Optimized Flip-Flop Selection and Gate Sizing to Reduce Re-Execution Penalty Due to Soft Errors

Warin Sootkaneung[1*] & Kewal K. Saluja[2]

[1] Rajamangala University of Technology Phra Nakhon, Faculty of Engineering, Department of Computer Engineering, Bangkok, Thailand

[2] University of Wisconsin-Madison, Department of Electrical and Computer Engineering, Madison, WI, USA

Abstract : Particle strike induced soft errors are becoming one of the most challenging reliability issues, as device geometries are decreasing to low nanometers. Circuit designers are actively researching and developing soft error mitigation techniques that provide satisfactory improvement with reasonable performance overheads. This work introduces a novel technique to reduce soft error rate in sequential circuits. First, flip-flop selection for soft error detection is formulated as a binary integer linear programing (BILP) problem. This approach ensures that maximum soft error detection can be achieved for a given number of selected flip-flops. To improve the efficacy in CPU runtime and memory usage of the optimization based technique, a heuristic based approach for flip-flop selection was proposed. For all experimental circuits, the proposed heuristic solutions achieved the same target as the optimal solutions while requiring very small computational resources. Finally, a combined flip-flop selection and gate sizing technique is proposed. To assess the performance of this technique, a "re-execution penalty index" was defined to represent the re-execution penalty required for soft error recovery. This metric uses the contributions of SER reduction gains from flip-flop selection and sizing to overall circuit reliability improvement. The re-execution penalty index was used to identify the best distribution of area overhead assigned to flip-flops and logic gates.

Keywords : flip-flop selection; gate sizing; re-execution penalty; soft error

*Corresponding Author : warin.s@rmutp.ac.th Tel. +662-282-9009-15 ext. 7176

1. Introduction

When extraterrestrial neutrons carried by cosmic rays or alpha particles from impure package strike a tiny MOS device, a nuclear reaction in silicon substrate can produce charge deposition near its reverse-biased junction. If the amount of charge deposition is sufficiently large, it can cause a transient current inducing a single event transient (SET) flowing forward through the junction even if the device is in “off” state [HYPERLINK \l "Mav02" 1]. Consequently, in a digital circuit, if the SET can propagate to primary output or appear within latching window of a flip-flop, a miscalculation due to bit flips happens. In other words, this circuit suffers a soft error. Nowadays, while high-energy alpha particles are significantly reduced as a result of the advances in packaging development2], decreased node capacitance due to technology scaling intensifies neutron-induced soft error vulnerability of digital circuits. This makes soft error caused by neutron strikes one of the most challenging reliability issues for circuit design in nanometer generation.

As soft error rate (SER) in modern processors is steadily increasing, the circuit designers are diligently investigating for efficient SER reduction techniques. Recently, many approaches to harden storage and combinational components against soft errors have been proposed. Works in [HYPERLINK \l "Pau11" 3],4], and [HYPERLINK \l "Rev12" 5] use error correcting code (ECC) based techniques to reduce SER in memories and register files. Moreover, compiler and microarchitecture approaches6], [HYPERLINK \l "Lee10" 7] are friendly with register files since no power overhead from additional hardware is required. On the other hand, hardware based techniques are generally used to mitigate soft error in combinational parts of processors. Various techniques to protect combinational circuits against soft error that have been developed include gate sizing8], [HYPERLINK \l "She09" 9],10], [HYPERLINK \l "Soo11" 11]; flip-flop hardening12], [HYPERLINK \l "Mit10" 13]; and flip-flop selection14], [HYPERLINK \l "Hil08" 15]. Selectively sizing sensitive gates can reduce the probability of soft error generation, yet, in general, requires large power and area overheads. Protecting and detecting soft errors at flip-flops can achieve superior SER reduction but this approach significantly impacts timing performance and clock tree network complexity of the design14].

In this study, we investigate soft errors in sequential circuits and develop two techniques to improve circuit immunity against soft errors in gate level. The main contributions of this paper are as follows. First, we introduce a novel optimization formulation for selecting candidate flip-flops which can maximize soft error coverage. Second, we propose a heuristic soft error reduction technique based on the combination of sizing and flip-flop selection approaches. In this technique, we define a new metric called re-execution penalty index as a function of SER to indicate overall timing penalty required to resume the operation after appearance of a soft error. The idea behind this defining this metric is the fact that if a soft error is not detected at lower level (gate level) the recovery mechanism at higher level of abstraction will consume larger performance resource. We must pay more attention to those undetected soft errors in which sizing based approach is the only solution. Hence, we define the re-execution penalty index to give higher weight to undetected SER

than detected SER. We believe that this new metric can fairly address the relative soft error immunity yields from flip-flop selection and sizing approaches. In addition, for the combined flip-flop selection and sizing technique, a heuristic flip-flop selection is developed to improve CPU runtime and memory usage requirement of the first proposed method for solving the optimization problem.

This paper is organized as follows. Section 2 describes some related theories. Section 3 introduces the proposed optimization based technique for flip-flop selection to maximize soft error detection. Section 4 defines the re-execution metric and discusses the combined flip-flop selection and sizing technique for reducing the overall re-execution penalty. Section 5 provides the experimental results and we finally conclude this work in Section 6.

2. Related Theories

A. Soft Error Model

For CMOS gate, when neutron particles strike around body-drain junction of a transistor, a current glitch or SET as a result of charge deposition generation flows into the device's drain. This transient current can be modeled as a single exponential current source as shown in (1) [HYPERLINK \l "Haz00" 16],17].

$$I(t) = \frac{Q}{\tau}\sqrt{\frac{t}{\Gamma}}e^{-t/\tau} \quad (1)$$

In the equation above, Q is the amount of charge deposition and τ is the charge collection time constant. If a

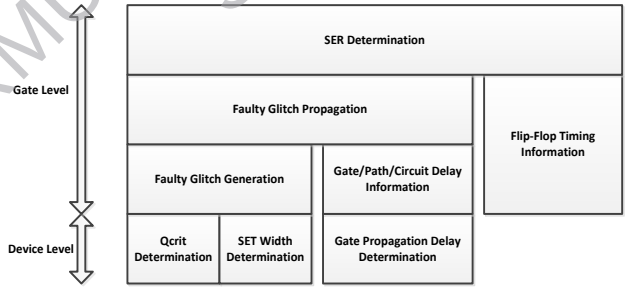

Figure 1. Soft error simulation framework

strike can complement the output logic of a gate or causes a change in gate output voltage greater than Vdd/2, the charge deposition Q is defined as the critical charge (Q_{crit}).The Q_{crit} of each gate relates to the amount of atmospheric neutron energy which brings the gate to faulty state [HYPERLINK \l "Mav02" 1]. Due to technology scaling, Q_{crit} is steadily decreasing, and this means

that the device cannot tolerate low energy neutrons which has high rate of strike. In addition, small amount of Q_{crit} extends the width of SET and hence, this reduces electrical masking probability when a SET is propagating to the circuit output. The larger width of the faulty glitch also increases the probability that the SET can appear within the latching window of a flip-flop which consequently stores an incorrect value. For the reasons explained above, soft error tolerance in nanometer circuits is sharply degrading. The information of atmospheric neutron was recorded in the JEDEC89A standard18]. The study in [HYPERLINK \l "Deo05" 19] used this standard to map the charge deposition due to neutron strike to the probability that the strike with corresponding energy can be found. During our simulation, we use this probability of strike to weight randomly generated charge deposition for each faulty glitch injection. We assume that a transient current from the strike is totally masked if the amount of charge deposition of the induced SET is smaller than Q_{crit}. The framework for soft error simulation in this work is illustrated in Figure 1. The simulation procedures involve device and gate levels of the design. In device level, we performed SPICE simulation to determine Q_{crit}, SET width, and propagation delay for each mapping gate in the library. After that our Java based gate level simulator was used to estimate overall SER of the circuit. Gate level simulation includes faulty glitch generation and propagation, and SER estimation. Further, gate/path/circuit delay and flip-flop timing information is also taken into account to capture the electrical and timing window masking probabilities of the circuit. The detailed process of the gate level simulation is discussed as follows. The flowchart in Figure 2 shows our gate level simulation methodology. To begin with, we randomly assign input to a circuit for each cycle. Then, a SET is injected to the output of each gate. The value of charge deposition of the SET is weighted using the probability of atmospheric neutron strike as discussed before, and the time of the strike is randomly assigned within the circuit cycle time. Next, we check if the amount of charge deposition of any SET is greater than the Q_{crit} of that gate. If it is, we acquire corresponding SET pulse width from SPICE simulation and continue to propagate the faulty glitch. During fault propagation, in addition to performing logic simulation, we take into consideration the propagation delay of each gate obtained from the previous SPICE simulation to update timing characteristic of the propagating glitch(es) along the sensitized path(s). However, apart from being logically masked, any glitch that has time duration smaller than gate propagation delay is considered to disappear from gate output. Finally, circuit SER can be estimated based on the appearance of glitches at primary output or within the latching window of a flip-flop during simulating time frame. We assume that only single node is hit once at a time. Thus, using equation (2), we can obtain the circuit SER.

$$circuit\ SER = \frac{\sum_{i=1}^{i=k*\#gates} E_i * w_i}{k * cycle\ time} \quad (2)$$

In (2), E_i is 0 if the injected SET does not cause a soft error, and it is 1 if the soft error appears at an output or flip-flop, w_i is the ratio of active area of the gate being hit to the circuit area, and k is the number of total cycles we simulate.

B. Prior Studies

There are numerous previously proposed SER mitigation techniques that reconfigure latches/flip-flops to protect combinational parts of a circuit against soft errors or other types of transient errors. Two major mechanisms used in these

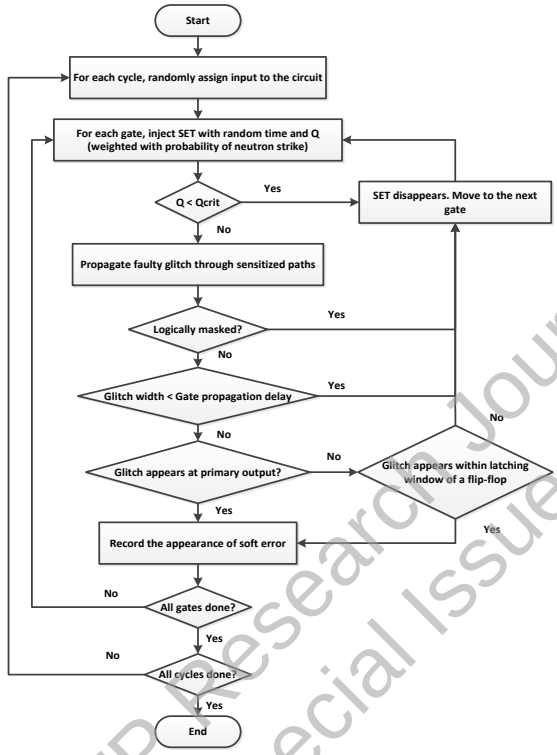

Figure 2. Flowchart of gate level soft error simulation

techniques include error correction12], [HYPERLINK \l "Mit10" 13],14], and error detection [HYPERLINK \l "Hil08" 15],20]. The flip-flop which is hardened for error correction purpose requires additional hardware for keeping the correct value, or for changing its timing behavior to filter undesired transient glitches. This may significantly impact delay performance of the circuit. On the other hand, flip-flops designed to detect the error require less hardware overhead, yet the circuit needs to re-execute the task after the soft error is detected.

Another approach to handle soft errors in combinational logic circuits focuses on selectively upsize most vulnerable gates [HYPERLINK \l "Zho06" 8],9], [HYPERLINK \l "Sooss" 10],11]. This approach directly enhances soft error immunity at the source of SET by increasing the amount of Q_{crit} of the victim devices. As a result, the probability of SET generation decreases. However, for very small technologies, trading SER reduction outcomes with area and delay overheads required in sizing techniques still need to be improved.

Our work proposes a flip-flop selection, and a combination of flip-flop selection and sizing techniques to reduce SER. For flip-flop related treatment, we employ the detection scheme as used in [HYPERLINK \l "Hil08" 15]. Yet, while the work in15] uses greedy methodology to select flip-flops, our proposed solution is optimization based which can maximize soft error coverage for given number of selected flip-flops. The work in [HYPERLINK \l "Rao06" 14] proposes a technique which integrates flip-flop selection with sizing. The flip-flop selection algorithm in14] uses the slack available at flip-flops and hardens them with additional delay overhead. Therefore, it may yield little or no improvement for circuits where most flip-flops are in long paths. Also, the method of [14] is unable to directly address the impact of soft errors that are not captured at selected flip-flops. In this work, our technique addresses the soft error-related reliability degradation with the re-execution penalty index. We believe that this common metric treats two SER reduction gains from two different techniques fairly.

2.1 Optimal Solution for Flip-Flop Selection

In this section, a binary integer linear programing (BILP) based optimization approach for flip-flop selection is introduced. This technique guarantees the best soft error coverage obtainable by flip-flops.

The BILP formulation consists of solving for two vectors of binary variables. The first vector **a** = [a_1, a_2, a_3... a_n], $a_j \in \{0, 1\}$ is called a vector of variables of selection. A variable a_j indicates whether the corresponding flip-flop j is selected (value of 1) or not (value of 0). The second vector **v** = [v_1, v_2, v_3... v_m], $v_i \in \{0, 1\}$ is related to faults that are candidate set of soft errors**.** If the Fault-i is detected, then the corresponding variable v_i is assigned a value of 1 and if this fault is not detected, then the variable is assigned a value of 0.

We also construct a fault coverage table as shown in Figure 3. For a circuit with m total faults and n flip-flops, this table records information of all faults, including the weight of each fault, and flip-flops where the fault can be detected. In the table, each component b_{ij} is a binary constant. We set b_{ij} =1 if Fault-i is stored in (or can be detected by) flip-flop j, and b_{ij} = 0 otherwise. The last constant is the weighting factor w_i of corresponding fault i. For some faults that can be detected by the same set of flip-flops, we reckon all of them as one combined fault i with its w_i equal to the sum of weighting factor of each fault. The value of w_i is the ratio of active area to the circuit area and it is also used in (2) in the previous section.

For example, in Figure 3, Fault-3 can be detected at FF-2 and FF-3 but not at FF-1. The corresponding weighting factor of Fault-3 is equal to 0.13. All constants in fault coverage table can be obtained during gate level simulation. The variables in **a** and **v** will be solved by BILP optimizer. The proposed BILP formulation is as follows.

We define the number of flip-flops, l, as a constraint and the objective is to maximize the number of soft errors detected at these flip-flops with appropriate weights. These conditions are stated as a constraint function shown in (3) below:

$$\sum_{j=1}^{n} a_j = l \quad (3)$$

and an objective function in the BILP formulation for flip-flop selection expressed in (4) below.

$$max \sum_{i=1}^{m} w_i v_i \quad (4)$$

	FF-1	FF-2	FF-3	…	FF-*j*	…	FF-*n*	Weight
Fault-1	b_{11}	b_{12}	b_{13}	...	b_{1j}	…	b_{1n}	w_1
Fault-2	b_{21}	b_{22}	b_{23}	...	b_{2j}	…	b_{2n}	w_2
Fault-3	0	1	1	...	b_{3j}	…	b_{3n}	0.13
Fault-*i*	b_{i1}	b_{i2}	b_{i3}	...	b_{ij}	…	b_{in}	w_i
:	:	:	:	:	:	:	:	:
Fault-*m*	b_{m1}	b_{m2}	b_{m3}	...	b_{mj}	…	b_{mn}	w_m

Figure 3. Fault coverage table

For fault detection mechanism, a Fault-i is detected (v_i is set to be 1) when it appears in one of the flip-flops which is selected (one of the b_{ij} components is equal to 1 while a_j of the corresponding flip-flop j is also equal to 1). As a result, this condition can be written as a set of the constraints given in (5) and (6).

$$\forall j: b_{ij} a_j \leq v_i \quad (5)$$

$$\sum_{i=1}^{m} b_{ij} a_j \geq v_i \quad (6)$$

We use MATLAB optimization toolbox [HYPERLINK \l "Mat" 21] to solve the BILP problem as expressed in (3)-(6). The results of applying this technique on a number of benchmark circuits are reported and discussed in Section 5.

2.2 Combined Flip-Flop Selection and Sizing Technique

This section first discusses the re-execution penalty index used as a metric to justify the performance of our proposed technique. Next, the SER reduction that combines flip-flop selection and sizing is introduced.

C. Re-execution penalty index

We believe that a fault that is not detected at lower level of abstraction may potentially bring severe penalty to a system to resume the operation at higher level. This performance penalty can be considered as re-execution requirement. For a hardened circuit against soft errors using error detection scheme, a re-execution penalty index P is defined as a function which combines the detected soft error ($SER_{removed}$) with the undetected soft error ($SER_{not\text{-}removed}$) as shown in (7). Clearly, there is no penalty associated with faults that are masked.

$$P = K_{removed} SER_{\_\_} + K_{not-removed} SER_{not-removed} \quad (7)$$

In equation (7), $K_{removed}$ and $K_{not-removed}$ are constants which vary from circuit to circuit. Although addressing both types of soft errors requires re-execution, soft errors that cannot be detected ($SER_{not-removed}$) contribute more substantially to the total performance loss or the defined re-execution penalty index, compared to those which are detected. As a result $K_{removed}$ is smaller than $K_{not-removed}$. For comparison purpose, the re-execution penalty index after hardening is normalized with respect to the original circuit condition. If P_0 is the re-execution penalty index of a circuit at the original design with SER_0 as the original SER, the normalized re-execution penalty index can be written as (8)

$$\frac{P}{P_0} = \frac{1}{SER_0}\left(\frac{K_{\_\_\_\_}}{K_{not-removed}} SER_{removed} + SER_{not-removed}\right) \quad (8)$$

D. Combination of flip-flop selection and sizing technique

We integrate flip-flop selection and sizing approaches together to improve our ability to handle soft errors. This combined technique shares the area overhead between selected flip-flops and sensitive gates. The best distribution of area overhead to flip-flop and logic gate parts is finally chosen based on the reduction in normalized re-execution penalty index of a circuit.

```
For each injected fault
{
    If fault appears at primary output or flip-flop(s)
Record fault's weight (wi), gate that generates the fault, and flip-
        flops that capture the fault;
}
For each selected FF flip-flop {
    Select a flip-flop with the largest sum of weight of each fault;
    Remove the detected faults from other flip-flops and primary output;
}
Update sensitivity of all gates;
```

Figure 4. Heuristic flip-flop selection pseudocode

For each round of treatment, flip-flop selection is performed first. For small circuits, we can use the optimization approach to select candidate flip-flops as discussed before, yet solving the BILP problem for large circuits require

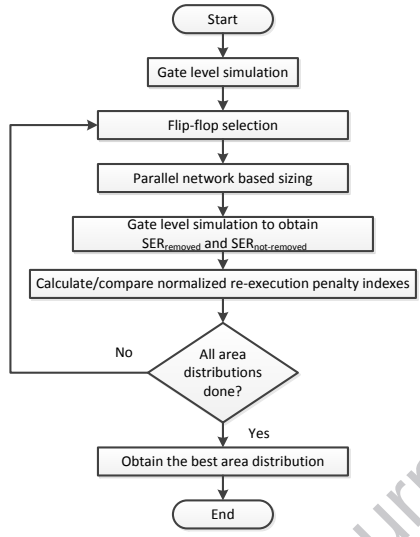

Figure 5. Flowchart of the combined flip-flop selection and sizing technique

significant computation time. Hence, for large circuits, we use a heuristic with greedy approach to select flip-flops which offers large amount of soft error coverage as shown in Figure 4. This flip-flop selection approach can be explained as follows. For each injected fault, we record fault information including weighting factor, gate that is the source of the soft error, and flip-flops that capture the fault. After all faults are injected, we search for a flip-flop with the largest sum of fault weight and add it to the set of selected flip-flops. Then, we remove the detected faults from primary outputs and other flip-flops. We repeatedly search for a flip-flop with the largest fault coverage one at time until the number of selected flip-flops is satisfied. After we completely identify all selected flip-flops, the sensitivity to soft error of each gate is updated to be used in the next step.

In the next step, we apply sizing technique to further reduce $SER_{not\text{-}removed}$ of a circuit. The part of the area overhead which is not used for flip-flops is distributed to most sensitive gates using parallel network sizing based technique11]. The sizing technique proposed in [HYPERLINK \l "Soo11" 11] includes the weighted area distribution algorithm which fairly assigns additional area to gates based on their sensitivity by providing larger area to more sensitive gates. In addition, like the work in11] we use the SER saturation consideration algorithm which limits an increase in SER of a gate when its parallel network is over upsized. These two algorithms proposed in [HYPERLINK \l "Soo11" 11] provided impressive SER reduction for some 32-nm experimental circuits. We recommend the readers to read11] for detailed understanding of the sizing approach adopted in this paper. After candidate flip-flops are selected and additional area distributed to most sensitive gates, we perform gate level simulation (as illustrated in Figure 2) on the hardened circuit again to obtain the $SER_{removed}$ and $SER_{not\text{-}removed}$. Information from this simulation is used to evaluate soft error mitigation yield of our method.

We use (8) to evaluate the improvement by the methods proposed in this paper. For a given circuit, various area distributions (related to the number of selected flip-flops and area given to sensitive gates) may be investigated. Therefore, we can compare the normalized re-execution penalty indexes for different area distributions, and identify the best distribution providing the smallest normalized re-execution penalty index. The flowchart of combined flip-flop selection and sizing technique is illustrated in Figure 5. We provide experimental results of our proposed re-execution penalty reduction in the next section.

3. Experimental Results

To assess the soft error mitigation using our methods and to compare it with other methods, we selected a number of benchmark circuits from ISCAS'89 to be our experimental circuits. All circuits are mapped with 32-nm predictive technology model from [HYPERLINK \l "HSP" 21]. The gate cell library used for all circuits consists of 2-, 3-, and 4-input NAND and NOR gates; inverters; and flip-flops. During gate level simulation, 100,000 random input vectors are generated at the beginning of the cycle time for each circuit. The results of SER and re-execution penalty index are reported in normalized form with respect to the original design.

TABLE I. AREA OVERHEAD AND NORMALIZED SER FROM OPTIMAL AND HEURISTIC FLIP-FLOP SELECTION

Fraction of selected flip-flops = 0.4

Circuit	*#Selected flip-flops*	*%Area overhead*	*Optimal* $SER_{removed}$ [this work]	*Heuristic* $SER_{removed}$ [this work]	*Heuristic* $SER_{removed}$ [15]
S298	6	4.58	0.5265	0.5265	0.5188
S344	6	4.60	0.4691	0.4691	0.4690
S526	8	3.32	0.4752	0.4752	0.4604
S838	13	4.31	0.2500	0.2500	0.2499
S1196	7	1.49	0.1985	0.1985	0.1985

Fraction of selected flip-flops = 0.6

Circuit	*#Selected flip-flops*	*%Area overhead*	*Optimal* $SER_{removed}$ [this work]	*Heuristic* $SER_{removed}$ [this work]	*Heuristic* $SER_{removed}$ [15]
S298	8	6.11	0.6412	0.6412	0.6342
S344	9	6.91	0.6172	0.6172	0.6172
S526	13	6.56	0.6881	0.6881	0.6854
S838	19	6.30	0.3467	0.3467	0.3456
S1196	11	8.64	0.2719	0.2719	0.2719

Fraction of selected flip-flops = 0.8

Circuit	*#Selected flip-flops*	*%Area overhead*	*Optimal* $SER_{removed}$ [this work]	*Heuristic* $SER_{removed}$ [this work]	*Heuristic* $SER_{removed}$ [15]
S298	11	8.40	0.8018	0.8018	0.8004
S344	12	9.21	0.7525	0.7525	0.7525
S526	17	7.07	0.8343	0.8343	0.8322
S838	26	8.62	0.4339	0.4339	0.4339
S1196	14	2.98	0.3166	0.3166	0.3166

TABLE II. COMBINED FLIP-FLOP SELECTION AND SIZING RESULTS

All flip-flops selected without sizing

Circuit	*#Selected FFs*	*%Area overhead*	*Not removed SER*	*Removed SER*	*Re-execution penalty index* k^*=1.5	*k*=2	*k*=2.5
S298	14	10.69	0.094	0.906	0.698	0.547	0.456
S344	15	11.51	0.147	0.853	0.716	0.574	0.488
S526	21	8.73	0.047	0.953	0.682	0.524	0.428
S838	32	10.61	0.585	0.416	0.862	0.792	0.751
S1196	18	3.82	0.553	0.447	0.851	0.776	0.732
S1494	6	0.82	0.779	0.221	0.926	0.889	0.867
S5378	179	8.99	0.326	0.674	0.775	0.663	0.595

Fraction of selected flip-flops = 0.8 with sizing

Circuit	*#Selected FFs*	*%Area overhead*	*Not removed SER*	*Removed SER*	*Re-execution penalty index* *k*=1.5	*k*=2	*k*=2.5
S298	11	10.69	0.097	0.783	0.619	0.489	0.411
S344	12	11.51	0.188	0.661	0.628	0.518	0.452
S526	17	8.73	0.085	0.789	0.611	0.479	0.400
S838	26	10.61	0.615	0.303	0.817	0.766	0.736
S1196	14	3.82	0.555	0.378	0.807	0.744	0.706
S1494	5	0.82	0.798	0.192	0.926	0.894	0.875
S5378	143	8.99	0.269	0.585	0.659	0.562	0.503

Fraction of selected flip-flops = 0.6 with sizing

Circuit	*#Selected FFs*	*%Area overhead*	*Not removed SER*	*Removed SER*	*Re-execution penalty index* *k*=1.5	*k*=2	*k*=2.5
S298	8	10.69	0.100	0.721	0.580	0.460	0.388
S344	9	11.51	0.274	0.581	0.661	0.564	0.506
S526	13	8.73	0.181	0.655	0.618	0.509	0.443
S838	19	10.61	0.732	0.249	0.898	0.857	0.832
S1196	11	3.82	0.585	0.337	0.809	0.753	0.719
S1494	4	0.82	0.821	0.165	0.931	0.904	0.887
S5378	107	8.99	0.265	0.562	0.640	0.546	0.490

Fraction of selected flip-flops = 0.4 with sizing

Circuit	*#Selected FFs*	*%Area overhead*	*Not removed SER*	*Removed SER*	*Re-execution penalty index* *k*=1.5	*k*=2	*k*=2.5
S298	6	10.69	0.145	0.648	0.576	0.468	0.404
S344	6	11.51	0.314	0.463	0.623	0.546	0.499
S526	8	8.73	0.331	0.458	0.636	0.560	0.514
S838	13	10.61	0.877	0.187	1.000	0.970	0.951
S1196	7	3.62	0.663	0.254	0.832	0.790	0.765
S1494	2	0.82	0.907	0.091	0.967	0.952	0.943
S5378	72	8.99	0.280	0.525	0.630	0.543	0.490

$$\frac{s}{k} = \frac{K_{removed}}{K_{not-removed}}$$

E. Results from optimal flip-flop selection

Table I shows the results from the BILP flip-flop selection algorithm based on equations (3)-(6). In Table I, we also compare the normalized $SER_{removed}$ of our optimization formulation with our proposed heuristic solution and the method proposed in15] for different fractions of selected flip-flops. Interestingly, we notice from Table I that our heuristic flip-flop selection achieves exactly the same results as the optimal solution for each experimental circuit. On the other hand our methods (heuristic as well as optimal) outperform the method proposed in [HYPERLINK \l "Hil08" 15]. However, the runtime of the optimal method, though not reported in the table, is an order of magnitude larger and also for very large circuits, the optimal method could not complete execution even in a very long time. Though optimality of the proposed heuristic flip-flop selection method

cannot be guaranteed, we choose our proposed heuristic flip-flop selection method for use in further experiments reported in the next subsection for its high efficiency in CPU runtime and memory usage compared to solving BILP problems.

F. Results from combined flip-flop selection and sizing technique

Table II shows the results from our combined flip-flop selection and sizing for improving the re-execution penalty. For all experiments, we vary the fractions of selected flip-flops and the area distributed to logic gates, while we set the area overhead of each experimental circuit to be the same as the area required for selecting all the flip-flops in the circuit without upsizing combinational logic gates (as reported in the topmost section of Table II). For example, for the circuit s526, if all flip-flops were selected, then the total area overhead would be 8.73%. We set this limit for the overhead of s526 in the remaining experiments. With different values of k, the ratio of $K_{removed}$ to $K_{not\text{-}removed}$, the normalized re-execution penalty index yields of various experimental circuits are reported in Table II. It can be seen from this table that when all flip-flops are selected, the $SER_{not\text{-}removed}$ for all circuits decreases significantly; e.g., the $SER_{not\text{-}removed}$ of the circuit S526 is reduced by 95%. However, if we consider the values of normalized re-execution penalty index of the circuit S526, this circuit receives the largest improvement when we apply flip-flop selection with 0.8 fractions of total flip-flops and upsizing the most sensitive gates with the rest of area budget. For other circuits, the largest decrease in re-execution penalty index can be altered depending on the value of k. The best flip-flop selection and sizing combinations are chosen such that it provides the smallest value of re-execution penalty index.

4. Conclusion

This paper proposes two novel techniques to mitigate soft errors in sequential circuits. First, we introduce a BILP formulation for soft error coverage maximization by selecting some flip-flops to be as soft error detectors. This technique guarantees that for a given number of selected flip-flops, maximum soft error detection is accomplished, though it requires large CPU runtime and memory usage. Second, a combined technique of flip-flop selection and gate sizing is developed. In this method, we propose a heuristic approach for flip-flop selection to reduce computational resource consumption. Experimental results show that our heuristic flip-flop selection outperforms a previously proposed method and reaches the optimality for all experimental circuits. In addition, we define a re-execution penalty index which represents the re-execution penalty required to recover a soft error. We use the re-execution penalty index to guide our simulator to assign appropriate area overhead to flip-flops and logic gates. Our experimental results show that each circuit receives the best area overhead distribution when its re-execution penalty index is minimized.

5. Acknowledgements

This work is supported by Faculty of Technical Education, Rajamangala University of Technology Phra Nakhon.

6. References

[1] D. G. Mavis and P. H. Eaton, "Soft Error Rate Mitigation Techniques for Modern Microcircuits," in *Proc. of the 40th International Reliability Physics Symposium*, Dallas, Texas, 2002, pp. 216-225.

[2] F. Wang and V. D. Agrawal, "Single Event Upset: An Embedded Tutorial," in *Proc. of the 21st International Conference on VLSI Design*, Hyderabad, India, 2008, pp. 429-434.

[3] S. Paul, F. Cai, X. Zhang, and S. Bhunia, "Reliability-Driven ECC Allocation for Multiple Bit Error Resilience in Processor Cache," *IEEE Transactions on Computers*, vol. 60, no. 1, pp. 20-34, January 2011.

[4] H. Amrouch and J. Henkel, "Self-Immunity Technique to Improve Register File Integrity against Soft Errors," in *Proc. of the 24th International Conference on VLSI Design*, Chennai, India, 2011, pp. 189-194.

[5] P. Reviriego, M. Flanagan, and J. A. Maestro, "A (64,45) Triple Error Correction Code for Memory Applications," *IEEE Transactions on Device and Material Reliability*, vol. 12, no. 1, pp. 101-106, March 2012.

[6] J. Hu, S. Wang, and S. G. Ziavras, "On the Exploitation of Narrow-Width Values for Improving Register File Reliability," *IEEE Transactions on Very Large Scale Integration Systems*, vol. 17, no. 7, pp. 953-963, July 2009.

[7] J. Lee and A. Shrivastava, "A Compiler-Microarchitecture Hybrid Approach to Soft Error Reduction for Register Files," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 29, no. 7, pp. 1018-1027, July 2010.

[8] Q. Zhou and K. Mohanram, "Gate Sizing to Radiation Harden Combinational Logic," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and System*, vol. 25, no. 1, pp. 155-166, January 2006.

[9] W. Sheng, L. Xiao, and Z. Mao, "Soft Error Optimization of Standard Cell Circuits Based on Gate Sizing and Multi-Objective Genetic Algorithm," in *Proc. of the 46th Design Automation Conference (DAC)*, San Francisco, CA, 2009, pp. 502-507.

[10] W. Sootkaneung and K. K. Saluja, "On Techniques for Handling Soft Errors in Digital Circuits," in *Proc. of the International Test Conference (ITC)*, Austin, TX, 2010, pp. 1-9, paper 25.2.

[11] W. Sootkaneung and K. K. Saluja, "Soft Error Reduction through Gate Input Dependent Weighted Sizing in Combinational Circuits," in *Proc. of the 12th International Symposium on Quality Electronic Design (ISQED)*, Santa Clara, CA, 2011, pp. 603-610.

[12] M. Chen and A. Orailoglu, "Flip-flop Hardening and Selection for Soft Error and Delay Fault Resilience," in *Proc. of the 24th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems*, Chicago, IL, 2009, pp. 49-57.

[13] S. Mitra, "Robust System Design," in *Proc. of the 23rd International Conference on VLSI Design*, Bangalore, India, 2010, pp. 434-439.

[14] R. R. Rao, D. Blaauw, and D. Sylvester, "Soft Error Reduction in Combinational Logic Using Gate Resizing and Flipflop Selection," in *Proc. of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, San Jose, CA, 2006, pp. 502-509.

[15] E. L. Hill, M. H. Lipasti, and K. K. Saluja, "An Accurate Flip-Flop Selection Technique for Reducing Logic SER," in *Proc. of the International Conference on Dependable Systems and Networks (DSN)*, Anchorage, AK, 2008, pp. 128-136.

[16] P. Hazucha and C. Svensson, "Impact of CMOS Technology Scaling on the Atmospheric Neutron Soft Error Rate," *IEEE Transactions on Nuclear Science*, vol. 47, no. 6, pp. 2586-2594, December 2000.

[17] P. Shivakumar, M. Kistler, S. W. Keckler, D. Burger, and L. Alvisi, "Modeling the Effect of Technology Trends on the Soft Error Rate of Combinational Logic," in *Proc. of the International Conference on Dependable Systems and Networks (DSN)*, Bethesda, MD, 2002, pp. 389-398.

[18] JEDEC89A Standard, "Measurement and Reporting of Alpha Particles and Terrestrial Cosmic Ray-Induced Soft Errors in Semiconductor Devices," Joint Electron Device Engineering Council, Solid State Technology Association, 2006.

[19] H. S. Deogun, D. Sylvester, and D. Blaauw, "Gate-Level Mitigation Techniques for Neutron-Induced Soft Error Rate," in *Proc. of the 6th International Symposium on Quality of Electronic Design (ISQED)*, San Jose, CA, 2005, pp. 175-180.

[20] D. Ernst et al., "Razor: Circuit-Level Correction of Timing Errors for Low-Power Operation," *IEEE Micro*, vol. 24, no. 6, pp. 10-20, November-December 2004.

[21] Mathworks website. [Online]. Available: http://www.mathworks.com.

[22] HSPICE PTM website. [Online]. Available: http://www.eas.asu.edu/~ptm.