probabilities ($EPP_{n_i \to O_j}$) are precisely computed for each reachable output (the accuracy is more than 97% for ISCAS'89 benchmark circuits, as shown in the next section), 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 Figure 5.a. In this example, the exact propagation probabilities to O_j ($EPP_{A \to O_j} = P_a(O_j) + P_{\bar{a}}(O_j) = 0.25$) and O_k ($EPP_{A \to O_k} = P_a(O_k) + P_{\bar{a}}(O_k) = 0.25$) are calculated. The computed system failure probability using the above equation is $SF(A) = P_{SEE}(A) \times (1 - 0.75 \times 0.75) = 0.4375 \times P_{SEE}(A)$. However, the actual $SF(A)$ equals to $0.25 \times P_{SEE}(A)$. In this example, an SEE is propagated to both or none of the two outputs. So, there is a dependency between O_j and O_k. Examples of different types of output dependency are shown in Figure 5.

The output dependencies shown in Figure 5.a and Figure 5.b can be resolved by forwarding the signal probability of O_k to the other output (O_j). In other words, instead of EPP of O_k, SP of O_k is forwarded to the next stages. A more complicated output dependency has been shown in Figure 5.c. In this example, the output dependency can be resolved by forwarding the SP of the input of the NOT gate to the other output (O_j). The output dependencies shown in these three figures have been resolved in our implementation, accordingly.

Note that the exact solution to the provisional probability calculation of the complex output dependencies such as the example shown in Figure 5.d, requires logic implication computation and is computationally intractable. However, according to our results, the complex output dependencies have less effect in the computed error propagation probabilities ($EPP_{n_i \to O_j}$).

		#	#gates	Our Time (sec)		Sim. Time (sec)		%ave.	SP	Speedup	
circuit	m+n	gates	simul.	per gate	Total	per gate	Total	Diff	Time	ISP	ESP
s820	23	289	289	0.00013	0.04	19.39	5604	3.9	147.53	38.0	142600
s832	23	287	287	0.00010	0.03	19.43	5578	3.9	149.15	37.4	188720
s838	66	446	446	0.00048	0.23	46.90	20919	3.0	253.31	82.5	97520
s953	45	395	395	0.00035	0.15	28.30	11178	4.3	150.17	74.4	79950
s1196	32	529	529	0.00075	0.41	54.60	28883	3.6	313.01	92.2	72800
s1238	32	508	508	0.00053	0.28	36.97	18784	3.4	207.66	90.3	69510
s1423	91	657	657	0.00223	1.63	53.10	34891	3.9	250.39	138.5	23810
s1488	14	653	653	0.00042	0.28	7.31	4778	4.4	14.83	316.3	17220
s1494	14	647	647	0.00070	0.46	10.89	7045	4.4	22.76	303.7	15480
s5378	214	2779	100	0.00155	4.61	222.9	NA	15.0	1208	510.9	143070
s9234	247	5597	52	0.00936	54.41	817.2	NA	11.3	4659	970.8	87230
s15850	611	9772	136	0.03417	352.24	972.1	NA	12.6	5270	1695.1	28440
s35932	1763	16065	110	0.00702	124.9	1904.1	NA	4.5	9648	3133.9	271240
s38584	1464	19253	77	0.01386	286.61	2317.1	NA	7.1	12833	3405.5	167180
s38417	1664	22179	85	0.01418	292.20	2412.4	NA	6.0	12951	3480	170126
ave.	-	-	-	0.00324	40.00	325	NA	5.4	110.77	549.1	93072

Table 2. Comparison of our systematic approach to random simulation
m+n: Number of FFs plus the number of PIs NA: Not Available because of very long simulation time.
ISP: Including SP computation time in the analytical approach
ESP: Excluding SP computation time from analytical approach

We have further classified the output dependency shown in Figure 5.d to the cases shown in Figure 5.e and Figure 5.f. In Figure 5.e and Figure 5.f an erroneous value can propagate to O_k, O_j, or both of them. In these two figures, if one assumes that the off-path signals are independent, the computed system failure probability using the proposed approach would be accurate. As an example, considering the signal probabilities given in Figure 5.e $EPP_{A \to O_j}$ equals to $0.8 \times (1 - 0.4) = 0.48$ and $EPP_{A \to O_k}$ equals to $0.8 \times 0.3 = 0.24$. Therefore, the computed system failure probability would equal to $SF(A) = P_{SEE}(A) \times (1 - (1 - 0.48) \times (1 - 0.24)) = 0.6048 \times P_{SEE}(A)$. However, if signal probability of off-path signals are correlated and dependent to each other, it will introduce a negligible inaccuracy in the computed system failure probability. As will be shown in the experimental results, the computed system failure probability using the proposed approach is as accurate as those computed by fault-simulations while orders of magnitude faster than the fault-simulation method.

3.4 Experimental Results

The proposed approach has been implemented and applied to ISCAS'89 benchmark circuits. All experiments have been performed on the DELL Precision 450© system equipped with 2 GB main memory. Table 2 shows the results for the presented analytical approach as well as the random simulation for combinational logic core of ISCAS'89 (sequential) circuits. In the random simulation method, a small fraction of 2^{m+n} vectors are needed to compute system failure probability of a sequential circuit, where m is the number of bistables and n is the number of primary inputs ($m + n$ is the total number of inputs of combinational core). For large circuits, such simulations for all internal gates are intractable. So, in large circuits, random simulations are performed on a fraction of the circuit. Due to this fact, the speedups reported in Table 2 are smaller than the actual value for larger circuits.

In order to verify the accuracy of the proposed technique, we have developed a simulation-based fault injection engine using Monte-Carlo (MC) simulations. The simulation-based fault injection engine works based on timing-accurate at the gate-level. For each logic gate, we have randomly injected an SEE at the output of the logic gate. Then, we apply several random vectors to the primary inputs in order to measure the probability that the injected SEE can propagate to the primary outputs. The Monte-Carlo simulation terminates if the accuracy of the estimated system failure probability falls within a pre-defined confidence interval.

The time required for propagation probability computation using the analytical approach varies from less than one second for small circuits to at most 5 minutes for the largest circuits. For larger circuits, a limited number of gates of the circuits has been simulated due to extremely long simulation time of the random-simulation method. To show the efficiency of our approach, the speedups are reported in two ways, a) excluding the SP computation time from the total run time, b) including the SP computation time in the total computation time. When SP time is excluded, the speedups are 4-5 orders of magnitude. When included, our approach is still 2-3 orders of magnitude faster than the random simulation method. One interesting result is that the propagation probability computation time per gate using our approach increases almost linearly with the circuit size. That is the reason for the scalability of the presented approach.

As shown in Table 2, the difference between the results of our approach and random simulation is about 5.4%, on average. There are some circuits in which the difference between the analytical approach and the random-simulation method is larger. This is mainly due to the fact that the random-simulation method was so time-consuming that makes it so difficult to simulate a large enough sample of inputs. In other word, the difference is partly due to the inaccuracy of random-simulation approach for those circuits. In particular, results presented in next section further verify that the variance of computed EPPs using the proposed approach is negligible with different variances of SPs.

For the largest circuits shown in Table 2, SER estimation using the MC simulation method takes more than 2000 seconds per logic gate while using the proposed analytical method it takes about 0.01 second per logic gate. This makes the proposed approach applicable to very large ASIC designs used in industry. Comparing our proposed approach to previous techniques detailed in Sec. 2, our proposed approach provides very accurate SERs while maintaining the execution time very low. Those techniques which are based on fault-injection or Monte-Carlo simulation can provide accurate results but the corresponding execution time for large circuits is completely intractable as shown in Table 2. On the other hand, there are some techniques that can provide SERs for larger circuits within a short amount of time but such techniques do not consider the effect of reconvergent fanouts on the computed SERs [36, 35].

4 Uncertainty Bounds for Estimated EPP Values

The previous section presented an analytical method to obtain EPP values for SER estimation based on SP values. This section describes a methodology to obtain bounded accuracy (variance) of estimated EPP values based on the accuracy (variance) of SP values.

Let $P(I_i)$ and $V(I_i)$ be the value and *uncertainty bound* (variance) of the i^{th} input, and $P(O)$ and $V(O)$ be the value and variance of the output. The following formulas can be used to compute addition/subtraction and multiplication operations [53, 54].

$$Add/Sub: P(O) = \sum_{i=1}^{n} P(I_i) \Longrightarrow \quad V(O) = \sqrt{\sum_{i=1}^{n} V^2(I_i)} \tag{5}$$

$$Mult: P(O) = \prod_{i=1}^{n} P(I_i) \Longrightarrow \quad V(O) = P(O) \times \sqrt{\sum_{i=1}^{n} \frac{V^2(I_i)}{P^2(I_i)}} \tag{6}$$

Equations (5) and (6) are used to calculate variances in our computation. To compute the variances of an $n-$input AND gate, which computes $P_1(O)$, $P_0(O)$, $P_a(O)$, and $P_{\bar{a}}(O)$ according to Table 1 and Equations (5) and (6), the following formulas can be used:

$$V_1(O) = P_1(O) \times \sqrt{\sum_{i=1}^{n} (\frac{V_1(I_i)}{P_1(I_i)})^2} \tag{7}$$

$$V_a(O) = \sqrt{V_1^2(O) + [P_1(O) + P_a(O)]^2 \times \sum_{i=1}^{n} \frac{V_1^2(I_i) + V_a^2(I_i)}{[P_1(I_i) + P_a(I_i)]^2}} \tag{8}$$

$$V_{\bar{a}}(O) = \sqrt{V_1^2(O) + [P_1(O) + P_{\bar{a}}(O)]^2 \times \sum_{i=1}^{n} \frac{V_1^2(I_i) + V_{\bar{a}}^2(I_i)}{[P_1(I_i) + P_{\bar{a}}(I_i)]^2}} \tag{9}$$

$$V_0(O) = \sqrt{V_1^2(O) + V_a^2(O) + V_{\bar{a}}^2(O)} \tag{10}$$

Equation (7) is directly derived from Equation (6) and the first equation of Table 1. Equation (8) can be derived by combining the second equation of Table 1 with Equation (5) and Equation (6), as follow:

$$Variance(\prod_{i=1}^{n} [P_1(X_i) + P_a(X_i)]) = [P_1(O) + P_a(O)] \times \sqrt{\sum_{i=1}^{n} \frac{V_1^2(I_i) + V_a^2(I_i)}{[P_1(I_i) + P_a(I_i)]^2}}$$

$$\Longrightarrow Variance(\prod_{i=1}^{n} [P_1(X_i) + P_a(X_i)])^2 = [P_1(O) + P_a(O)]^2 \times \sum_{i=1}^{n} \frac{V_1^2(I_i) + V_a^2(I_i)}{[P_1(I_i) + P_a(I_i)]^2} \tag{11}$$

On the other hand, $Variance(P_1(O)) = V_1(O)$. This along with Equation (11) derives Equation (8). Equation (9) and Equation (10) can be obtained similarly.

In the above formulas, while Equations (7), (8), and (9) keep the variances in the same order of size, the last formula (Equation (10)) causes that the variances get increased after each logic stage. This is because,

$$For\ Equation\ (7):\ V_1(I_i) << P_1(I_i) \Rightarrow V_1(O) \lessapprox V_1(I_i) \tag{12}$$

$$For\ Equation\ (8):\ \sum_{i=1}^{n} \frac{V_1^2(I_i) + V_a^2(I_i)}{[P_1(I_i) + P_a(I_i)]^2} \lessapprox 1 \Rightarrow V_a(O) \approx V_1(O) \tag{13}$$

$$For\ Equation\ (9):\ \sum_{i=1}^{n} \frac{V_1^2(I_i) + V_{\bar{a}}^2(I_i)}{[P_1(I_i) + P_{\bar{a}}(I_i)]^2} \lessapprox 1 \Rightarrow V_{\bar{a}}(O) \approx V_1(O) \tag{14}$$

$$For\ Equation\ (10):\ V_0(O) > V_1(O)\ , V_0(O) > V_a(O)\ , V_0(O) > V_{\bar{a}}(O) \tag{15}$$

In other words, $V_1(O)$, $V_a(O)$ and $V_{\bar{a}}(O)$ are less than or at least in the same order of inputs variances ($V_1(I_i)$, $V_0(I_i)$, $V_a(I_i)$ and $V_{\bar{a}}(I_i)$) but $V_0(O)$ is bigger than all inputs variances at each stage. To resolve this problem, if we compute $P_0(O)$ and $V_0(O)$ according to Equation (16) and Equation (17), the corresponding variance gets significantly reduced compared to Equation (10).

$$P_0(O) = 1 - \prod_{i=1}^{n} [P_1(I_i) + P_a(I_i) + P_{\bar{a}}(I_i)] = 1 - \prod_{i=1}^{n} [1 - P_0(I_i)] \tag{16}$$

The corresponding variance equals to:

$$V_0(O) = (1 - P_0(O)) \times \sqrt{\sum_{i=1}^{n} [\frac{V_0(I_i)}{1 - P_0(I_i)}]^2} \tag{17}$$

Variance propagation probability rules have been summarized in Table 3.

GATE	RULE
AND	$V_1(O) = P_1(O) \times \sqrt{\sum_{i=1}^{n} (\frac{V_1(I_i)}{P_1(I_i)})^2}$
	$V_a(O) = \sqrt{V_1^2(O) + [P_1(O) + P_a(O)]^2 \times \sum_{i=1}^{n} \frac{V_1^2(I_i)+V_a^2(I_i)}{[P_1(I_i)+P_a(I_i)]^2}}$
	$V_{\bar{a}}(O) = \sqrt{V_1^2(O) + [P_1(O) + P_{\bar{a}}(O)]^2 \times \sum_{i=1}^{n} \frac{V_1^2(I_i)+V_{\bar{a}}^2(I_i)}{[P_1(I_i)+P_{\bar{a}}(I_i)]^2}}$
	$V_0(O) = (1 - P_0(O)) \times \sqrt{\sum_{i=1}^{n} [\frac{V_0(I_i)}{1-P_0(I_i)}]^2}$
OR	$V_0(O) = P_0(O) \times \sqrt{\sum_{i=1}^{n} (\frac{V_0(I_i)}{P_0(I_i)})^2}$
	$V_a(O) = \sqrt{V_0^2(O) + [P_0(O) + P_a(O)]^2 \times \sum_{i=1}^{n} \frac{V_0^2(I_i)+V_a^2(I_i)}{[P_0(I_i)+P_a(I_i)]^2}}$
	$V_{\bar{a}}(O) = \sqrt{V_0^2(O) + [P_0(O) + P_{\bar{a}}(O)]^2 \times \sum_{i=1}^{n} \frac{V_0^2(I_i)+V_{\bar{a}}^2(I_i)}{[P_0(I_i)+P_{\bar{a}}(I_i)]^2}}$
	$V_1(O) = (1 - P_1(O)) \times \sqrt{\sum_{i=1}^{n} [\frac{V_1(I_i)}{1-P_1(I_i)}]^2}$

Table 3. Variance propagation probability rules for elementary gates

4.1 Experimental Results

In order to verify and compare the accuracy of the proposed technique, we have developed a simulation-based fault injection engine using Monte-Carlo (MC) simulation. MC simulation and fault injection allow us to obtain both EPP values and their variances for the simulation flow, as our reference. The proposed analytical approach was implemented and applied to ISCAS89 sequential benchmark circuits.

Note that in all experiments, we have excluded the confidence level factor from the variance results. So, assuming confidence level of 99% (90%). all variances should be divided by $z_{\alpha/2} = 2.576$ ($z_{\alpha/2} = 1.645$). Figure 6 shows the average variances when propagating errors along the paths from an arbitrary error site to any arbitrary FF/PO for different variances of signal probabilities (1%, 5%, 10%, and 20%). As discussed in Sec. 3.1, Equation (3) is used to

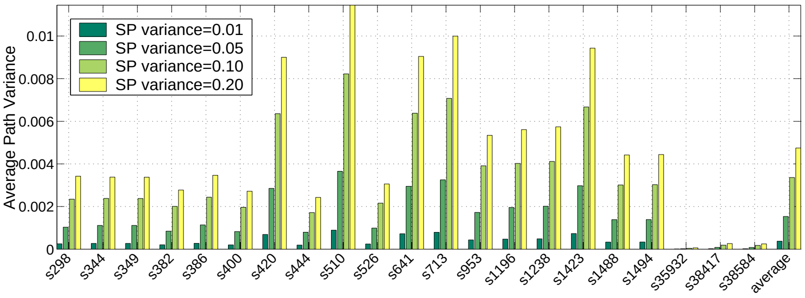

Figure 6. Average variance of EPP values along the path

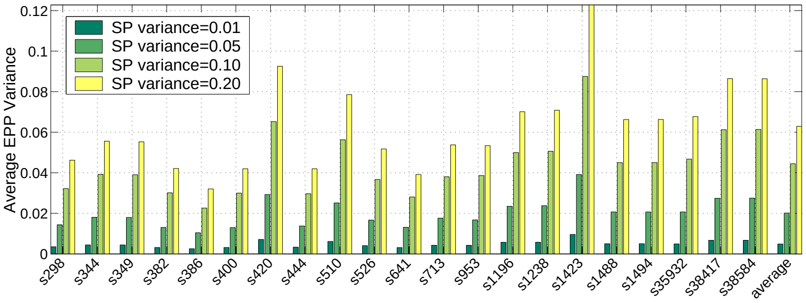

Figure 7. Average variance of the overall EPP values

compute the EPP of a given node, to account for the effect of error propagation to multiple outputs from a given error site. The corresponding variances after EPP computation have been reported in Figure 7. These results show that the variances of the EPP values grow sub-linearly with the variances of SP values. For instance, the variance of EPP values (both path and overall) increases to 1.38 when the variance of SP values doubles (from 10% to 20%). Note that the (runtime) complexity of SP estimation is exponentially related with the required variances.

Also, average (per-gate) EPP values for different types of error sites such as combinational gate outputs or latch outputs, and different observation points such as primary outputs (PO) or latch inputs, have been depicted in Figure 8. These cases are gate-to-latch, gate-to-PO, latch-to-latch, latch-to-PO, gate-to-latch/PO, and finally latch-to-latch/PO. Detailed EPP histograms for two largest ISCAS'89 circuits (s35932 and s38417) have been shown in Figure 9 and Figure 10, respectively.

Finally, Figure 11 and Figure 12 show the run time of the presented systematic analytical approach (sys) and the Monte-Carlo simulation (sim) based on different variances of signal probability values. The results show that the analytical time including variance calculation time for the ISCAS'89 largest circuits is less than 10 minutes while the MC simulation time for these circuits is bigger than 10^7 seconds (variances=0.01). As can be seen in these figures, SP estimation time exponentially grows for smaller variances.

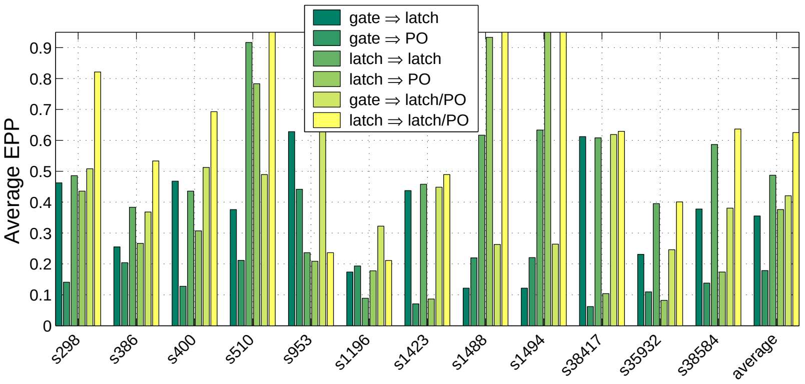

Figure 8. Average EPP for gate-to-latch, gate-to-PO, latch-to-latch, latch-to-PO, gate-to-latch/PO, and latch-to-latch/PO

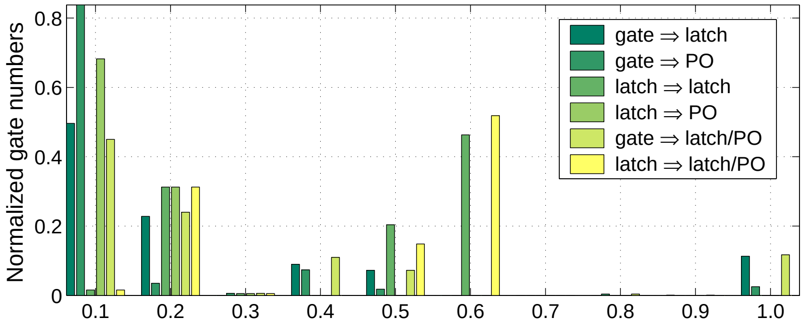

Figure 9. Detailed EPP results of s35932 for gate-to-latch, gate-to-PO, latch-to-latch, latch-to-PO, gate-to-latch/PO, and latch-to-latch/PO

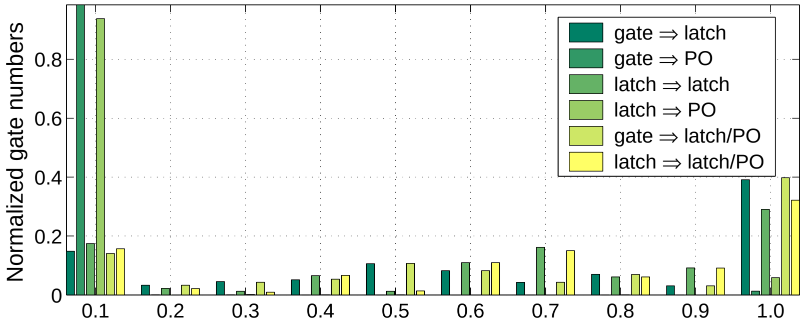

Figure 10. Detailed EPP results of s38417 for gate-to-latch, gate-to-PO, latch-to-latch, latch-to-PO, gate-to-latch/PO, and latch-to-latch/PO

4.2 Sensitivity to SP Variance of Individual Gates

Since signal probability estimation with high accuracy (i.e. low uncertainty bound) can be time consuming for large circuits , it is possible that the signal probabilities for some gates in the circuit are estimated with low accuracy. We have performed a set of experiments to analyze the impact of this effect on the overall accuracy of EPP estimation. In this experiment, we have considered 5% uncertainty bounds in the signal probabilities of all gates for some ISCAS'89 benchmark circuits. We have randomly chosen 100 to 1000 gates from these circuits (depending on the size of the circuit) and increased their signal probability variances. Then, the variance of overall EPP values have been re-computed. The results are presented in Figure 13. Each data point corresponds to the average variance of overall EPP due to change in the SP variance of selected gates. These results clearly show that this method has very small sensitivity to the SP variance of particular gates.

4.3 Effect of Different Implementations on Variance Propagation

We have considered three different implementations of a 16-input AND function to study the implementation effects on the variance of analytical EPP values. The implementations are 1) a single 16-input AND gate (wide), 2) cascade chain consisting of 15 2-input AND gates, and 3) a balanced binary tree implementation consisting of 15 2-input AND gates.

We have reported the results for some of primary inputs in Table 4. Each row represents the EPP and its variance for the output due to error at each particular input. These have been computed using both our presented approach and the MC-simulation. Due to simplicity of the function, it is also possible to accurately calculate the exact EPP values (zero variance). It can be seen that the first implementation (wide-input) has the lowest variance (best accuracy) among the three implementations. Also, the cascade chain has the highest variances due to large combinational depth. These results show that the presented analytical approach can perform very well for early-stage designs obtained from high-level synthesis (i.e. before optimization and technology mapping) in which there are several very wide input gates in the netlist.

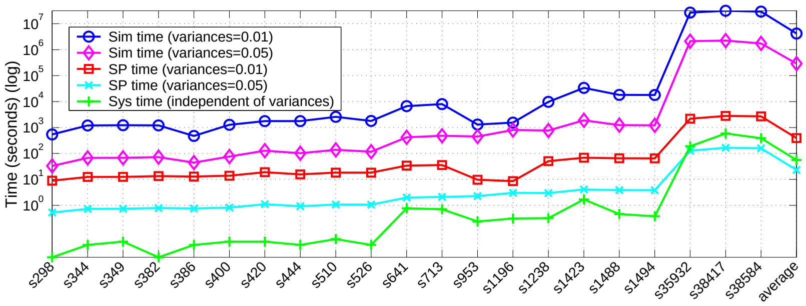

Figure 11. Run time of analytical error-bounded EPP estimation vs. MC fault simulation for small variance values (0.01, 0.05)

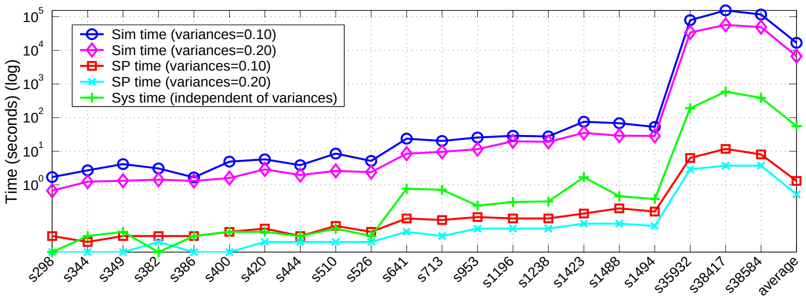

Figure 12. Run time of analytical error-bounded EPP estimation vs. MC fault simulation for large variance values (0.10, 0.20)

4.4 Discussion

4.4.1 Mixed Analytical-Fault-Simulation Approach

It is possible that for very long combinational paths (largest circuits), the maximum path variance becomes greater than the threshold variance. Based on the formulas presented for uncertainty bound estimation, this situation may occur when the depth of combinational logic along a path is more than 20 gates. Although such very deep combinational logic rarely happens in real designs, we present a mixed analytical and *fault simulation* (FS) method to selectively reduce the variance and boost up the accuracy.

In this mixed analytical-FS method, once the variance of the EPP values at the output of a gate exceeds the predefined threshold, we use a special fault simulation method to re-calculate the EPP values (P_1, P_0, P_a, and $P_{\bar{a}}$) by performing a Monte-Carlo fault simulation only for the logic cone driving that particular gate. The simulation is performed such that the uncertainty bounds (variances) of EPP values fall within the threshold bounds. Note that this may happen only for very deep gates along the path, and once the uncertainty bounds are re-adjusted, the analytical method can be used for the remaining gates along the path. Again, this fault simulation is not required to be performed for the entire circuit; only the logic cone driving that particular gate needs to be fault simulated. So, the time complexity of this special MC fault simulation is tractable and the effect on the overall run time should not be considerable.

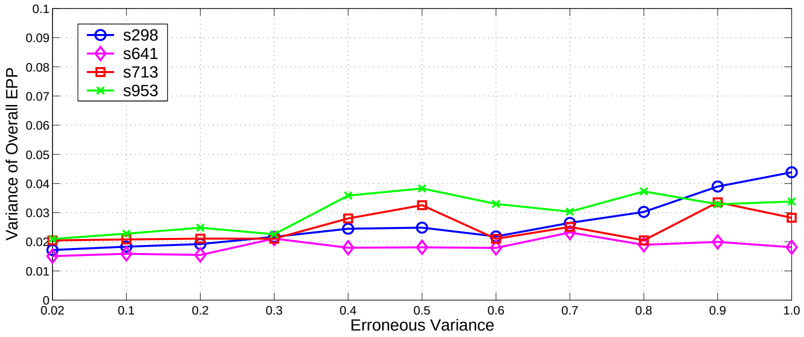

Figure 13. Sensitivity to SP Variance of Individual Gates

	EPP					**Variance**				
Input	Simple	Cascade	Binary	MC	Exact	Simple	Cascade	Binary	MC	Exact
In 1	0.000031	0.000031	0.000027	0.000000	0.0000305	0.000001	0.000001	0.000004	0.0033	0
In 2	0.000030	0.000030	0.000027	0.000000	0.0000305	0.000001	0.000001	0.000004	0.0033	0
In 3	0.000031	0.000031	0.000027	0.000000	0.0000305	0.000001	0.000001	0.000004	0.0033	0
In 14	0.000031	0.000015	0.000028	0.000000	0.0000305	0.000001	0.000015	0.000004	0.0033	0
In 15	0.000030	0.000000	0.000028	0.000000	0.0000305	0.000001	0.000030	0.000004	0.0033	0
In 16	0.000031	0.000000	0.000028	0.000000	0.0000305	0.000001	0.000032	0.000004	0.0033	0

Table 4. EPP and variance comparison of three different implementation of a 16-input AND gate

4.4.2 Electrical Masking

As mentioned in Sec. 2, electrical masking is one of the factors that affects the circuit SER. Recently, there has been some research addressing electrical masking [10, 1, 55, 15]. These methods either use the *capacitance model* or the *drain current model*, which effectively capture the non-linear property of the CMOS transistors [55]. By implementing either of these models using SPICE simulations, logic cells can be pre-characterized in the target library for any arbitrary input SEE (charge values). After the cell pre-characterization phase, a lookup table can be developed to compute *electrical attenuation factor* for different values of SEE and logic cell properties (e.g. SET pulse width, SET magnitude, or cell fanout capacitance). To have more accurate results, more entries can be added to the lookup table. More detailed description on cell characterization can be found in [10].

The capacitance model presented in [10] can be incorporated in our methodology to compute the electrical masking factor of logic gates. Note that SPICE simulation for primary logic cells is accomplished in a small fraction of a second. So, pre-characterization phase is not a time-consuming process and it has to be performed only once (i.e. not in a per-circuit basis). To incorporate the effect of the electrical masking factor in our methodology, the logical masking factor needs to be multiplied by the attenuation factor when computing $P_a(U_i)$ and $P_{\bar{a}}(Ui)$. Specifically, for each logic gate output, one extra parameter, the magnitude of the propagated transient due to the electrical masking effect, has to be associated with each propagation probability, $P_a(U_i)$ and $P_{\bar{a}}(Ui)$. Similar to the propagation probability rules presented in Table 1, the attenuation lookup tables are used to compute the magnitudes of the propagated values at the output of the gate based on the magnitudes of the transients at the inputs of the gate and the fanout of the gate.

5 Soft Error Remediation in Combinational Logic

The ability of a CMOS logic gate to tolerate SEEs is a function of the SEE injected charge and the transistor size ratio [23]. By increasing the size of transistors (transistor *upsizing*), it is possible to absorb the transient pulse caused by the injected charge, preventing its propagation to the next logic stage. Therefore, transistor upsizing is a possible solution to reduce soft error susceptibility of logic gates. Nevertheless, this approach imposes area and delay penalties [16]. A practical approach is selective gate sizing based on SER reduction requirements as well as cost (area, delay, and power consumption) budgets. While transistor upsizing reduces the logic SER by absorbing the injected charge, transistor *downsizing* (decreasing the transistor size) can also improve the logic SER by reducing the sensitive region of the transistors [56]. As transistors are scaling down, the sensitive region of transistors becomes smaller which reduces the magnitude of transient glitches.

It has been demonstrated that the size of a gate driving a node and the amount of capacitance at the node determine the magnitude and duration of the SET transient [23]. A large transistor can dissipate the charge injected by particle hit so that the effect cannot be propagated to the gate output. As transistor aspect ratio ($\frac{W}{L}$) increases, the transient pulse width due to SEE decreases in the same proportion. As the result, the optimal transistor sizing is a linear function of the injected charge. Based on the energy level of particle and device characteristics, the amount of injected charge can be calculated. Using this information, the size of the logic gate for completely absorbing transient pulse will be determined. However, increasing the size of a gate increases its area and power consumption accordingly. Moreover, the delay of the driving gate (previous stage) will be also increased. Therefore, it is not practical to resize (upsize) all logic gates in the circuit for SER reduction.

EPP of the gates can be used as a metric to measure the contribution of each gate to the overall soft error rate. A higher EPP means that a bit-flip at the output of the gate will more likely cause an error at the circuit primary outputs. The technique presented in [16] exploits the information regarding EPP of the gates in the circuit to use transistor sizing for gates with highest EPPs. Although this approach can reduce the area overhead associated with gate sizing, it does not particularly address delay penalties since it does not consider the *timing slack* of the gates chosen for resizing. Note that if a gate in the critical path (i.e. with zero slack) is resized, the overall delay of the circuit will be also increased. However, if a non-critical gate is resized so that its slack remains non-negative after resizing, the

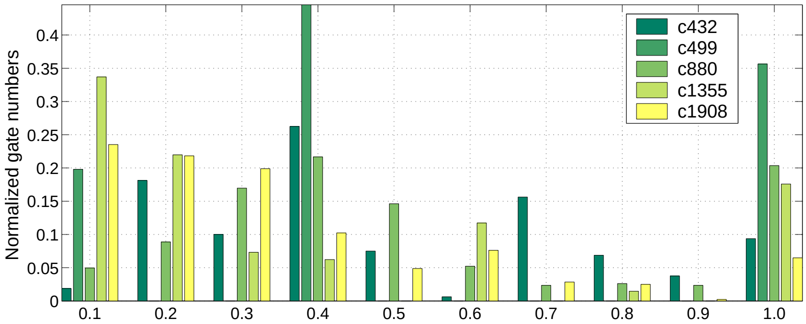

Figure 14. EPP distribution in the circuit

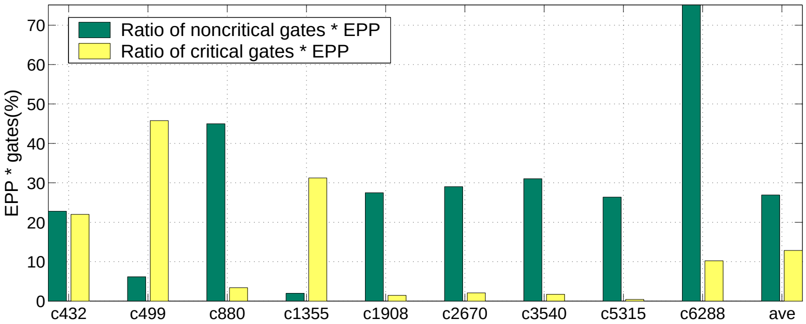

Figure 15. EPP of critical and non-critical gates

overall circuit delay is not affected. Consequently, for cost-effective SER reduction, timing and layout information as well as EPPs have to be considered.

This section presents efficient gate sizing techniques for SER reduction in which the contribution of each logic gate to the system-level SER and its criticality in the overall performance are carefully considered. Fast and efficient soft error vulnerability minimization algorithms under different constraints, such as area, delay, or both, are also presented.

5.1 Proposed Soft Error Remediation Technique

Our proposed soft error remediation technique uses the fact that most logic gates of circuits reside on non-critical paths and only small ratio of logic gates are on critical paths. In particular, our analysis on ISCAS'85 benchmark circuits shows that 75% of logic gates are non-critical (positive timing slack). This means that resizing of these gates will not impose any performance penalty as long as the slacks of the resized gates are still non-negative.

Figure 14 shows the distribution of error propagation probabilities of the logic gates for some ISCAS'85 benchmark circuits. These values are normalized to the number of gates in each circuit. For instance, on average, 63% of gates in these circuits have EPP of 0.4 or less. In other words, a small subset of gates contributes to the majority of system-level SER. Figure 15 shows the product of the average EPP and the number of critical and non-critical gates for ISCAS'85 circuits. It can be seen that the share of non-critical gates in the overall SER is twice more than that for critical gates. This ratio is even bigger in larger circuits of this benchmark set. This shows an opportunity to hide the delay overhead associated with SER reduction. For example, one can expect that a slack-aware approach for gate resizing can reduce the SER to one third (3x reduction) with no performance overhead.

5.2 Timing-Aware SER Reduction

In this approach, both EPP and timing slack of logic gates are considered in the resizing selection [57]. Note that changing the size of one gate affects not only its own timing slack, but also the slack of other logic gates. Figure 16, as an example, shows the gates whose timing slacks are affected due to resizing one particular gate (changing its input capacitance and its propagation delay). The slack-aware resizing algorithm is outlined below.

1. Sort the gates based on their EPPs in the descending order.
2. At each step, choose a gate from the top of this list as a candidate for resizing.
3. If the slack of this gate is large enough such that after resizing its slack remains non-negative, this gate is resized.

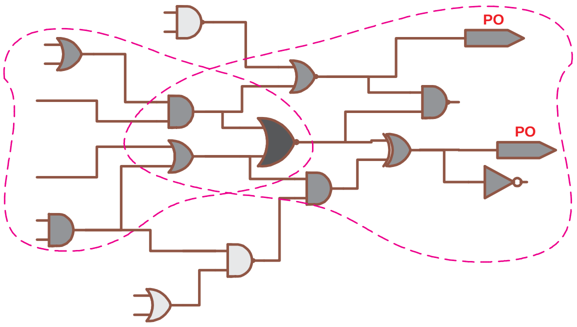

Figure 16. Gates affected by slack change in a gate

- The delay of this resized gate along with those gates driving and being driven by this are recomputed.
- The slacks of all gates in the forward and the backward logic cones originated from the resized gate are recalculated and updated (Figure 16).

4. Otherwise, discard this candidate gate and examine the next gate in the list (go to step 2).
5. Repeat this process until the list becomes empty.

The above procedure minimizes the SER without introducing any performance penalty. In other words, we only resize those logic gates that have positive timing slack. In order to minimize SER under performance constraints, extra *user-defined* timing budget can also be considered in the above procedure. This extra budget can be expressed as a percentage of the original delay of the circuit. To incorporate this, an initial step is required to be added to the above procedure in which the slack of all gates are adjusted based on the additional delay. The remainder of the algorithm remains the same. Note that an SETs can occur on non-critical paths such that it arrives right on the latching window of bistables. As an example, let's consider a non-critical path with delay of 600ps within a circuit whose clock cycle time is 800ps. If an SET occurs 200ps before rising edge of the previous clock cycle, it can arrive right within the latching period of the next clock cycle.

We use the example shown in Figure 17 to show how this procedure is applied. In this figure, G4, G5, G6, G7, and G9 are in the critical path. We first sort the non-critical gates according to the their EPPs. ($Gi(x)$ means that the EPP of Gi is equal to x.)
Non-critical gate list= $\{G3(1), G8(1), G12(1), G11(0.96),$
$G14(0.76), G2(0.53), G10(0.51), G13(0.37), G1(0.36)\}$.

As indicated in [58, 23], the amount of charge collected due to neutrons can vary from $10fC$ to $150fC$ (or $0.15pC$). Here let's assume that the particle energy is $0.15pC$. The resizing process is done in such a way that it immunes the gates against SEEs with this energy. To do so, we need to resize the transistors to $\frac{W}{L} = 4$, i.e. 4x sizing [23]. We start from the gates with the highest EPP. G3 has a considerable slack of 29.5 ps. After resizing this gate, we recompute the slacks of the forward and backward logic cones of this gate. The next candidate in the non-critical list is G8 and its slack is 18 ps. Although this gate is not on the critical path, once it is resized, the critical path delay will be affected. This is because if G8 is resized, the load capacitance of G6, which is on the critical path, will be affected. So, we discard this gate.[3] We can successfully resize the next candidates (G12, G11, G14, G10, G2, G13, and G1) using the available slack budget. In this example, we assumed that the primary inputs provide the required current to the next gate levels. If we buffer the primary inputs (using a BUF gate), we will no longer be able to resize the last three gates of the non-critical list (G10, G13, and G1). In this example, the SER of this circuit is reduced by almost 60% while the circuit performance has not been affected.

5.3 General Optimization Algorithms for SER Reduction

The problem of maximizing SER reduction with minimum area and delay overhead, as an optimization problem, is addressed here. The heuristic algorithm presented in Sec. 5.2 tries to maximize SER reduction with bounded delay overhead. Due to the heuristic nature of the presented algorithm, achieving the "maximum" SER reduction is not always guaranteed. Here, we look at other variations of this problem and investigate possible heuristic solutions.

[3] This gate could be resized if 5% extra delay overhead were considered.

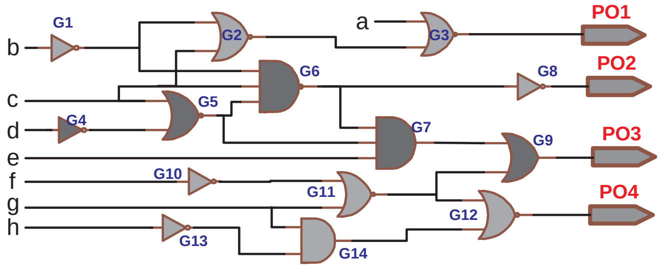

Figure 17. A sample logic circuit used for our proposed hardening technique

5.3.1 Maximizing SER Reduction Under Area Constraints

In order to maximize SER reduction with minimum area overhead, both EPP (as a metric for the *benefit* in SER reduction) and the area (as a metric for cost) of each gate must be taken into account. This optimization problem can be converted into the classical *knapsack* problem [52]. In knapsack problem, there are n items, each item i has value v_i and weight w_i. The objective is to select a subset of these items such that the sum of weights of selected items is not greater than W (the size of knapsack) while the sum of their values is maximized.

This optimization problem is proven to be NP-hard [52]. However, a heuristic solution to this problem is to sort items based on $\frac{v_i}{w_i}$ (value per unit weight) in the descending order. Then, the items will be chosen from this list as long as the sum of weights of selected items does not exceed W. The time complexity of this heuristic is $O(n \log n)$ since the sorting part is $O(n \log n)$ and the selection part is $O(n)$.

The same heuristic can be used for this SER reduction problem by sorting the logic gates based on $\frac{EPP_i}{\Delta area_i}$ in the descending order, where EPP_i and $\Delta area_i$ are the EPP and area increase (due to sizing) of gate g_i, respectively. Given a user-specified area budget A, gates in this list are resized until the area increase exceeds the area budget ($\sum \Delta area_i > A$). Similar approach can be used for power-constrained SER minimization in which power increase is used instead of area increase.

5.3.2 Maximizing SER Reduction Under Delay Constraints

This problem is basically the original problem addressed in Sec. 5.2. Note that unlike the previous problem (Sec. 5.3.1), this optimization problem cannot be directly converted to the knapsack problem. This is because resizing one gate (changing its delay) affects the timing of other gates, as well. This is why our presented solution in Sec. 5.2 is not the same as the straightforward heuristic for the knapsack problem.

5.3.3 Maximizing SER Reduction Under Area and Delay Constraints

Given user-specified area and delay budgets, the objective here is to maximize SER reduction such that delay and area overhead does not exceed the specified budget. Since this problem is a combination of the two previous problems, we can use a heuristic based on the algorithms presented in Sec. 5.2 and Sec. 5.3.1. Specifically, we sort logic gates based on $\frac{EPP_i}{\Delta area_i}$ in the descending order. While choosing each gate from the top of this list for resizing, we consider the timing slack of the circuit after resizing that particular gate, and compare it with the timing budget. If resizing this gate results in violation of the timing budget, this gate is discarded (not resized) and the next gate in the list is considered.

5.4 Experimental Results

These heuristic slack-aware resizing algorithms have been implemented and applied to ISCAS'89 sequential circuits which have larger combinational logic cores compared to pure combinational ISCAS'85 circuits. Figure 18, Figure 19, and Figure 20 show the SER reduction achieved by delay-constrained, area-constrained, and area/delay constrained resizing techniques, respectively. Different values of delay and area budget have been considered as the user-defined constraints. The gates chosen for sizing are resized four times (4x) their original sizes to completely block the propagation of the injected charge. As can be seen in these figures, the SER of these circuits can be greatly reduced with modest area and/or delay penalties. In particular, the results show that the overall SER can be reduced, on average, by more than 6x (84%) without any performance penalty (shown in Figure 18). Also, this figure shows that the circuits SER can be reduced by more than 10x with only 10% delay overhead.

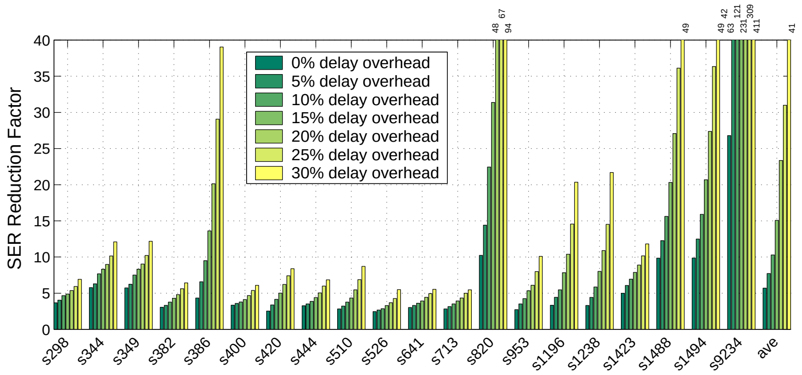

Figure 18. Soft error vulnerability reduction using slack-aware resizing technique with bounded delay overhead

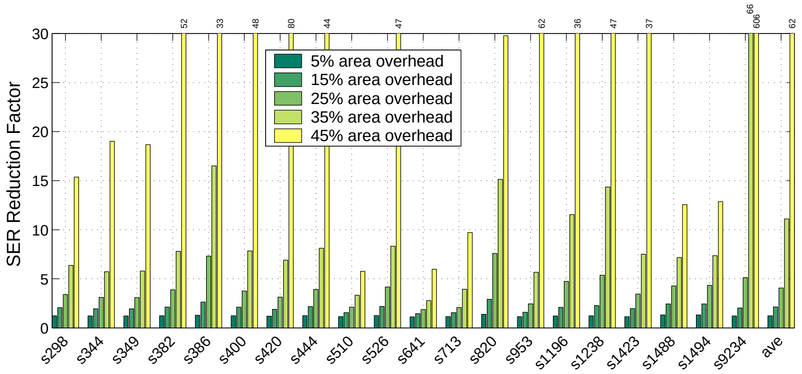

Figure 19. Soft error vulnerability reduction using slack-aware resizing technique with bounded area overhead

6 Conclusions

Soft errors due to cosmic radiations are the main reliability threat of digital systems. In particular, vulnerability of ASICs grows in direct proportion to the Moore's law. Therefore accurate estimation of SERs and efficient remediation methods are critical for achieving reliable computing in successive technology nodes.

When estimating the soft error rates, it is very critical to examine the accuracy of the obtained values. In this work, we have presented an approach to analytically estimate the error propagation probabilities (EPPs) (used in SER estimation) along with their uncertainty bounds (variances) for a logic network. The experimental results show that the maximum variances along the deepest combinational paths in the benchmark circuit is quite small (0.04 for the signal probability values estimated with the variance of 0.05). Moreover, the sensitivity of the computed SERs using proposed approach to the accuracy of SP values is substantially sub-linear (0.38x). This means that by spending exponentially less time in obtaining less accurate SP values, the overall accuracy of estimated SER values will not be proportionally compromised. We have also presented a mixed analytical and fault simulation method to boost up the accuracy of the EPP estimation method for very deep combinational paths.

Moreover, a gate-level SER remediation technique with bounded area and delay penalties has been presented. We have developed gate resizing algorithms for the entire circuit in which the error propagation probability, area overhead, and timing slack of each gate are carefully considered. Different versions of the SER minimization problem under different area and delay constraints have been addressed. Our results show that more than 6x reduction in the overall SER can be achieved without any performance penalty.

References

[1] H. T. Nguyen, Y. Yagil, N. Seifert, and M. Reitsma. chip-level soft error estimation method. *IEEE Transactions on Device and Materials Reliability*, 5(3), September 2005.

[2] E. Normand. Single Event Upset at Ground Level. *IEEE Transactions on Nuclear Science*, 43(6), December 1996.

[3] R.C. Baumann. Radiation-induced soft errors in advanced semiconductor technologies. *IEEE Transactions on Device and Materials Reliability*, 5(3):305–316, September 2005.

[4] S. Mitra W. Bartlett and P. Sanda. Workshop recap. In *Proceedings of the 2nd Workshop on Silicon Errors in Logic - System Effects (SELSE)*, April 2006.

[5] Fujitsu Corporation. International Solid-State Circuits Conference (ISSC), 2003.

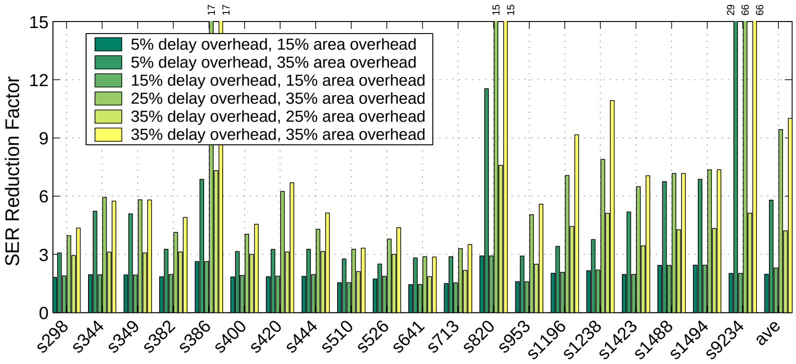

Figure 20. Soft error vulnerability reduction using slack-aware resizing technique with bounded area/delay overhead

[6] 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. In *Proceedings of the 36th Annual IEEE/ACM International Symposium on Micro-architecture (MICRO-36)*, pages 29–40, 2003.

[7] L.B. Freeman. Critical charge calculations for a bipolar SRAM array. *IBM Journal of Research and Development*, 40(1):119–129, January 1996.

[8] R. C. Baumann. Soft errors in advanced computer systems. *IEEE Design & Test of Computers*, 22(3):258–266, May-June 2005.

[9] K. Mohanram and N. A. Touba. Cost-effective approach for reducing soft error failure rate in logic circuits. In *Proceedings of the International Test Conference (ITC)*, pages 893–901, October 2003.

[10] B. S. Gill, C. Papachristou, F. G. Wolff, and N. Seifert. Node sensitivity analysis for soft errors in cmos logic. In *Proceedings of the International Test Conference (ITC)*, November 2005.

[11] A. Maheshwari, I. Koren, and W. Burleson. Techniques for transient fault sensitivity analysis and reduction in vlsi circuits. In *Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, pages 597–604, November 2003.

[12] R.R. Rao, K. Chopra, D.T. Blaauw, and D.M. Sylvester. Computing the soft error rate of a combinational logic circuit using parameterized descriptors. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 26(3):468–479, March 2007.

[13] M. Sonza Reorda and M. Violante. Accurate and efficient analysis of single event transients in vlsi circuits. In *Proceedings of 9th IEEE International On-Line Testing Symposium (IOLTS)*, July 2003.

[14] A. Sanyal, K. Ganeshpure, and S. Kundu. An improved soft-error rate measurement technique. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(4):596–600, April 2009.

[15] M. Zhang and N. R. Shanbhag. A soft error rate analysis (sera) methodology. In *Proceedings of the IEEE International Conference on Computer Aided Design (ICCAD)*, pages 111–118, November 2004.

[16] Q. Zhou and K. Mohanram. Cost-effective radiation hardening technique for combinational logic. In *Proceedings of the IEEE International Conference on Computer Aided Design (ICCAD)*, pages 100–106, November 2004.

[17] G. Asadi and M. B. Tahoori. An accurate ser estimation method based on propagation probability. In *Proceedings of the IEEE/ACM International Conference on Design, Automation and Test in Europe (DATE)*, pages 306–307, March 2005.

[18] G. Asadi and M. B. Tahoori. An analytical approach for soft error rate estimation in digital circuits. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 3, pages 2991–2994, May 2005.

[19] K. P. Parker and E. J. McCluskey. Probabilistic treatment of general combinational networks. *IEEE Transactions on Computers*, c24(6):668–670, June 1975.

[20] H. Asadi, M. B. Tahoori, and C. Tirumurti. Estimating error propagation probabilities with bounded variances. In *Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, pages 41–49, September 2007.

[21] A. Dharchoudhury, S.M. Kang, H. Cha, and J. H. Patel. Fast timing simulation of transient fault in digital circuits. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD)*, pages 719–722, 1994.

[22] G. C. Messenger. Collection of charge on junction nodes from ion tracks. *IEEE Transactions on Nuclear Science*, 29:2024–2031, December 1982.

[23] Q. Zhou and K. Mohanram. Transistor sizing for radiation hardening. In *Proceedings of the International Reliability Physics Symposium (IRPS)*, pages 310–315, 2004.

[24] B.S. Gill, C. Papachristou, and F.G. Wolff. Soft delay error effects in cmos combinational circuits. In *Proceedings of the 22nd IEEE VLSI Test Symposium (VTS)*, 2004.

[25] H.T. Nguyen and Y. Yagil. A systematic approach to ser estimation and solutions. In *Proceedings of the 41st Annual International Reliability Physical Symposium (IRPS)*, pages 60–70, 2003.

[26] R. Rajaraman, J.S. Kim, N. Vijaykrishnan, Y. Xie, and M.J. Irwin. Seat-la: A soft error analysis tool for combinational logic. In *Proceedings of the 19th International Conference on VLSI Design*, January 2006.

[27] T. Nakamura, T. Nunomiya, S. Abe, K. Terunuma, and H. Suzuki. Sequential measurements of cosmic-ray neutron spectrum and dose rate at sea level in sendai, japan. *Journal of Nuclear Science and Technology*, 42(10):843–853, October 2005.

[28] S. Onoda, T. Hirao, J.S. Laird, K. Mishima, and K. Kawano H. Itoh. Transient currents generated by heavy ions with hundreds of mev. *IEEE Transactions on Nuclear Science*, 53(6):3731–3737, December 2006.

[29] B. Zhang and M. Orshansky. Symbolic simulation of the propagation and filtering of transient faulty pulses. In *Proceedings of the first Workshop on Silicon Errors in Logic - System Effects*, April 2005.

[30] B. Zhang, W.S. Wang, and M. Orshansky. Faser: Fast analysis of soft error susceptibility for cell-based designs. In *Proceedings of the 7th International Symposium on Quality Electronic Design (ISQED)*, March 2006.

[31] R. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, 35:677–691, 1986.

[32] N. Seifert and N. Tam. Timing vulnerability factors of sequentials. *IEEE Transactions on Device and Materials Reliability*, 4(3):516 – 522, September 2004.

[33] Y.S. Dhillon, A.U. Diril, and A. AChatterjee. Soft-error tolerance analysis and optimization of nanometer circuits. In *Proceedings of the IEEE/ACM International Conference on Design, Automation and Test in Europe (DATE)*, pages 288 – 293, 2005.

[34] S. Krishnaswamy, G.F. Viamontes, I.L. Markov, and J.P. Hayes. Accurate reliability evaluation and enhancement via probabilistic transfer matrices. In *Proceedings of the IEEE/ACM International Conference on Design, Automation and Test in Europe (DATE)*, pages 282–287, 2005.

[35] S. Weiguang, X. Liyi, and M. Zhigang. Fast soft error rate computing technique based on state probability propagating. In *4th IEEE Conference on Industrial Electronics and Applications*, pages 734–738, May 2009.

[36] S. Weiguang, X. Liyi, and M. Zhigang. A novel soft error sensitivity characterization technique based on simulated fault injection and constrained association analysis. In *15th IEEE International Conference on Electronics, Circuits and Systems*, pages 766–769, September 2008.

[37] A. Ejlali, S.G. Miremadi, H. Zarandi, G. Asadi, and S.B. Sarmadi. A hybrid fault injection approach based on simulation and emulation co-operation. In *Proceedings of the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 479–488, June 2003.

[38] K. Mohanram and N. A. Touba. Partial error masking to reduce soft error rate in logic circuits. In *Proceedings of the 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT03)*, pages 433–440, November 2003.

[39] K. Mohanram, E.S. Sogomonyan, M. Gossel, and N. A. Touba. Synthesis of low-cost parity-based partially self-checking circuits. In *Proceedings of the IEEE On-Line Testing Symposium (IOLTS)*, pages 35–40, July 2003.

[40] S. Krishnamohan and N.R. Mahapatra. A highly-efficient technique for reducing soft errors in static cmos circuits. In *IEEE International Conference on Computer Design*, pages 126–131, 2004.

[41] S. Mitra, N. Seifert, M. Zhang, Q. Shi, and K. Kim. Robust system design with built-in soft-error resilience. *IEEE Transactions on Computers*, 38:43–52, February 2005.

[42] S. Mitra, M. Zhang, T. Mak, N. Seifert, V. Zia, and K.S. Kim. Logic soft errors a major barrier to robust platform design. In *Proceedings of the International Test Conference (ITC)*, 2005.

[43] S. Mitra, M. Zhang, S. Waqas, N. Seifert, B. Gill, and K. S. Kim. Combinational logic soft error correction. In *Proceedings of the International Test Conference (ITC)*, pages 1–9, 2006.

[44] M. Nicolaidis. Time redundancy based soft-error tolerant circuits to rescue very deep submicron. In *17th IEEE International VLSI Test Symposium (VTS)*, pages 86–94, April 1999.

[45] M. Nicolaidis. Design for soft error mitigation. *IEEE Transactions on Device and Materials Reliability*, 5(3):405–418, September 2005.

[46] R.R. Rao, D. Blaauw, and D. Sylvester. Soft error reduction in combinational logic using gate resizing and flip-flop selection. In *Proceedings of the IEEE/ACM International Conference on Computer Aided Design (ICCAD)*, November 2006.

[47] V. Degalahal, R. Ramanarayanan, N. Vijaykrishnan, Y. Xie, and M. J. Irwin. The effect of threshold voltages on the soft error rate memory and logic circuits. In *Proceedings of 5th International Symposium on Quality Electronic Design*, pages 503–508, 2004.

[48] J. Kumar and M. B. Tahoori. A low power soft error suppression technique for dynamic logic. In *Proceedings of the IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT)*, 2005.

[49] R. Kodavarti and D. E. Ross. Signal probability calculation using partial functional manipulation. In *Proceedings of the IEEE VLSI Test Symposium (VTS) [Digest of Papers]*, 1993.

[50] A. Majumdar and S. B. K. Vrudhula. Analysis of signal probability in logic circuits using stochastic models. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 1(3):365–379, September 1993.

[51] J. Savir G. S. Ditlow and P. H. Bardell. Random pattern testability. *IEEE Transactions on Computers*, c33(1):79–90, January 1984.

[52] T. H. Cormen, C. L. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press & McGraw-Hill, 2nd edition, 2001.

[53] C. F. Dietrich. *Uncertainty, Calibration and Probability*. Adam Hilger Pub., 1991.

[54] V. Lindberg. Uncertainties and error propagation. http://www.rit.edu/ uphysics/uncertainties/Uncertaintiespart1.html, July 2000.

[55] F. Wang and Y. Xie. An accurate and efficient model of electrical masking effect for soft errors in combinatorial logic. In *Proceedings of the 2nd Workshop on Silicon Errors in Logic - System Effects*, April 2006.

[56] C. Zhao and S. Deys. Improving transient error tolerance of digital vlsi circuits using robustness compiler (roco). In *Proceedings of the 7th International Symposium on Quality Electronic Design (ISQED)*, March 2006.

[57] H. Asadi and M. B. Tahoori. Soft error hardening for logic-level designs. In *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2006.

[58] R.C. Baumann. Soft errors in advanced semiconductor devices-part i: the three radiation sources. *IEEE Transactions on Device and Materials Reliability*, 1(1):17–22, March 2001.