Clock Skew Scheduling for Soft-Error-Tolerant Sequential Circuits*

Kai-Chiang Wu and Diana Marculescu
Department of Electrical and Computer Engineering
Carnegie Mellon University
{kaichiaw, dianam}@ece.cmu.edu

Abstract
Soft errors have been a critical reliability concern in nanoscale integrated circuits, especially in sequential circuits where a latched error can be propagated for multiple clock cycles and affect more than one output, more than once. This paper presents an analytical methodology for enhancing the soft error tolerance of sequential circuits. By using clock skew scheduling, we propose to minimize the probability of unwanted transient pulses being latched and also prevent latched errors from propagating through sequential circuits repeatedly. The overall methodology is formulated as a piecewise linear programming problem whose optimal solution can be found by existing mixed integer linear programming solvers. Experiments reveal that 30-40% reduction in the soft error rate for a wide range of benchmarks can be achieved.

1. Introduction
Soft errors, process variations, and device aging phenomena are currently some of the main factors in reliability degradation. With the continuous scaling of transistor dimensions, soft errors, which cause unpredictable transient circuit failure, are becoming increasingly dominant for functional reliability concerns [1]. A radiation-induced charged particle passing through a microelectronic device ionizes the material along its path and generates free pairs of electrons and holes. The free (ionized) carriers deposited around the particle track can be attracted or repelled by an internal electric field of the device and lead to an electrical pulse, referred to as a single-event transient (SET) or a glitch. A single-event upset (SEU) or a soft error refers to transient bit corruption that occurs when a single-event transient is large enough to flip the state of a storage node. The rate at which soft errors occur is called soft error rate (SER).

During SEU propagation in logic, three mechanisms used to provide logic circuits with effective protection against soft errors: (i) logical masking, (ii) electrical masking, and (iii) latching-window (timing) masking [2]. However, as technology scaling proceeds aggressively (e.g., decreasing node capacitance and increasing clock frequency), the impact of these three masking mechanisms is lessened. On the other hand, error detecting and correcting codes have been mature enough to successfully mitigate soft error susceptibility of memory elements. A recent study [3] showed that soft errors significantly degrade the robustness of logic circuits, while the nominal SER of SRAMs tends to be nearly constant from 130nm to 65nm technologies. As a result, unless explicitly dealt with, the SER of logic will become as great of a concern and is expected to be comparable to that of unprotected memories by 2011 [4].

When the combinational block of a sequential circuit can propagate SETs freely, the sequential circuit may become very sensitive to such events. This is because, once latched, soft errors can circulate through the circuit in subsequent clock cycles and affect more than one output, more than once. The untraceable propagation of soft errors greatly affects the circuit operation for consecutive cycles and thus, necessitates design methods for soft error tolerance of sequential circuits, in a similar manner to classic design constraints such as performance and power consumption.

In this paper, we present an analytical methodology for soft error tolerance of sequential circuits. Our work proposes to adjust the arrival times of clock signals to memory elements (latches or flip-flops) such that the probability of capturing unwanted transient pulses is significantly decreased. The technique, called clock skew scheduling (CSS), is formulated in our methodology as a piecewise linear programming (PLP) problem, and its optimal solution can be found by existing mixed integer linear programming (MILP) solvers. The proposed framework involves only minor modifications of the clock tree synthesis step and does not touch the combinational logic of sequential circuits. Hence, this CSS-based approach can also act as a post-processing procedure for additional SER improvement on top of techniques targeting only combinational logic, which typically change the circuit timing and topology (e.g., resizing [5] and rewiring [6]).

The rest of this paper is organized as follows: Section 2 gives an overview of related work and outlines the contribution of our paper. In Section 3, we illustrate an example motivating clock skew scheduling for soft error tolerance. Section 4 introduces several metrics associated with SER analysis. In Section 5, our proposed framework, using clock skew scheduling and based on a piecewise linear programming formulation, is presented. Section 6 reports the experimental results for a set of standard benchmarks. Finally, we conclude our work in Section 7.

2. Related Work and Paper Contribution
2.1. Previous Work on Soft Error Tolerance
Intensive research has been done in the area of soft error tolerance for combinational circuits. To reduce the overall cost for realizing soft error tolerance, gate resizing [5] and partial duplication [7] strategies target only nodes with high error susceptibility and ignore nodes with low error susceptibility. A potentially large overhead in area and power is still needed for a higher degree of soft error tolerance. In [8] and [9], voltage scaling/assignment is used to enhance the circuit robustness to soft errors. These methods trade power penalty for SER reduction by applying higher supply voltage(s) to a certain portion of gates. Approaches based on rewiring or resynthesis [6][10] can achieve relatively smaller SER improvement while incurring little overhead.

Sequential circuits, as opposed to combinational circuits, have received less attention in terms of soft error tolerance. Since a sequential circuit has a feedback loop leading back to state inputs of the circuit, it is possible that errors latched at state lines propa-
gate through the circuit for multiple clock cycles. The intuitive way to address this problem is by replacing sequential elements with hardened latches or flop-flips that are less sensitive to soft errors, as developed in [11]. A flip-flop sizing scheme [12] increases the probability of timing masking by lengthening the latching window intervals of vulnerable flip-flops. Nevertheless, this scheme does not take into account logical masking and electrical masking, which are also important factors in determining circuit SER. In [13], gates are locally relocated such that, for each gate, delays to different outputs are balanced as much as possible. In effect, this strategy minimizes the probability that an error originating at a gate is registered by any of the flip-flops. The error, however, may reach more than one output simultaneously due to balanced path delays and be registered by multiple flip-flops, resulting in so-called multiple-bit upsets (MBUs). For sequential circuits, MBUs imply that there will be multiple errors propagating in subsequent cycles, further degrading circuit reliability. This is a crucial reliability concern in sequential circuits that has not been addressed so far.

2.2. Paper Contribution

This paper presents a SER mitigation framework where the MBU impact is explicitly considered and alleviated. To the best of our knowledge, this is the first work addressing MBU-aware soft error tolerance in sequential circuits. On one hand, for an original error (SEU) in the clock cycle when a particle strikes, we maximize the probability of timing masking via clock skew scheduling (CSS). On the other hand, during clock cycles following the particle hit, we avoid multiple errors (MBU) from propagating repeatedly by exploring the effects of (i) implication-based masking and (ii) mutually-exclusive propagation, as explained later in Section 3.1 and Section 3.2, respectively. In this paper, we take advantage of intentionally induced skews to increase the probability of timing masking via CSS, while accounting for the MBU impact to further enhance soft error robustness. The contributions and advantages of our framework are twofold:

- **Optimality/Complexity**: The overall methodology for MBU-aware soft error tolerance is formulated as a piecewise linear programming (PLP) problem and its optimal solution can be found by existing mixed integer linear programming solvers. The worst-case problem size of our PLP formulation is \(O(n^2)\) where \(n\) is the number of flip-flops in a sequential circuit. Therefore, the runtime spent on solving the PLP-based SER mitigation problem is quite reasonable.

- **Compounding results**: CSS itself involves only modifications of clock tree synthesis during the physical design stage. In other words, the difference between original and optimized designs lies in their clock trees, whereas the combinational network remains identical. Hence, our CSS-based framework, when applied as a post-processing procedure, can provide additive SER reduction without destroying existing SER improvements. On average, an extra 30-40% reduction in SER can be achieved with a drastic decline of MBU effects.

3. A Motivating Example

To motivate the use of clock skew scheduling for soft error tolerance, we use benchmark \(s27\) (see Figure 1) from the ISCAS’89 suite, where flip-flops (FFs) are positive-edge-triggered. Without loss of generality, we assume that the delay of each gate is 1 (unit delay model) and wires do not contribute to the circuit delay. The assumption can be relaxed for a non-uniform delay model, with consideration of wire loads. In this example, we focus on a SEU which occurs at gate \(G_8\) and may be captured by flip-flops \(FF_2\) and/or \(FF_3\).

**Definition 1** (error-latching window): The error-latching window [13] of a flip-flop is a time interval, \([t-t_s, t+t_s]\), where \(t\) is the moment when a clock edge happens, \(t_s\) and \(t_b\) are the setup and hold times of the flip-flop. An error must be present during this interval to be latched; otherwise, it is filtered by latching-window (timing) masking. The error-latching window associated with a flip-flop can be backward propagated to internal gates (according to respective propagation delays) to determine when an error has to occur to be latched by that flip-flop.

Under unit delay model, the delays from \(G_8\) to \(FF_2\) and to \(FF_3\) are 0 and 1, respectively. Our goal is to overlap the error-latching windows of \(FF_2\) and \(FF_3\) at \(G_8\) by adjusting the arrival times of clock signals to \(FF_2\) and/or \(FF_3\), which in effect decreases the probability that an error at \(G_8\) is latched with increased impact of timing masking. The idea of overlapping error-latching windows, first proposed in [13], is based on the fact that the probability of timing masking is inversely proportional to the sum of sizes of disjointed error-latching windows. For example, in Figure 2(a), there are two separate error-latching windows at \(G_8\) (one at time \(t-1\) and the other at \(t\)) before skewing any flip-flop. If we lengthen the arrival time of clock signals to \(FF_3\), by 1 and its new error-latching window is shown as the upper right diagram in Figure 2(b), there will be only one joint error-latching window at \(G_8\) (at time \(t\)) due to complete overlapping. This implies that, after skewing \(FF_3\), only errors occurring at \(G_8\) during the error-latching window at time \(t\) will be latched, while errors occurring during the already-non-existing window at time \(t-1\) will be filtered by timing masking, leading to a significant reduction in SER. Since the overlapped error-latching window (at time \(t\)) can be backward propagated to primary inputs, the positive impact on circuit SER is also valid for those gates in \(G_8\)’s fanin cone.

However, in the case where \(FF_1\) has been skewed, MBUs may become more frequent because an error occurring at \(G_8\) during the joint error-latching window at time \(t\) will be latched by both \(FF_2\) and \(FF_3\) simultaneously. Instead of using all flip-flops in a sequential circuit as candidates for clock skew scheduling, we carefully pick pairs of flip-flops that are beneficial for MBU elimination. In the sequel, we demonstrate how to identify pairs of flip-flops that are capable of alleviating MBU effects (during clock cycles subsequent to particle hits) and suitable to be managed by CSS for MBU-aware soft error tolerance.
3.1. Implication-Based Masking

We consider the following example to illustrate the concept of implication-based masking required for our methodology. The function of primary output $O$ of circuit $s27$ is:

$$O = (a + f') + (c + d' + e + g) \quad (1)$$

The complement of Boolean difference of $O$ with respect to (w.r.t.) $FF_2$’s present-state line $f$ is:

$$F = (\partial O / \partial f)' = a + c'd'e' + g \quad (2)$$

Equation (2) represents the Boolean expression of logical masking patterns for errors propagated from $f$ to $O$.

Similarly, the complement of Boolean difference of $O$ w.r.t. $FF_1$’s present-state line $g$ is:

$$G = (\partial O / \partial g)' = (a + f')(c + d' + e) \quad (3)$$

Note that $F$ is a function of $g$ and $G$ is a function of $f$, where $f$ and $g$ are present-state lines of $FF_2$ and $FF_1$ and may be corrupt due to the presumed SEU at $G_6$. To remove $f$ and $g$ while keeping the logical masking patterns, we apply universal quantification.

The universal quantification of $F$ w.r.t. $g$ is:

$$\forall_g F = F_{g=1} \cdot F_{g=0} = a + c'd'e' \quad (4)$$

Equation (4) describes the patterns for logical masking of errors from $f$ to $O$, for all possible values of $g$ (0 and 1). Since we do not know whether $g$ is corrupt, applying universal quantification makes sense and will correctly reflect logical masking of errors from $f$ to $O$, irrespective of $g$.

Similarly, the universal quantification of $G$ w.r.t. $f$ is:

$$\forall_f G = G_{f=1} \cdot G_{f=0} = a \cdot (c + d' + e) \quad (5)$$

Up to now, (4) and (5), which no longer include $f$ or $g$, have been functions of inputs $a$, $c$, $d$, and $e$. In addition, one can find that (5) is a subset of (4); that is to say, with respect to $O$, the logical masking of an error on $g$ implies the logical masking of an error on $f$. More precisely in this case, both errors on $f$ and $g$ will be masked when (5) is satisfied.

**Definition 2** (implication-based masking): A pair of flip-flops $X$ and $Y$ is called an implication-based masking (IM) pair if, with respect to all outputs and flip-flops:

(i) the set of logical masking patterns for errors propagated from $X$ (denoted by $LM(X)$) contains the one for errors from $Y$ (denoted by $LM(Y)$), i.e., $LM(X) \supseteq LM(Y)$, or

(ii) the set of logical masking patterns for errors propagated from $Y$ (denoted by $LM(Y)$) contains the one for errors from $X$ (denoted by $LM(X)$), i.e., $LM(Y) \supseteq LM(X)$.

Based on Definition 2, the first category of candidates for CSS can be identified. In circuit $s27$, as shown in Figure 1, $(FF_2$ and $FF_3)$ is a pair of candidates falling into this category. By overlapping the error-latching windows of these two flip-flops via CSS (see Figure 2(b)), not only can SER be reduced, but also CSS-induced MBUs will be eliminated by implication with a certain probability. This will be demonstrated in Section 6.

3.2. Mutually-Exclusive Propagation

The second type of candidate flip-flops, mutually-exclusive propagation pair, in $s27$ can be identified by a single side-input assignment, where a side input is a wire along which no error is propagated. Again, we focus on a SEU which occurs at $G_8$ and may be captured by $FF_2$ and/or $FF_3$.

To propagate errors from $FF_1$’s present-state line $g$ to $R$, $G_{10}$ needs a non-controlling value “0” on its side input $G_7 \rightarrow G_{10}$. As seen in Figure 1, the value assignment at the output of $G_7$ is a controlling value for $G_8$, at which errors from $FF_2$’s present-state line $f$ are thus logically masked. Therefore, with respect to $R$, the propagation of an error on $g$ implies that an error propagated from $f$ is logically masked. In other words, errors on $f$ and $g$ cannot be observable at $R$ simultaneously.

**Definition 3** (mutually-exclusive propagation): A pair of flip-flops $X$ and $Y$ is called a mutually-exclusive propagation (MEP) pair if, with respect to all outputs and flip-flops, the set of logical masking patterns for errors propagated from $X$ (LM(X)) contains the complement of the one for errors from $Y$ (LM(Y)), i.e., $LM(X) \supseteq LM(Y)'$. Intuitively, the sets of patterns for propagating errors from $X$ and $Y$ (LM(X) and LM(Y)) are disjoint.

Based on Definition 3, the second category of candidates for CSS can be identified. Similar to IM pairs, we can overlap the error-latching windows of two flip-flops falling into this category (e.g., $FF_2$ and $FF_3$ in $s27$) to achieve MBU-aware soft error tolerance because, due to the property of mutually-exclusive propagation, at least one of the two errors propagated from this pair of flip-flops will be logically masked before reaching a primary output or a flip-flop. The mutually-exclusive property guarantees that the MBU impact after applying CSS is at most equivalent to the case of not applying CSS, whereas circuit SER can be significantly reduced as a result of increased timing masking. It is also probable that two errors from a MEP pair are both masked and consequently less MBU impact is expected.

Any two flip-flops are regarded as candidates and will be
beneficial for SER reduction as long as they are either IM or MEP pairs. These two properties are the major motivation for our framework aiming at soft error tolerance, and both address the MBU issue by mitigating the occurrence of multiple-bit upsets. More precisely, as mentioned earlier, overlapping the error-latching windows of flip-flops increases the probability of timing masking and in turn decreases the soft error rate of a circuit. Furthermore, overlapping the error-latching windows of a candidate pair of flip-flops, which meet the IBM or MEP condition, can not only reduce circuit SER but also alleviate potential MBU effects. Hence, for our objective of MBU-aware soft error tolerance, we check all possible pairs of flip-flops and extract as candidates for the proposed CSS-based framework those satisfying the IM or MEP property.

4. Analysis of Soft Error Susceptibility

Before presenting the overall methodology for MBU-aware soft error tolerance, we briefly introduce two metrics associated with SER analysis in this section. The metrics, mean error impact (MEI) and mean error susceptibility (MES), are used to evaluate the circuit susceptibility to soft errors. Relying on a symbolic framework [14][15] which provides unified treatment of three masking mechanisms through decision diagrams, MEI and MES are calculated and thereafter, the soft error rate (SER) of a sequential circuit can be derived accurately and efficiently.

4.1. Mean Error Impact (MEI) of Internal Gates

The MEI value of a gate quantifies the probability that at least one primary output is affected by an error originating at this gate. The larger MEI a gate has, the higher the probability that an error occurring at this gate will be latched. This implies that those gates with higher MEI make the circuit more vulnerable to soft errors. Please refer to [14][15] for more details about MEI.

4.2. Mean Error Susceptibility (MES) of Primary Outputs

For each primary output Fi, initial duration dj and initial amplitude a, mean error susceptibility (MES) [14] is defined as the probability of output Fi failing due to errors at internal gates. In [14][15], the authors compute MES of each primary output in the circuit for a discrete set of pairs (d, a) of initial glitch durations and amplitudes. Then, the probability of output Fi failing (output failure probability) due to errors with various durations and amplitudes is calculated as a weighted sum of the discrete set of MES values. Finally, the soft error rate (SER) of output Fi can be derived based on the output failure probability.

5. Clock Skew Scheduling Based on Piecewise Linear Programming

The motivating example in Section 3 is a special case of CSS for MBU-aware soft error tolerance. A fundamental assumption in the example is that we can completely overlap the error-latching windows of a given pair of flip-flops (FFs) which have been recognized as candidates for CSS. This assumption is not realistic because it is not always possible to completely overlap error-latching windows without incurring any timing violations, i.e., setup time violations owing to long paths or hold time violations owing to short paths. Moreover, adjusting the skew between two FFs may also change skews between affected FFs and unaffected FFs. For a large sequential circuit with hundreds of FFs, optimal skew scheduling, shown to be a signomial problem [16], is difficult to be determined algorithmically. To address this problem, we develop an analytical method which can apply CSS with a global view on all extracted candidate FFs while suppressing timing violations. A generalized problem formulation, based on piecewise linear programming (PLP), is presented in the sequel.

5.1. Problem Formulation

Given a non-skewed sequential circuit (i.e., skew(FFi, FFj) = 0 for all i and j) and all possible pairs of flip-flops as candidates for MBU elimination, our objective is to achieve the highest level of MBU-aware soft error tolerance by maximizing the overlap between error-latching windows of each flip-flop pair via clock skew scheduling.

Definition 4 (intersecting gate): The intersecting gate of two flip-flops FFi and FFj is the root gate for the intersection of FFi’s and FFj’s fanin cones. In case of more than one such gate, the one with the largest MEI value is selected.

Definition 5 (skew): Given two flip-flops FFi and FFj for which the arrival times to clock pins are ci and cj respectively, the skew between FFi and FFj, denoted by skew(FFi, FFj), is (ci – cj).

In Figure 3, flip-flops FFi and FFj are a pair of candidates whose intersecting gate is gate G0. The propagation delays from G0 to FFi and to FFj are denoted by di and dj respectively. Let the amounts of adjustments in the arrival times of clock signals to FFi and FFj be si and sj, where sj and si can be positive or negative. To completely overlap the error-latching windows of FFi and FFj at G0, we have to determine si and sj such that skew(FFi, FFj) = (si – sj) = (di – dj). But complete overlapping may need significantly large |si| and/or |sj| and thereby, may induce timing violations, which must be avoided in the resulting design. To suppress timing violations, we set up the first two constraints as follows.

For each possible pair of flip-flops FFi (skewed by si) and FFj (skewed by sj) between which there exist combinational paths from FFi to FFj, (6) is to prevent setup time violations and (7), hold time violations:

\[ s_i + t_{iq} + A_{ij} + t_{iq} < s_j + T_{clk} \]  
\[ s_i + t_{iq} + A_{ij} > s_j + t_h \]

where T_{clk} is the clock period of the sequential circuit, t_{iq}, t_{iq}, and t_h are respectively the clock-to-output delay, setup and hold times of flip-flops, and A_{ij} and a_{ij} are the maximum and minimum delays
of combinational paths from \( FF_i \) to \( FF_j \), which can be obtained by performing static timing analysis.

Let \( w_y \) denote the reduction in SER of the given circuit obtained by completely overlapping the error-latching windows of \( FF_i \) and \( FF_j \), at \( G_2 \). The reason for selecting an intersecting gate with the largest MEI is that, by doing so, it is very likely to obtain the largest \( w_y \) for CSS.

The theoretical optimal SER reduction is:

\[
\sum_{i,j,k} f_{ij,k} w_{ij,k} \text{(Candidates)} (w_{ij,k})
\]

Since the optimum (8) may be unachievable due to constraints (6) and (7), we use another variable, \( f_y \) (0 \( \leq f_y \leq w_y \)), to denote the actual reduction in SER resulting from the overlapping (complete or partial) of \( FF_i \)'s and \( FF_j \)'s error-latching windows. Figure 4 shows \( f_y \) as a function of \( s_y \). The rationale behind is that, once overlapped, \( f_y \) is linearly proportional to the size of the overlap between \( FF_i \)'s and \( FF_j \)'s error-latching windows, and \( f_y = w_y \) when completely overlapped at \( s_y = (d_i - d_j) \).

From Figure 4, one can note that the relationship of \( f_y \) versus \( s_y \) is neither convex nor concave. Instead, the formulation becomes piecewise linear if \( f_y(s_y) \) is broken into four pieces: \( s_y = (d_i - d_j) - (t_m + t_0), s_y = (d_i - d_j), \) and \( s_y = (d_i - d_j) + (t_m + t_0) \). By introducing four new binary variables \( p_{y,1}, p_{y,2}, p_{y,3}, \) and \( p_{y,4} \) such that

\[
p_{y,1} + p_{y,2} + p_{y,3} + p_{y,4} = 1
\]

and four new floating variables \( r_{y,1}, r_{y,2}, r_{y,3}, \) and \( r_{y,4} \) where

\[
0 \leq r_{y,k} < p_{y,k} \text{ for } k = 1, 2, 3, \text{ and } 4,
\]

we can re-express \( s_y \) as:

\[
s_y = s_i - s_j
\]

\[
= \left[ p_{y,1} \times (LB) + r_{y,1} \times (d_i - d_j - t_m - t_0 - LB) \right] + \left[ p_{y,2} \times (d_i - d_j - t_m - t_0) + r_{y,2} \times (t_m + t_0) \right]
\]

\[
+ \left[ p_{y,3} \times (d_i - d_j) + r_{y,3} \times (t_m + t_0) \right] + \left[ p_{y,4} \times (d_i - d_j) + t_m + t_0 + r_{y,4} \times (UB - d_i - d_j - t_m - t_0) \right]
\]

where \( LB \) and \( UB \) are the lower and upper bounds on \( s_y \).

Similarly, \( f_y \) can be rewritten as:

\[
f_y = \left[ p_{y,1} \times 0 + r_{y,1} \times 0 \right] + \left[ p_{y,2} \times 0 + r_{y,2} \times (w_y - 0) \right] + \left[ p_{y,3} \times w_y + r_{y,3} \times (0 - w_y) \right] + \left[ p_{y,4} \times 0 + r_{y,4} \times 0 \right]
\]

Geometrically, as shown in Figure 4, \( p_{y,k} = 1 \) means \( s_y \) is within the \( k^\text{th} \) piece of \( f_y(s_y) \) and \( r_{y,k} \) indicates the ratio of \( s_y \) within the \( k^\text{th} \) piece. For a valid solution, there must be only one among the four binary variables \( p_{y,k} \) equal to 1 and only one among the four floating variables \( r_{y,k} \) greater than or equal to 0. All of the other variables are 0.

Lastly, our proposed PLP-based SER mitigation framework, for MBU-aware soft error tolerance, is formulated as:

Maximize \( \sum_{i,j,k} f_{ij,k} w_{ij,k} \text{(Candidates)} (f_{ij,k}) \)

Subject to \( (6), (7), (9), (10), \) and \( (11) \)

6. Experimental Results

In this section, we demonstrate various experiments of our proposed framework for MBU-aware soft error tolerance. The benchmark circuits are chosen from the ISCAS’89 suite. The technology used is 70nm, Predictive Technology Model (PTM). The setup \( (t_m) \) and hold \( (t_s) \) times of flip-flops are both assumed to be 10ps. The overall methodology is implemented in C++, where the piecewise linear programming formulation is solved by GNU Linear Programming Kit (GLPK) version 4.33 on a 3GHz Pentium 4 workstation running Linux.

Table 1 reports the experimental results for average MES improvement and SER reduction. For each benchmark in Table 1, we list the numbers of primary inputs, primary outputs and internal gates in column two, and the numbers of flip-flops, candidate pairs along with the corresponding percentage among all possible pairs in column three. For a circuit with \( n \) FFs, we check all possible \( (n^2) \) pairs and extract those satisfying the IM or MEP property as candidates for clock skew scheduling. The average MES values over all primary outputs before and after applying our PLP-based CSS are shown in columns five and six, for three different initial duration sizes (small: 60ps, medium: 100ps, large: 140ps). Columns seven and eight demonstrate the MES improvement and the overall SER reduction. The runtime spent on solving the PLP problem, which is not included in the table, is about 1 minute for circuits \( s1196 \) and \( s1238 \) and very few or even less than 1 second for all the others.

For example, circuit \( s208 \) has 10 primary inputs, 1 primary output, 68 internal gates, and 8 flip-flops. Among 28 \((= 8*7/2)\) pairs of FFs, 21 pairs (75%) can be identified as candidates for CSS. Based on (17), we formulate the CSS problem with these 21 pairs and then find its optimal solution by using GLPK. The MES improvements for small (60ps), medium (100ps), and large (140ps)
duration sizes are 15.9%, 35.7%, and 36.1%, respectively. When considering all possible sizes of glitches, the overall SER reduction is 29.2%. On average across all benchmarks, 35.8% SER reduction can be achieved.

Table 1 also shows the corresponding amount of skews due to CSS. This is measured by normalized absolute adjustment in clock signal, which is defined as:

\[ \frac{\sum |\Delta AT(FF_i)|}{\#FFs \cdot T_{clk}} \] (14)

where \(\Delta AT(FF_i)\) is the amount of adjustment in the arrival time of clock signal to \(FF_i\) and \(T_{clk}\) is the clock period of the circuit.

Normalized absolute adjustment (14) quantifies the cost imposed by CSS in terms of the degree of clock network modification. Intuitively, the larger the value of normalized absolute adjustment, the more aggressive modification the clock network may suffer. As it can be seen in the last column of Table 1, on average 4.4% normalized absolute adjustment is needed by our CSS-based framework. Note that the adjustment does not necessarily imply additional logic on the clock tree.

Figure 5 shows the mitigation of MBU effects during clock cycles subsequent to particle hits (SEUs). In addition to the SER reduction for the first clock cycle via CSS, the potential overall methodology using clock skew scheduling is formulated as a piecewise linear programming problem and can be solved efficiently by GLPK. Experiments on a set of ISCAS’89 benchmarks reveal the effectiveness of our framework.

7. Conclusion

In this paper, we propose an analytical method for MBU-aware soft error tolerance of sequential circuits. The approach adjusts the arrival times of clock signals such that error-latching windows of flip-flops can be overlapped, which in effect increases the probability of timing masking and decreases the soft error rate of a sequential circuit. Moreover, two types of candidate pairs of flip-flops, beneficial for MBU elimination, are introduced. The overall methodology using clock skew scheduling is formulated as a piecewise linear programming problem and can be solved efficiently by GLPK. Experiments on a set of ISCAS’89 benchmarks reveal the effectiveness of our framework.

References


Table 1. Average mean error susceptibility (MES) improvement and overall soft error rate (SER) reduction

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>s208</td>
<td>10</td>
<td>6</td>
<td>0.699</td>
<td>0.001</td>
<td>0.008</td>
<td>0.0015</td>
<td>15.59%</td>
<td>29.21%</td>
<td>5.6%</td>
</tr>
<tr>
<td>12</td>
<td>2</td>
<td>100</td>
<td>0.0049</td>
<td>0.0023</td>
<td>0.016</td>
<td>0.016</td>
<td>15.69%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>68</td>
<td>79%</td>
<td>0.0077</td>
<td>0.0049</td>
<td>0.036</td>
<td>0.036</td>
<td></td>
<td>36.05%</td>
<td></td>
</tr>
<tr>
<td>s298</td>
<td>5</td>
<td>14</td>
<td>60</td>
<td>0.0097</td>
<td>0.0071</td>
<td>0.0071</td>
<td>27.01%</td>
<td>37.79%</td>
<td>2.83%</td>
</tr>
<tr>
<td>14</td>
<td>44</td>
<td>100</td>
<td>0.0022</td>
<td>0.0014</td>
<td>0.0042</td>
<td>0.0042</td>
<td>41.27%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>49%</td>
<td>140</td>
<td>0.0064</td>
<td>0.0025</td>
<td>0.0255</td>
<td>0.0255</td>
<td>45.19%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s344</td>
<td>9</td>
<td>15</td>
<td>60</td>
<td>0.0032</td>
<td>0.0023</td>
<td>0.0023</td>
<td>27.96%</td>
<td>31.35%</td>
<td>1.29%</td>
</tr>
<tr>
<td>11</td>
<td>64</td>
<td>100</td>
<td>0.0031</td>
<td>0.0019</td>
<td>0.0093</td>
<td>0.0093</td>
<td>29.34%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>63%</td>
<td>140</td>
<td>0.0026</td>
<td>0.0018</td>
<td>0.0162</td>
<td>0.0162</td>
<td>37.69%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s400</td>
<td>3</td>
<td>21</td>
<td>60</td>
<td>0.0028</td>
<td>0.0019</td>
<td>0.0029</td>
<td>23.81%</td>
<td>38.28%</td>
<td>1.51%</td>
</tr>
<tr>
<td>2</td>
<td>36</td>
<td>100</td>
<td>0.0013</td>
<td>0.0018</td>
<td>0.0095</td>
<td>0.0095</td>
<td>27.84%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>17%</td>
<td>140</td>
<td>0.0015</td>
<td>0.0011</td>
<td>0.0113</td>
<td>0.0113</td>
<td>38.28%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s526</td>
<td>3</td>
<td>21</td>
<td>60</td>
<td>0.0020</td>
<td>0.0015</td>
<td>0.0025</td>
<td>22.15%</td>
<td>40.12%</td>
<td>6.28%</td>
</tr>
<tr>
<td>21</td>
<td>63</td>
<td>100</td>
<td>0.0018</td>
<td>0.0010</td>
<td>0.0060</td>
<td>0.0060</td>
<td>44.01%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>165</td>
<td>20%</td>
<td>140</td>
<td>0.0024</td>
<td>0.0029</td>
<td>0.0092</td>
<td>0.0092</td>
<td>54.20%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s1196</td>
<td>14</td>
<td>45</td>
<td>60</td>
<td>0.0015</td>
<td>0.0015</td>
<td>0.0015</td>
<td>18.71%</td>
<td>40.41%</td>
<td>7.21%</td>
</tr>
<tr>
<td>14</td>
<td>76</td>
<td>100</td>
<td>0.0028</td>
<td>0.0020</td>
<td>0.0128</td>
<td>0.0128</td>
<td>51.02%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>197</td>
<td>59%</td>
<td>140</td>
<td>0.0014</td>
<td>0.0011</td>
<td>0.0155</td>
<td>0.0155</td>
<td>51.53%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>s1238</td>
<td>14</td>
<td>18</td>
<td>60</td>
<td>0.0014</td>
<td>0.0011</td>
<td>0.0027</td>
<td>15.28%</td>
<td>33.31%</td>
<td>6.32%</td>
</tr>
<tr>
<td>13</td>
<td>76</td>
<td>100</td>
<td>0.0023</td>
<td>0.0020</td>
<td>0.0031</td>
<td>0.0031</td>
<td>40.23%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>540</td>
<td>50%</td>
<td>140</td>
<td>0.0010</td>
<td>0.0017</td>
<td>0.0057</td>
<td>0.0057</td>
<td>43.84%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avg.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>35.75%</td>
<td>4.44%</td>
</tr>
</tbody>
</table>

Figure 5. Mitigation of MBU effects during clock cycles subsequent to particle hits (SEUs) s208 (see Figure 5(a)) and s298 (see Figure 5(b)) can be mitigated by 43% and 63%, respectively.