Soft Error Reduction in Combinational Logic Using Gate Resizing and Flipflop Selection

Rajeev R. Rao, David Blaauw, Dennis Sylvester
Department of EECS, University of Michigan, Ann Arbor, MI 48109
{rrrao, blaauw, dennis}@eecs.umich.edu

Abstract

Soft errors in logic are emerging as a significant reliability problem for VLSI designs. This paper presents novel circuit optimization techniques to mitigate soft error rates (SER) of combinational logic circuits. First, we propose a gate sizing algorithm that trades off SER reduction and area overhead. This approach first computes bounds on the maximum achievable SER reduction by resizing a gate. This bound is then used to prune the circuit graph, arriving at a smaller set of candidate gates on which we perform incremental sensitivity computations to determine the gates that are the largest contributors to circuit SER. Second, we propose a flipflop selection method that uses slack information at each primary output node to determine the flipflop configuration that produces maximum SER savings. This approach uses an enhanced flipflop library that contains flipflops of varying temporal masking ability. Third, we propose a unified, co-optimization approach combining flipflop selection with the gate sizing algorithm. The joint optimization algorithm produces larger SER reductions while incurring smaller circuit overhead than either technique taken in isolation. Experimental results on a variety of benchmarks show SER reductions of 7.9X with gate sizing, 6.6X with flipflop assignment, and 28.2X for the combined optimization approach, with no delay penalties and area overheads within 5-6%. The runtimes for the optimization algorithms are on the order of 1-3 minutes.

1 Introduction

Energetic cosmic particles interact with the silicon substrate in integrated circuits to produce transient noise events. A radiation particle strike on an SRAM cell or a memory register that can cause a bit flip is called a *single event upset* (SEU). Similarly, a particle strike on a logic gate in a combinational circuit can produce a voltage glitch referred to as a *single event transient* (SET). An SET can potentially propagate to an output node and cause an erroneous signal to be latched into a flipflop. These types of radiation induced faults are called soft errors and their frequency is referred to as the *soft error rate* (SER). The quantitative metric used to measure SER is *failures-in-time* (FIT), corresponding to the number of errors in one billion device hours.

Continued technology scaling has resulted in the emergence of soft errors as one of the major reliability challenges for current and future digital VLSI designs. The failure rate due to soft errors is expected to exceed the failure rate due to all other reliability mechanisms (such as gate oxide breakdown, electromigration, etc.) combined [1]. Several works have studied the impact of soft errors on the various components of a typical IC [1][2][3]. A simultaneous reduction in both the critical charge and collection efficiency has resulted in relatively constant SRAM SER over several technology generations. In addition, error correction codes enable a high level of soft error protection for memories. Similarly, industrial estimates project that the nominal SER of latches is nearly constant from 130nm to 65nm technologies [4]. The use of radiation hardened latches [5] further immunizes latches from particle strikes. In contrast, SER due to particle hits on combinational logic is predicted to increase rapidly and a recent estimate [2] shows that SETs in logic will significantly influence chip SER at the 45nm node. In large-scale applications such as server farms and communications systems, logic soft errors are predicted to be significant contributors to system-level silent data corruption events [6]. It is, therefore, critical to develop analysis and mitigation techniques to combat the effects of soft errors on logic.

Combinational logic circuits can be immunized against the effects of soft errors using two methods. First, the probability of a transient glitch occurring at any sensitive node in the circuit can be minimized. This approach targets the soft error problem at the source by lowering the probability of an erroneous SET pulse from being generated. Selectively hardening the set of susceptible gates can result in the absence of most faulty pulses in the circuit. Second, the probability of an SET being latched into the flipflop can be minimized. This approach targets the soft error problem at the sink because, although it permits SETs to originate at any node inside the logic, it disallows such erroneous glitches from being registered by the sequential element. By carefully designing a flipflop to filter a large fraction of the SETs incident at its data port, it is possible to completely suppress a soft error occurring in logic to permeate to the architectural or the system level. Naturally, the selection of one approach over the other is dictated by the amount of overheads that they introduce. Directly modifying the gates inside a circuit incurs, in general, large overheads in power, delay and area that can prohibit design convergence. Conversely, modifying only the flipflop elements present on the boundary of a logic circuit incurs small cost in terms of power and area but can vastly influence the timing characteristics of the overall design and also place additional constraints on the clock tree network. Hence, it is necessary to consider these gate-based and flipflop-based SER mitigation approaches separately as well as in unison, along with their associated overheads while optimizing logic circuits for better SER immunity.

This paper proposes novel circuit level optimization techniques to minimize SER of combinational logic circuits. First, we present a new gate resizing algorithm that uses accurate sensitivity measurements to guide the optimizer. This approach first prunes the entire circuit to a smaller subset of gates by efficiently computing bounds on the SER reduction achievable by modifying a gate. We then use

ICCAD'06, November 5-9, 2006, San Jose, CA

this subset of gates as possible candidates for resizing and identify gates that provide the maximum SER improvement while incurring the least amount of area overhead. Second, we propose the use of an enhanced library of flipflop variants that trades off increased amounts of pulse filtering (and, hence, reduced SET latching susceptibility) with larger amounts of delay overhead. We present a slack-based optimization method where output flipflops are selected from this library based on the slack available at the node. Third, we present a joint optimization algorithm that performs simultaneous gate resizing and slack-based flipflop assignment. This combined approach produces a near ideal design point by providing significant SER reduction while modifying the original circuit in a minimal fashion. The three techniques incur *zero delay overhead* and instead trade-off small amounts of increase in circuit area for SER reduction.

Each proposed optimization technique is exercised on a wide variety of benchmark circuits. Results show that for circuits synthesized with tight delay constraints, we achieve SER reductions of 19.7X while increasing area by 0.4% on average. For circuits synthesized with loose delay constraints, we achieve larger SER reductions of 28.2X while incurring area overhead of up to 3.2% on average.

The paper is organized as follows. Section 2 discusses previous work targeted towards logic SER reduction. Section 3 describes how gate sizing and temporal masking in flipflops can reduce SER. Section 4 provides a detailed description of the proposed algorithms. In Section 5, we present results and conclude in Section 6.

2 Prior Work

Early techniques proposed for circuit level radiation hardening are based on classical fault tolerance techniques such as triple modular redundancy [7]. A more cost-effective approach, proposed in [8], duplicates only a portion of the circuit to achieve the target fault coverage. Node-specific optimization methods that propose to use techniques such as transistor sizing [9][10][11] and gate cloning [12] to alter some aspects of the gate structure in the circuits to make them more resilient fall under the class of gate-based SER mitigation methods described in the previous section. On the other hand, flipflop directed optimization approaches include the dual-sampling latch [13], flipflops with delayed data/clock signal sampling [14], dual-ported latches [15], flipflops redesigned for SET filtering [16][17], latches with additional keepers [5] and scan flipflop based designs [18][4].

A large number of these techniques rely on the replicate/recompute design methodology by using time/space redundancy. However, the usage of checkers and logic duplication inherently introduces significant delay, area and power overhead. The partial duplication method in [8] incurs an area overhead of about 85%. Gate-specific SER mitigation techniques operate on a small set of susceptible nodes chosen using a circuit-specific criterion (such as gates with maximal fanout count). However, optimizing gates without accounting for the overheads they introduce produces ambiguous estimates for the amount of SER reduction and also incurs significant power/area penalties. [11] proposes the concept of sizing up the output load in conjunction with multi-V_{dd}/V_{th} circuit design. This method incurs a fixed amount of delay/power overhead and can also worsen the overall circuit reliability due to the presence of non-robust cells. The concept of gate cloning [12] attempts to redistribute soft error susceptibility by locally splitting a multiple input gate; however, increasing the number of vulnerable nodes increases the number of particle strike locations thereby impacting the total circuit SER. The methods in [10][11][18] consume over 25% area overhead and [11] also reports a delay increase of 6.2%. Flipflop directed approaches, on the other hand, incur significant delay overhead because they impact all the paths located in the fanin cone associated with the output node.

In this paper, we present a methodology that optimizes the gates and flipflops simultaneously. The key contribution of our work is this ability to conjoin gate modification with appropriate flipflop selection to achieve maximum SER reduction while accruing small increases in area and power and zero delay overhead.

3 SER Analysis Preliminaries

This section discusses the mechanisms by which gate sizes and temporal masking impact circuit SER. We describe the efficacy of employing gate resizing and specially designed flipflops to minimize the circuit SER value. We then provide an overview of the underlying SER computation algorithm.

3.1 Impact of gate sizing on SER

The amount of charge generated at a susceptible node in any gate due to a neutron strike is a strong function of its drain area. By sizing up a gate, the effective capacitance of the device is increased thereby making it less likely that the injected transient current will cause a voltage glitch of sufficient magnitude. For instance, consider a single inverter with a fixed output load. Replacing an INVX1 with another inverter INVX4 (with 4X more drive strength) decreases glitch amplitude significantly (see circled waveforms in Figure 1). As a result, upsizing a gate always decreases the probability of a soft error occurring due to direct particle hits. On the other hand, an upsized device has significantly higher drive strength which allows for better propagation of the input transients at a gate. This is particularly true in cases where the output load of the cell is large. Figure 1 qualitatively shows the two types of input transients at a gate: 1) Non-linear waveform shapes that can possibly occur due to a strike on the immediately preceding gate and 2) standard trapezoidal shapes that occur when an injected transient propagates through a few logic stages. In this plot, the INVX1 completely filters the short, non-linear waveform while allowing the trapezoidal shape to propagate with little or no attenuation. On the other hand, the INVX4 allows the propagation of both types of transients and in fact, produces a slight boost in the signal strength of the non-linear transient. Transient waveforms with small pulse widths typically correspond to

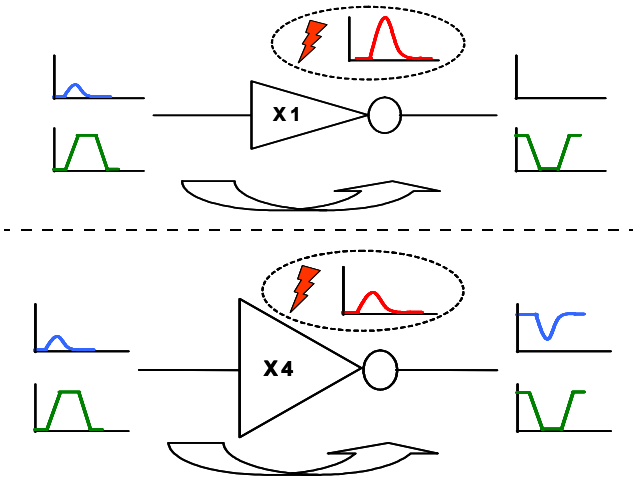

Figure 1. Qualitative comparison (in terms of electrical properties) between INVX1 and INVX4. Top (Circled) = Injected waveforms and Sides = Propagated waveforms.

particle hits that inject a small amount of charge but have a larger error rate probability associated with them. Since upsized gates have a higher propensity to propagate these short transients, it is possible that increasing gate sizes unilaterally can worsen circuit SER.

In a sensitivity-based timing optimization algorithm (such as TILOS [19]), gate sizes are incrementally increased in small steps to determine the size that provides the best delay value. From the previous discussion, we make the key observation in our work that gate sizes can be either increased or decreased to achieve SER reductions. For each gate, it is important to consider the relative significance of the injected and propagated waveforms to the total SER value. This approach is in contrast to [10] which considers only the impact of first strike waveforms and [20] which only targets the waveforms that are propagated. Further, [10] considers only the worst case injection charge value of 150fC in the analysis, thereby disregarding the vast majority of strikes that inject charge lower than 150fC but contribute a much greater fraction to total SER events. Considering both injected and propagated waveforms at a gate, across the entire spectrum of neutron strikes, provides a more accurate and realistic assessment of the impact of an individual gate on the total circuit SER. Hence, the proposed algorithms in our work consider gate resizing (upsizing and downsizing) to achieve SER improvement.

In our analysis, we assume that the baseline (unoptimized) circuits are synthesized based on a prescribed set of delay/power constraints. Thus, depending on the available resources, each gate is chosen from a set of sizes so that both upsizing and downsizing can be performed on them.

3.2 Impact of temporal masking on SER

Temporal masking is the mechanism that determines whether a transient arriving at a flipflop input is latched as an erroneous value. A flipflop is susceptible to capturing a spurious pulse if the transient occurs inside its latching (T_{setup} + T_{hold} time) window. For a given waveform *k*, *z(k)* is defined as the temporal probability that *k* causes a faulty bit to be registered.

Cosmic particles that strike the logic contain a finite amount of energy. For the 0.13µm technology, [10] notes that the energy levels of neutron strikes can be mapped to deposited charge values in the range [10fC, 150fC]. As a result, the pulse widths of transient glitches also occur for a finite duration in a characterizable range. [23] reports a range of [78ps, 206ps] for a 0.13µm cell library. This observation of a finite duration for the pulses leads to the possibility of designing flipflops that filter transients based on the pulse widths. If the master latch in the flipflop is sufficiently slowed down, the filtering window is widened so that a subset of the transients are disallowed from being registered by the flipflop. This effect is specifically targeted towards the fast (short pulse width) transient waveforms. Since fast transients typically correspond to soft errors with high strike rate probabilities, preventing these SETs from latching enables a significant reduction in the circuit SER.

In [16], the authors propose the addition of extra resistors at the input stage of the latch to filter transients from appearing in the latching window. However, in addition to the large (about 300%) power/delay costs associated with this method, the usage of passive elements is impractical in current digital designs. In contrast, [17] proposes the use of transistor sizing to perform the aforementioned filtering operation. By resizing the forward inverter in the cross-coupled inverter pair that constitutes the master latch, the filtering window is sufficiently increased so that SETs with short pulse widths are disallowed from being latched by the flipflop. It was observed that the temporal probability *z(k)* for any waveform is strongly correlated to its pulse width T_{pw} and the $(T_{setup} + T_{hold})$ time window.

Table 1. Delay/area overheads for the flipflop variants. A single FO4 delay = 40.1ps

Flipflop Variant	T_{pw} Filtering Threshold (in ps)	Overhead		
		Delay (ps)	Delay (xFO4)	Area (%)
Lib	27	0	0	0
F100	100	62.4	1.6	< 0.1
F130	130	92.4	2.3	0.1
F160	160	122.5	3.1	0.1
F210	210	153.4	3.8	0.2

A side effect of this sizing operation is that it increases the setup and hold times considerably thereby introducing additional delay overhead. However, a small amount of the performance degradation can be recovered by sizing up the drivers connected to the input and output port. Table 1 summarizes the total overheads associated with the construction of these redesigned flipflops from [17]. Beginning with a library flipflop, the devices are progressively sized to obtain various filtering threshold values. For instance, the F130 flipflop filters all transient pulses with width $T_{pw} \leq$ 130ps. The Lib flipflop does not filter any pulses because its T_{pw} threshold of 27ps is much lower than the minimum SET pulse width of 78ps. The F210 flipflop can potentially eliminate all possible transient pulses from latching into the flipflop since the maximum transient pulse width for the cell library is 206ps. Note, however, that this filtering operation is valid only when the flipflop data bit is not switching from the previous cycle. For the case of switching input data, the temporal probability is independent of T_{pw} and only depends on the location of the pulse in the overall time interval. As a result, the flipflop variants are ineffective in handling these types of error events.

While employing flipflop directed SER optimization approaches, it is crucial to assess their impact on the performance of the circuit. A simplistic method to use this FF library for SER mitigation is to replace each library flipflop in a logic circuit with one of the new variants. However, this would impose a flat, delay overhead of at least 62.4ps which is not a viable option for most performance sensitive designs. A more effective method is to use the slack available at each output node and assign flipflops appropriately. In the subsequent sections, we present an exact formulation of this flipflop assignment problem. The work proposed in this paper aims to provide automated methods by which these SET tolerant flipflops can be inserted in a logic circuit.

3.3 SER Analysis Engine

Before we describe the SER optimization techniques, we briefly discuss the underlying SER estimation methodology used in our analysis. Recently, a number of logic soft error analysis algorithms have been presented; these include SERA [21], ASERTA [9], SEATLA [22], [23] and FASER [24]. These tools employ a variety of techniques such as circuit simulation, probability theory and binary decision diagrams to compute the logic SER. For the analysis presented in this paper we chose to use the tool in [23] for the following reasons: (1) It provides a quick and efficient method for SER computation. As we observe in Section 4.1.2, short runtime for the estimation engine is vital to perform fast incremental SER calculations. (2) Unlike the other tools, it considers the entire spectrum of neutron strikes (all charge values in the [10fC, 150fC] range) during SER computation. The strike probabilities associated with the individual

charge values varies greatly (by about four orders of magnitude). We therefore believe that, from an optimization perspective, it is important to consider the full range of charge values, instead of just 4-5 discrete values.

The authors in [23] model the transient glitch due to a neutron strike using the current pulse model presented in [25].

$$I(t) = \frac{2Q_0}{\tau\sqrt{\pi}}\sqrt{\frac{t}{\tau}}\exp\left(\frac{-t}{\tau}\right) \quad \text{(EQ 1)}$$

Here Q_0 is the amount of injected charge, τ is time-dependent pulse shaping parameter and $I(t)$ is the current. Empirical models from [26] are then used to map the deposited charge Q_0 with a strike rate value.

$$R = F \times K \times A \times \left(\frac{1}{Q_s}\right) \times \exp\left(\frac{-Q_0}{Q_s}\right) \quad \text{(EQ 2)}$$

Here R = rate of SET strikes, F = neutron flux with energy>10MeV, A = area of the circuit susceptible to neutron strikes (in cm^2), K = a technology independent fitting parameter, Q_0 = charge generated by the particle strike and Q_s = charge collection slope. A parametric descriptor object correlates these strike rate values with a corresponding transient waveform. The logic level SER analysis model consists of the injection and propagation of these descriptors through a circuit. The tool accounts for all the three types of masking mechanisms - logical, electrical and temporal - during the estimation flow. We refer the reader to [23] for further details about this tool.

4 SER Optimization Techniques

This section explains the three SER optimization techniques presented in this paper. We first discuss various aspects of the sensitivity-based gate sizing algorithm, including the methods used for gate-specific SER bound calculation and candidate set selection through circuit pruning. We then present the slack-based FF assignment method that uses the FF variant library to achieve significant SER savings. Third, we present a joint approach combining flipflop (FF) assignment with sizing to provide the best circuit solutions in terms of circuit SER.

4.1 Sensitivity-Based Gate Sizing Algorithm

A large variety of circuit optimization algorithms in VLSI CAD use sensitivity-driven engines to guide the optimizer towards the best solution. Figure 2 presents pseudo-code for the proposed sensitivity-based gate sizing algorithm for SER minimization. We begin by developing an efficient bounding technique to prune the circuit graph and produce a candidate set of gates C consisting of cells that can potentially be resized for maximum SER improvement. We then

```
GATE RESIZING
C = candidate set of gates
while (constraints NOT violated)
    for each gate g ∈ C
        Resize gate g
        Recompute ckt_area
        /* Traverse fanout cone of g */
        /* Visit each output node affected by this change */
        Recompute ckt_delay, ckt_SER
        Calculate sensitivity from EQ5
    Pick the gate with the best sensitivity
    Make a "move" by resizing this gate appropriately
    Repeat resizing operation
```

Figure 2. Pseudo-code for the proposed algorithm for gate resizing

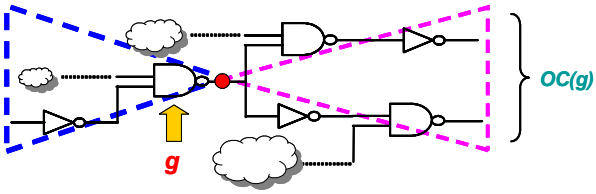

Figure 3. Fanin and fanout cones associated with a gate *g* and the definition for *OC(g)*, the output count of *g*

define a sensitivity metric to maximize SER gains while limiting area overhead. Efficient sensitivity calculations are a crucial aspect of any circuit optimization algorithm. In our approach, we pick a single gate with the best sensitivity value and make the appropriate sizing move on this gate. Note, however, that it is also possible to use the sensitivity information of all gates in a more complex non-linear optimizer that performs multiple, simultaneous gate sizing moves to achieve the optimal SER value.

4.1.1 Candidate gate selection

The selection of gates for the candidate set C significantly influences the performance of the proposed approach. In a non-ideal case, each gate in the circuit must be considered as a potential candidate for resizing. However, by identifying certain important characteristics related to the optimization metric we efficiently compute bounds on the SER value allowing for a subset of gates to be inserted into C. The SER bounds computation ensures that the circuit graph is pruned sufficiently to keep C relatively small.

The contribution of an individual cell to the total circuit SER is determined by various factors such as cell size, cell output load, input state probabilities, size of fanin/fanout cones, and depth from the output nodes. Since logic gates across a circuit vary significantly in these parameters, the relative contribution of individual gates to the total circuit SER can vary by as much as three orders of magnitude. This point shows that only a small fraction of the gates affect the circuit SER significantly. Therefore, the candidate set needs to be chosen carefully such that performing resizing on only this smaller set of gates provides the maximum amount of SER improvement.

To perform this selection, we first define new parameters *OC(g)*, *SER(g)* and *RedRatio(g)* for each gate g as follows: Each gate has fanin and fanout cones associated with it. As illustrated in Figure 3, *OC(g)* counts the number of outputs to which g is connected to in its fanout cone. Every gate g contains the set of descriptors due to all SETs that originate in the fanin cone of g and a single SET descriptor due to a strike on g itself. Suppose we disconnect the entire fanout cone of g and treat g as an output node (see Figure 4). *SER(g)* corresponds to this case when g is connected directly to a flipflop. In the actual circuit, as the transient waveforms propagate in the fanout cone of g, *SER(g)* can only be reduced due to logical and electrical masking mechanisms. For instance, consider a single path from g to an output node that is b levels away from g. Let p_i for $i = [1,b]$ be

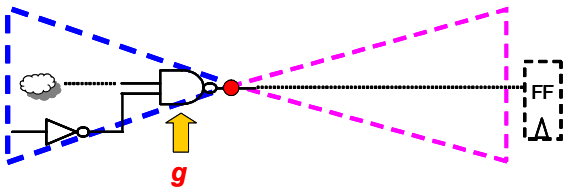

Figure 4. Calculation of *SER(g)*. Fanout cone of *g* is disconnected and *g* is assumed to be directly connected to an output FF

the logical probabilities associated with each gate in this path. The SER value due to SETs propagating through this path will be $\left(\prod_{i=1}^{b} p_i\right) SER(g)$. In this expression, since each $p_i \leq 1$, we obtain the following inequality.

$$\left(\prod_{i=1}^{b} p_i\right) SER(g) \leq SER(g) \quad \text{(EQ 3)}$$

SER(g) therefore represents an approximate upper bound on the SER contribution of *g* at a single output node in the fanout cone of *g*. Note that this relation is independent of the correlation characteristics of the logical probabilities along the path.

Since gate *g* can affect several output nodes in its fanout cone we calculate *(SER(g)*OC(g))* and see that this product is an upper bound on the relative contribution of the fanin cone of *g* to the total circuit SER. Given the total circuit SER (*TotalCktSER*), we then define *RedRatio(g)* as

$$RedRatio(g) = \frac{(SER(g) \times OC(g))}{TotalCktSER} \quad \text{(EQ 4)}$$

Any subsequent sizing operation on gate *g* will, at best, completely eliminate the SER contribution of *g* and its entire fanin cone and reduce the total circuit SER by at most *(SER(g)*OC(g))*. In this formulation we ignore the effects of reconvergence on the SER; however this can be included in the analysis by performing an initial pass on the fanout cone to determine the exact amount of magnification that reconvergent fanouts cause to *SER(g)*.

Next, we specify a minimum reduction ratio (*mrr*) value in order to prune gates and construct the candidate gate set. For each gate *g*, we add *g* into *C* only if *RedRatio(g) > mrr*. For instance, with *mrr* = 1%, we do not add any gates into *C* that will, at best, give SER improvement of <1%. All gates in *C* are not guaranteed to give an improvement of at least 1%; Instead the 1% figure represents the minimum *potential* gains and not the actual gains in SER. Since SER values vary dramatically across the gates in a circuit, this pruning operation is very efficient in removing all gates that produce little or no improvement on the circuit SER. For the gates that are added to the candidate gate set, we perform sensitivity computations as explained in the next sub-section. In practice, we find that using *mrr* = 1% prunes out a large fraction of the gates and only 10-20% of gates are typically considered for sizing.

4.1.2 Structure of the Algorithm

In our analysis, we consider three major circuit parameters - delay, SER and area - as the variables during sizing. For each cell, we first extract delay arcs from a standard timing library and define circuit delay as the maximum of the arrival times across all output nodes. We define cell area as the sum of device widths of all transistors in the gate and circuit area as the sum of areas of all cells. In our work, we focus only on the overhead aspect when resizing any given gate. While this definition of area is simplistic, we believe that it efficiently characterizes the overheads introduced during a sizing operation. Further, the device widths of the transistors are directly related to the total effective capacitance and hence, the total power dissipated by the cell. Thus, this definition of circuit area correlates fairly accurately with the total power consumed by the circuit.

The algorithm proceeds by picking each gate $g \in C$ in turn, perturbing the circuit by resizing this gate, and then recomputing the circuit delay, area and SER for this perturbed circuit. An important requirement for any sensitivity-based algorithm is the ability to perform incremental recomputation. In other words, by perturbing only a small portion of the circuit, we must not be required to perform a complete recomputation over the entire circuit. The change in circuit area is easy to quantify since local changes in cell area are reflected globally as well. For delay and SER, in our approach when any gate *g* is resized, we only consider the fanout cone of *g* while recomputing these parameters. Due to the modified size of *g*, the output capacitance seen by the immediate fanins of *g* is affected so that both delay and SER of *g* are altered.

To recalculate the new circuit delay and SER, we need to propagate the new arrival times and SET descriptors along the fanout cone until we reach an output node. However, during delay recalculation we frequently observe that after a few propagations, we encounter a path with greater arrival time so that further propagation along the cone for the new arrival time is unnecessary. This occurs because, in general, a vast majority of the gates are not critical and have no impact on circuit delay. Similarly, when propagating SET descriptors further along the circuit, a complete recalculation over the entire fanout cone is not required and propagation for at most 4-5 stages is sufficient. To detect cases of zero SER change due to a perturbation, we check that both the waveform shape and SER value of the descriptors are identical since both these factors impact circuit SER.

4.1.3 Sensitivity Measurement

After circuit parameters are recalculated, we perform a sensitivity measurement to determine the relative merits of each sizing move. First, we disregard all moves that worsen circuit performance and only consider cases where the circuit delay is equal to (or less than) the initial value. Next, since we seek to minimize area overhead while maximizing SER improvement we define the sensitivity as follows:

$$Sensitivity = \frac{\Delta SER}{\Delta Area} = \frac{SER_{original} - SER_{perturbed}}{Area_{perturbed} - Area_{original}} \quad \text{(EQ 5)}$$

We only consider cases where SER improves ($\Delta SER > 0$) and prioritize cases where gates are downsized ($\Delta Area < 0$) over those involving upsizing ($\Delta Area > 0$). As a delay constraint, we limit the total circuit delay to the initial delay point of the circuit. Thus, gates on critical paths are resized for SER improvement only if they also result in a delay improvement. We also impose an area constraint to avoid instances where circuit area increases significantly for marginal gains in SER.

4.1.4 Algorithm Complexity

The candidate gate selection mechanism significantly prunes the circuit and typically produces a subset containing at most 10-20% of the gates. The incremental recomputation method for delay and SER decreases the runtime further by eliminating the need for full recalculation over the entire circuit graph. In the worst case, the runtime per iteration for an *n*-gate circuit can still be $O(n^2)$; however, in practice, we find that it is significantly better than this bound. The total runtime for the algorithm depends mainly on the number of gates *j* that are resized, and is not directly influenced by the size of the circuit. Further, we impose additional constraints on the area and delay of the circuit so that the number of sizing moves is limited, making *j* a small fraction of the total circuit size. The inclusion of such stopping criteria also ensures that the algorithm converges. The worst case complexity of the entire algorithm is given by $O(jn^2)$. Runtimes shown in the results section indicate that even for the largest circuit with ~5000 gates, the total runtime is at most two hundred seconds.

4.2 Slack-based FF Assignment

The new flipflop variants provide an effective option for circuit SER optimization since they do not modify the logic circuit, instead focusing on filtering the faulty transients from being latched. Each variant incurs a certain amount of delay overhead such that FFs with better SER filtering incur larger overhead. In a standard logic circuit, each output node is connected to a standard library flipflop. By examining the slack available at each output node and assigning FF variants appropriately we can potentially reduce SER significantly.

The mathematical formulation of the slack-based FF assignment can be stated as follows: Each output node m is associated with an arrival time value of $AT(m)$. The circuit delay is set by the output node with the maximum value of AT so that:

$$Delay = max\{AT(m)\} \qquad m = [1, NumOutputs] \qquad \text{(EQ 6)}$$

The slack available at each output node is the difference between the delay of the circuit and the arrival time at that node.

$$Slack(m) = Delay - AT(m) \qquad \text{(EQ 7)}$$

Depending on the value of slack, one of the flipflop variants from Table 1 can now be assigned to each output node. For instance, for $0ps \le Slack_m < 62.4ps$, the Lib FF is assigned, while for $62.4ps \le Slack_m < 92.4ps$ the F100 FF is assigned, and so on. In each case, the sum of arrival time at the output $AT(m)$ and the overhead of the flipflop variant is always lower than the initial specified value of $Delay$ (EQ6). Thus, the worst case delay of the circuit is unchanged.

This type of flipflop assignment is best suited to circuits containing several outputs with significant slack. Given a circuit with a small number of critical paths all leading to a single output node, it is possible to assign all other output nodes to one of the flipflop variants and achieve significant SER reduction. Note that the runtime for this reassignment is negligible (compared to gate resizing) since it only requires a single pass through the output nodes of the circuit.

4.3 Combined FF Assignment + Gate Sizing

The combined optimization approach uses the electrical masking advantages of gate sizing and the temporal masking properties of the redesigned flipflops to achieve large SER reductions. In the co-optimization approach, three factors help reduce the total circuit SER. The characteristics of the slack-based FF assignment and simple gate sizing have been described previously in this paper. In addition, gate sizing may also create slack at an output leading to a better choice for the flipflop variant.

We illustrate this effect using the example shown in Figure 5. Suppose a flipflop contains multiple short paths and a single long path in its fanin cone. The pulse width ranges for the transient glitches corresponding to these paths are shown in the plot. First, note that simple flipflop selection as presented in Section 4.2 will not be possible because the presence of the long path imposes only a small amount of slack at the output. Second, although gate sizing (Section 4.1) is possible, it may not produce vast reductions in the

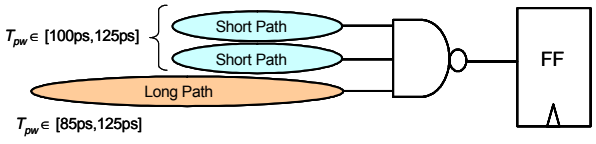

Figure 5. Multiple short paths and a single long path connected to the output node. Joint optimization enables significant SER reduction for this case

COMBINED FF ASSIGNMENT AND GATE RESIZING

```
/* Assign FFs initially based on available slack */
C = candidate set of gates
while (constraints NOT violated)
    for each gate g ∈ C
        Resize gate g
        Recompute ckt_area
        /* Traverse fanout cone of g */
        /* Visit each output node affected by this change */
        /* If slack changed, assign new FF variant */
        Recompute ckt_delay, ckt_SER
        Calculate sensitivity from EQ5
    Pick the gate with the best sensitivity
    Make a "move" by resizing this gate appropriately
    /* Change FF assignments if necessary */
    Repeat resizing operation
```

Figure 6. Pseudo-code for the proposed algorithm combining flipflop assignment with gate resizing

circuit SER due to the electrical characteristics associated with this gate. In other words, resizing this NAND3 gate could possibly result in only a small filtering of the transients arriving at this gate.

On the other hand, up sizing this cell potentially reduces the delay along the paths through the gate. For instance, suppose the sizing operation modifies the delay of the long path such that the slack at the output node changes from 80ps to 100ps. From Section 4.1 and Table 1, we recognize that the flipflop at this output can be changed from F100 to F130 (without affecting delay) thereby filtering all the pulses incident at the output node and obtaining even larger SER improvement. Thus, even if the SET waveforms at an output node are not affected due to the resizing of a candidate, the change in slack value at the gate can result in significant SER reductions due to the ability to reassign the flipflop. This concept of slack creation amplifies the usefulness of the combined optimization approach. Further, a small set of gates in each circuit enable both slack creation and SET waveform reduction such that the synergy between the two techniques produces considerable reductions in the total SER of the circuit.

The structure of the algorithm (see Figure 6) is similar to the one presented in Section 4.1. We first perform an initial pass on the output nodes and assign flipflops according to the slack availability. For each output node the set of gates on the critical path to this node can significantly affect the slack produced at the node. We recognize the potential gains offered by these gates by augmenting the candidate set C with the cells on the critical path to each output node. After an incremental recomputation of circuit delay, we visit all output nodes whose arrival times are affected and modify the output flipflop to the appropriate type. During the sensitivity calculation in Figure 6, changes in SER due to both sizing and flipflop assignment are reflected in the total SER value. At the end of a single resizing move, we update the flipflop assignment appropriately. Note that the complexity of this combined algorithm is identical to the complexity gate resizing algorithm (see Section 4.1.4).

The unified optimization method is expected to provide better SER reduction than either FF assignment or gate sizing considered separately. The additional sizing step after FF assignment further targets the gates most contributing to the total circuit SER. Moreover, compared to the sizing-only optimization method, a smaller fraction of gates need to be resized since the flipflop variants significantly filter out a large portion of the output transient waveforms.

5 Results

The proposed algorithms were implemented in C++ and run on a dual processor, AMD Opteron 2.4GHz machine with 4GB RAM running Linux. We used an industrial 0.13µm standard cell library consisting of four sizes of inverters, NANDs and NORs. All SER measurements were performed assuming a sea-level neutron flux of $56.5m^{-2}s^{-1}$. We employ three sets of benchmark circuits in our analysis: the ISCAS-85 suite [27], the MCNC circuit set [28] and standard multiplier circuits. In this paper, we present results for a subset of the largest circuits from these benchmark sets.

The flipflop based optimization approaches proposed in this paper rely on the amount of slack available to optimize the circuit SER. To provide an accurate assessment of the proposed approaches, it is necessary to quantify the SER improvement for circuits with different amounts of slack. We therefore synthesize each benchmark for two separate delay constraint values: a tight delay constraint circuit (TDCC) corresponding to a 5% backoff from the fastest possible circuit implementation and a loose delay constraint circuit (LDCC) corresponding to a 30% backoff point. Current CMOS designs are severely limited by the amount of power that they dissipate so that the usage of circuits with loose delay constraints (20-30% backoff) has become more prevalent to meet the power budget. Moreover, the 5% backoff point is fairly aggressive since it is typically beyond the knee of the power/delay curve and as such represents a highly constrained design.

Circuits with tighter delay constraints will naturally contain a substantial number of sized up gates. However, due to the higher fraction of large gates, the number of locations at which SETs are injected is reduced thereby producing a lower value for the overall circuit SER. In Table 2 we list the circuit SER (with the FIT rates scaled by 1E-05) and the circuit area (with units in microns since area is defined as the sum of all device widths) for both LDCCs and TDCCs. On average, TDCCs have roughly half the SER while the area is doubled. Table 2 also includes the number of primary outputs (POs) for each circuit. In the subsequent analysis, we measure overheads in delay, area and SER from this initially specified design point for each type of circuit.

Table 2. Comparison of baseline loose/tight delay constraint circuits. Ckt SER has FITs scaled by 1E-05s

Ckt	POs	LDCC			TDCC		
		Gates	Ckt SER	Ckt Area	Gates	Ckt SER	Ckt Area
i6	67	734	15.0	2575.7	783	2.6	6175.5
i7	67	943	8.0	3683.5	1000	0.8	8986.1
i8	81	1610	15.6	6077.6	1919	5.9	13364.3
i9	63	1026	10.9	3597.2	1172	1.1	9684.1
i10	224	3393	30.5	10730.8	3663	24.3	17928.6
c432	7	247	0.3	1144.2	279	0.1	2211.1
c499	32	750	1.9	4750.4	826	0.1	8554.2
c880	26	608	3.2	2295.3	768	2.1	4901.4
c1355	32	741	1.5	3836.5	774	0.2	7363.3
c1908	25	753	4.0	3720.5	859	1.8	6915.4
c3540	22	1950	2.6	7608.2	2124	1.7	14077.3
c6288	32	5216	4.7	25788.7	6117	4.2	46600.1
m8x8	16	1334	3.3	6856.4	1543	2.1	12841.4
m16x16	32	6217	7.9	33382.4	7234	5.2	57857.8
Avg			7.8	8289.1		3.7	15532.9

We label the three proposed optimization techniques as: (T1) Gate sizing only (Section 4.1) (T2) Slack-based FF assignment (Section 4.2) and (T3) Combined FF assignment and gate sizing (Section 4.3). Table 3 first demarcates the LDCCs from TDCCs. For each type of baseline circuit, we apply the three proposed techniques and quantify the reduction ratio (between the baseline SER and the optimized SER), % increase in circuit area, number of gates resized, and algorithm runtime. Recall from earlier discussions that there is no delay penalty and the maximum area penalty is set to 20%. Since T2

Table 3. SER Reduction, area change (%), number of resized gates, and runtimes for the three optimization techniques. T1 = gate sizing only, T2 = slack-based FF assignment, T3 = combined FF assignment and gate sizing

Ckt	Loose Delay Constraint Circuits (LDCC)									Tight Delay Constraint Circuits (TDCC)								
	SER(base)/SER(opt)			% Area change		# Resized gates		Runtime (s)		SER(base)/SER(opt)			% Area change		# Resized gates		Runtime (s)	
	T1	T2	T3	T1	T3	T1	T3	T1	T3	T1	T2	T3	T1	T3	T1	T3	T1	T3
i6	4.2X	1.6X	13.9X	7.1	7.1	117	116	8.3	5.6	2.0X	1.0X	2.0X	2.2	1.4	44	44	1.0	0.9
i7	4.6X	2.1X	12.8X	9.0	5.9	105	79	20.2	19.9	2.7X	1.2X	3.0X	0.5	0.5	16	16	0.7	0.6
i8	5.7X	2.1X	8.9X	5.3	3.0	156	110	42.7	37.1	4.1X	4.0X	10.8X	0.1	0.0	66	40	8.1	7.7
i9	5.4X	1.2X	14.7X	19.0	15.3	143	123	22.2	22.0	3.5X	1.0X	4.4X	1.1	0.8	26	20	2.8	1.9
i10	3.2X	19.5X	34.0X	8.8	1.2	199	33	131.6	116.4	2.8X	5.5X	30.6X	3.7	0.3	177	31	81.4	77.9
c432	6.0X	1.1X	12.8X	2.3	2.3	5	5	0.8	0.7	5.6X	1.2X	12.1X	0.2	0.2	2	2	0.2	0.2
c499	11.4X	2.0X	17.5X	0.8	0.6	55	44	9.4	8.5	1.0X	1.0X	1.3X	0.0	0.1	0	2	0.1	1.1
c880	7.1X	9.7X	42.5X	9.0	3.2	52	16	19.5	6.9	6.7X	1.2X	35.6X	1.5	0.3	32	6	6.9	2.2
c1355	3.7X	2.0X	9.2X	0.9	0.3	33	27	9.1	7.9	1.1X	1.0X	4.2X	0.1	0.1	2	2	0.1	2.2
c1908	23.7X	2.4X	26.7X	5.3	2.2	73	18	15.0	9.8	10.6X	2.3X	43.4X	1.6	1.1	43	29	4.8	4.8
c3540	8.8X	5.8X	57.3X	1.2	0.3	33	8	51.7	15.2	8.6X	4.7X	27.4X	0.5	0.1	22	4	17.7	4.9
c6288	10.4X	26.3X	30.9X	1.1	0.1	73	4	195.0	17.6	7.6X	6.2X	28.9X	0.6	0.1	74	12	98.3	33.4
m8x8	11.2X	6.5X	47.2X	2.2	0.8	45	11	25.8	8.3	5.2X	3.2X	41.4X	0.7	0.4	27	11	6.8	5.0
m16x16	5.7X	9.4X	66.3X	1.1	0.4	78	18	111.8	48.5	5.2X	6.1X	30.9X	0.3	0.2	58	19	83.4	48.9
Avg	7.9X	6.6X	28.2X	5.2	3.1					4.8X	2.8X	19.7X	0.9	0.4				

is a simple FF assignment algorithm that does not involve any modification of the gates, the area increase and number of gates resized is 0, and the runtime related to this reassignment is negligible.

The circuit delay tightness plays an important part in determining the performance of all three optimization techniques. For a TDCC, a larger fraction of gates are on critical or near critical paths, so that a particular resizing move on a specific gate may be disallowed since it violates delay constraints. On the other hand, for LDCCs a large number of gates have no impact on circuit delay and can be resized to achieve SER savings. Thus, comparing SER reductions for the two types of circuits by the application of T1, we observe that circuit SER is reduced on average by 7.9X in a LDCC versus only 4.8X in a TDCC. However, since the baseline SER of a TDCC is lower, the final FIT rate of the optimized TDCC, despite the smaller amount of SER reduction, will be less than the final FIT rate of an optimized LDCC. The larger number of critical paths also implies that the arrival times at several output nodes will be nearly identical. Hence, the amount of slack at each output node is small which lowers the gains offered by T2. On average, T2 produces SER reductions of 2.8X in a TDCC compared to 6.6X in a LDCC. However, since T2 is a technique that consumes zero area and delay overhead, it is still an attractive alternative due to its simplicity.

The slack creation concept described in Section 4.3 plays an important role in reducing the SER particularly in the TDCCs. We observe here that in general, T3 produces significantly more reductions compared to T1 while resizing a fewer number of gates. This effect is primarily due to the ability for T3 to identify slack-critical gates in the design. The sensitivity metric corresponding to such gates is particularly high given the possibility of achieving even greater gains by reassigning an output flipflop. Thus, generating enough slack by sizing even a small number of gates produces significant gains in the circuit SER value.

On average, the combined SER optimization method, T3, outperforms both T1 and T2 with average savings of up to 28.2X for LDCCs. The number of gates resized and, consequently, the area overhead using T3 is always lower than for T1. Furthermore, T3 runtime is also smaller than T1.

Although we limit area overhead to 20% we observe that in most cases the area increases by a much smaller amount (about 5-6%) and at most 200-250 gates in the entire circuit are resized. The runtimes for both T1 and T3 are quite small and on the order of 1-3 minutes.

6 Conclusions

In this work, we presented novel soft error rate optimization techniques for combinational circuits. These involve a sensitivity-based gate sizing algorithm, a slack-based flipflop assignment method, and a joint optimization approach combining flipflop assignment with gate sizing into a single algorithm. We explored the effectiveness of these methods for circuits synthesized at different delay constraints. Depending on the amount of slack available in the circuit and the amount of area overhead that is tolerable, we can choose between the three techniques to achieve the best circuit solution. Experimental results show SER reductions of up to 28.2X while accruing an area overhead of ~6% and no delay penalties.

7 References

[1] R. Baumann, "Soft errors in advanced computer systems," *IEEE Design and Test of Computers (D & T)*, 22 (3), pp. 258-266, May 2005.

[2] P. Shivakumar, M.Kistler, S. Keckler, D. Burger, L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," *Intl. Conf. on Dependable Systems and Networks (DSN),* pp. 389-398, Jun 2002.

[3] T. Karnik, B. Bloechel, K. Soumyanath, V. De, S. Borkar, "Scaling trends of cosmic ray induced soft errors in static latches beyond 0.18µm," *Symp. on VLSI Circuits,* pp. 61-62, Jun 2001.

[4] S. Mitra, N. Seifert, M. Zhang, Q. Shi, K. Kim, "Robust system design with built-in soft-error resilience," *IEEE Computer*, 38(2), pp. 43-52, Feb 2005.

[5] T. Karnik, S. Vangal, S. Veeramachaneni, P. Hazucha, V. Erraguntla, S. Borkar, "Selective node engineering for chip-level soft error rate improvement," *Symp. on VLSI Circuits*, pp. 204-205, Jun 2002.

[6] S. Mitra, T, Karnik, N. Seifert, M. Zhang, "Logic soft errors in sub-65nm technologies design and CAD challenges," *Design Automation Conf. (DAC),* pp. 2-3, Jun 2005.

[7] M. Baze, S. Buechner, D. Mcmurrow, "A CMOS design technique for SEU hardening," *IEEE Trans. on Nuclear Science (TNS)*, 47(6), pp. 263-2608, Dec 2000.

[8] K. Moharam, N. Touba, "Cost-effective approach for reducing the soft error failure rate in logic circuits," *Intl. Test Conf. (ITC)*, pp. 893-901, Sep 2003.

[9] Y. Dhillon, A. Diril, A. Chatterjee, "Soft-error tolerance analysis and optimization of nanometer circuits," *Design Automation and Test in Europe (DATE)*, pp. 288-293, Mar 2005.

[10] Q. Zhou, K. Mohanram, "Cost effective radiation hardening technique for combinational logic," *Intl. Conf. on Computer-Aided Design (ICCAD)*, pp. 100-106, Nov 2004.

[11] Y. Dhillon, A. Diril, A. Chatterjee, C. Metra, "Load and logic co-optimization for design of soft-error resilient nanometer CMOS circuits," *Intl. Online Testing Symp. (IOLTS),* pp. 35-40, Jul 2005.

[12] C. Zhao, S. Dey, "Improving transient error tolerance using robustness compiler (ROCO)," *Intl. Symp. on Quality Electronic Design (ISQED)*, pp. 133-138, Mar 2006.

[13] M. Zhang, N. Shanbhag, "An energy-efficient circuit technique for single event transient noise-tolerance," *Intl. Symp. on Circuits and Systems (ISCAS)*, pp. 636-639, Jun 2005.

[14] D. Mavis, P. Eaton, "Soft error rate mitigation techniques for modern microcircuits," *Intl. Reliability Physics Symp. (IRPS)*, pp. 216-225, Apr 2002.

[15] M. Zhang, N. Shanbhag, "A CMOS design style for logic circuit hardening," *Intl. Reliability Physics Symp. (IRPS)*, pp. 223-229, Apr 2005.

[16] H. Cha, J. Patel, "Latch design for transient pulse tolerance," *Intl. Conf. on Computer Design (ICCD)*, pp. 385-388, Oct 1994.

[17] V. Joshi, R. R. Rao, D. Blaauw, D. Sylvester, "Logic SER reduction through flipflop redesign," *Intl. Symp. on Quality Electronic Design (ISQED)*, pp. 611-616, Mar 2006.

[18] P. Elakkumanan, K. Prasad, R. Sridhar, "Time redundancy based scan flip-flop reuse to reduce SER of combinational logic," *Intl. Symp. on Quality Electronic Design (ISQED)*, pp. 617-622, Mar 2006.

[19] J. Fishburn, A. Dunlop, "TILOS: A posynomial programming approach to transistor sizing," *Intl. Conf. on Computer-Aided Design (ICCAD)*, pp. 326-328, Nov 1985.

[20] C. Zhao, X. Bai, S. Dey, "A scalable soft spot analysis methodology for compound noise effects in nano-meter circuits," *Design Automation Conf. (DAC),* pp. 894-899, Jun 2004.

[21] M. Zhang, N. Shanbhag, "A soft error rate analysis (SERA) methodology," *Intl. Conf. on Computer-Aided Design (ICCAD)*, pp. 111-118, Nov 2004.

[22] R. Rajaraman, J. Kim, N. Vijaykrishnan, Y. Xie, M. Irwin, "SEAT-LA: A soft error analysis tool for combinational logic," *Intl. Conf. on VLSI Design (VLSID)*, pp. 499-502, Jan 2006.

[23] R. R. Rao, K. Chopra, D. Blaauw, D. Sylvester, "An efficient static algorithm for computing the soft error rates of combinational circuits," *Design Automation and Test in Europe (DATE)*, pp. 164-169, Mar 2006.

[24] B. Zhang, M. Orshansky, "FASER: Fast analysis of soft error susceptibility for cell based designs," *Intl. Symp. on Quality Electronic Design (ISQED)*, pp. 755-760, Mar 2006.

[25] L. Freeman, "Critical charge calculations for a bipolar SRAM array," *IBM Journal of Research & Development*, 40(1), 1996.

[26] P. Hazucha, C. Svensson, "Impact of CMOS technology scaling on atmospheric neutron soft error rate," *IEEE Trans. on Nuclear Science (TNS),* 47(6), pp. 2586-2594, Dec 2000.

[27] F. Brglez, H. Fujiwara, "A neural netlist of ten combinational benchmark circuits and translator in Fortran," *Intl. Symp. on Circuits and Systems (ISCAS)*, pp. 663-698, Jun 1985.

[28] S. Yang, Logic synthesis and optimization benchmarks user guide, MCNC, Research Triangle Park, North Carolina, 1991.