Exploiting Data-Dependence and Flip-Flop Asymmetry for Zero-Overhead System Soft Error Mitigation

Liangzhen Lai
ARM Research
Email: liangzhen.lai@arm.com

Vikas Chandra
ARM Research
Email: vikas.chandra@arm.com

***Abstract*—Soft error is one of the major threats for resilient computing. Unlike SRAM soft error, which can be effectively protected by ECC, Flip-Flop soft error protection can be costly. We observe that flip-flops/latches can have asymmetric soft error rates when storing different logic values. This asymmetry can be used in conjunction with the different signal probabilities of registers in a design. In this work, we first demonstrate that flip-flop cells can be designed to have different soft error rates when storing different logic values. We also propose a methodology to match registers in a design with the flip-flop cells that minimize the soft error rates. Experimental results on commercial processor show that, with only flip-flop layout changes, our proposed scheme can reduce system SER by 16% with no overhead in performance, power and area. The system SER reduction can be improved to 48% with schematic changes and 6.7% average increase in flip-flop area.**

I. Introduction

Radiation-induced soft error has become one of the major threats for resilient computing. Soft error happens when energetic particles strike silicon substrate and alter the circuit logic states. This can result in corrupting the correct execution of the circuits. Soft error can occur in both combinational and sequential circuits [4]. Due to various masking effects, combinational circuits are less susceptible to soft errors [12], [13]. Sequential circuit elements, including SRAM bit-cell, latch, flip-flop (FF) contribute to the most of soft errors [2].

Because of its small size and storage node capacitance, individual SRAM bit-cell is very vulnerable to soft errors [1]. However, coding techniques, such as parity and Error-Correcting Code (ECC), are both effective and efficient in protecting SRAM bit-cell array from soft errors. With technology scaling, the soft error rate (SER) of FF becomes comparable to bit-cells at the same technology node [14]. As opposed to the regular and centralized layout of SRAM bit-cell arrays, FFs are usually distributed across the entire design. This makes it very difficult and costly to use the same coding techniques to protect FF soft errors.

Design hardening [8], [10], [11] are the most common techniques for FF soft error mitigation. These hardening techniques can be very effective in reducing FF SER rates. But, they usually come with significant (e.g., 1.5-3X) area overhead. Other than design techniques, runtime adaptation based on power management mechanisms [15], [9] can also be used to reduce the system SER.

This work is motivated by two observations: 1) FFs can be designed with asymmetric SER when holding different logic values (0 and 1); 2) FFs in a design can have different probabilities of holding the two logic values. We proposed an approach to mitigate soft error by designing FFs with different SER asymmetry and using them to implement different registers in the circuits, including control and pipeline registers. We demonstrate that the approach can be zero-overhead in performance, power and area. With minimal area increase, we can design FFs with larger SER asymmetry to achieve further system SER reduction.

The rest of the paper is organized as follows. Section II introduces some background on soft error and soft error sensitivity. Section III gives an overview of the proposed methodology. Section IV demonstrates different methods to design FFs with asymmetric SER rates. Section V describes the methodology for matching asymmetric FFs. Experimental results and discussions are presented in Section VI, and Section VII concludes the paper.

II. Background

A. Radiation-Induced Soft Error

When energetic particles strike the silicon substrate, electron and hole pairs are created, and the charge can be collected by PN junctions. The collected charge can alter the circuit state, causing radiation-induced soft error.

The SER depends on a lot of different factors, including technology factors and design factors. The technology factors include the manufacturing process and device materials. For example, the diffusion can determine how much charge will be collected. The design factors can include circuit topology and circuit layout. For example, Dual Interlocked Cell (DICE) design adds redundancy in the number of storage nodes to improve the resilience against soft errors. Meanwhile, layout style can also affect the SER of a cell. For example, the junction size determines the area that is susceptible to collected charge.

In CMOS circuits, different circuit junctions can also have different sensitivities to the collected charge [8]. For example, in an inverter, the collected charge at the drain diffusion of the ON transistor can be discharged through the channel. So the transistor is less sensitive than the OFF transistor where the collected charge can have higher impact on the inverter output.

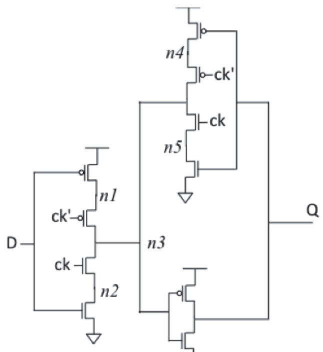
B. Soft Error Sensitivity Analysis

The basic storage element in logic circuits is a latch. Though FF-based designs are more popular, the storage element in a typical master-slave FF is still a latch. Depending on the clock state, the data is stored at either the master latch or the slave latch. Therefore, we will use latch as the example circuit to analyze the soft error sensitivity.

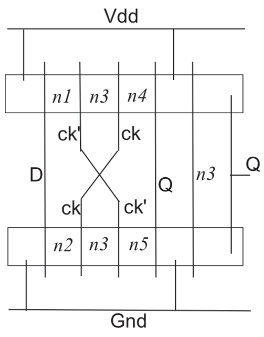
An example of a latch design [5] is shown in Fig. 1(a). When the latch is storing logic value 1, i.e., Q=1, the pull-up part of the inverter and the pull-down part of the feedback tri-state inverter (i.e., diffusion at node n3 and n5) are the sensitive nodes/junctions. Meanwhile, when the latch stores logic value 0, i.e., Q=0, the pull-down part of the inverter and the pull-up part of the feedback tri-state inverter (i.e., diffusion at node n3 and n4) are the sensitive nodes/junctions.

Different design styles of the master and slave latches can result in different FF SERs. How the master and slave latches are connected within a FF can also affect the SER. For example, the master and slave latches can be connected with either a transmission gate or a tri-state inverter. When a transmission gate is used, the two latches actually store complementary logic values. Meanwhile, using a tri-state inverter will make the two latches store the same logic value.

In this work, we will be leveraging both layout and circuit dependence of the latch and FF designs to introduce SER asymmetry. The detailed design changes and results will be described in Section IV.



(a) Schematic



(b) Layout Stick Diagram

Fig. 1. The schematic and stick diagram example of a D latch [5].

III. Methodology Overview

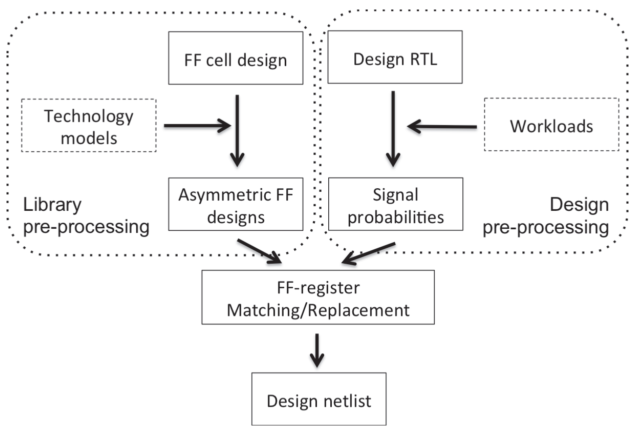
As described earlier, this work is motivated by two key observations about the logic values stored in FF cells. The first motivation is that FFs can be asymmetric, i.e., have different SER, when storing different logic values. This asymmetric SER has been reported [7], [6] at different technology nodes and FF cells. The second motivation is that registers in a design can have different signal probabilities of holding different logic values. In this work, we define signal probability (SP) as the probability of a register holding a value 1 during the program execution. Since registers in a design RTL are usually implemented using FF cells, we propose to implement registers of different SP with FF cells of different SER asymmetry.

The methodology flow chart is illustrated in Fig. 2. The flow involves some pre-processing steps for both the design and cell library. For the library pre-processing, the FF from the standard cell library is used to design different versions of the FF cells. Technology models can be used to characterize the SER of these FF cells. If technology models, especially soft error models are not available, Q_{crit}-based analysis [3] together with soft error models [14] can also be used to obtain the relative SER ratio.

For the design pre-processing, the SP of each register in the design can be obtained by running RTL simulation with several representative workloads. In absence of such representative workloads, logic-based analysis can also be applied by assuming certain input signal probabilities and analyzing the SP of the register inputs.

After the pre-processing, the registers can be matched with the specific FF design that will minimize the SER. For example, if a register has very high SP, i.e., more likely to store a logic value 1, the register should be implemented with the FF that is more robust holding logic 1, i.e., has smaller SER when Q=1.

This process can be done at different design stages. For example, it can be done during the logic synthesis so that it generates the design netlist with all FF cell types specified. The process can also be applied before circuit implementation. In this case, the registers will be annotated with the preferred FF cell asymmetry. The annotation can be implemented using methods similar to pragmas. If the design has been placed and routed, the process can also be applied as part of the Engineering Change Order (ECO) process. As we will show in later sections, the ECO process can be quite straightforward without significant increase in the design complexity.



(a) Schematic

Fig. 2. Overview of the proposed soft error mitigation methodology.

IV. Designing Asymmetric FFs

One of the key steps in the proposed methodology is designing FFs with asymmetric SER. FF SER can be affected by different factors. The underlying process technology, e.g., physical characteristic of the P and N diffusions, can result in varied amount of collected charge and thus different SER. However, the process technology is usually not in the control of chip designers.

Some cell-level optimization techniques can also greatly affect SER. For example, common hardening techniques can greatly reduce the overall SER. But these techniques usually rely on either adding storage node redundancy or increasing diffusion spacing, which incur significant area overhead.

In this work, we propose to exploit the FF SER asymmetry instead of designing FF cells with smaller SER. This allows us to modify the FF design with either zero or very small overhead. We will demonstrate how FF SER asymmetry can be introduced with layout changes in Section IV-A and schematic changes in Section IV-B. Some asymmetric FF results are presented in Section IV-C.

A. Asymmetry through Layout Changes

As discussed earlier in Section II-B, both diffusion area and sensitivity can affect the SER. Since the basic storage element in a FF is a latch, in this section, we will focus on latch design. We will demonstrate how to introduce SER asymmetry with zero overhead layout changes.

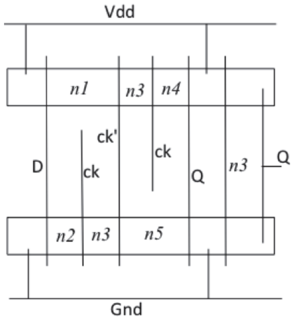
There are different ways to implement the D latch in Fig. 1(a). An example of the stick diagram is shown in Fig. 1(b). This layout is symmetric in terms of P and N diffusion ordering and area. However, the "X" shape crossing of the gate may violate the more strict design rules in advanced technology nodes. The splitting of the transistors is unavoidable. Two possible implementation options are shown in Fig. 3(a) and Fig. 3(b). The two stick diagrams are implementing the same exact schematic in Fig. 1(a) but with different transistor ordering and diffusion sizes. The design Latch_L has larger diffusion area for $n5$ while Latch_H has larger diffusion area for $n4$.

As discussed earlier in Section II-B, when the latch is holding a value 0, i.e., Q=0, $n4$ is the critical junction while $n5$ is not critical. Therefore, Latch_L will have smaller SER than Latch_H, i.e., better for holding value 0. When the latch is holding a value 1, i.e., Q=1, Latch_H will have smaller SER because of smaller critical area for $n5$.

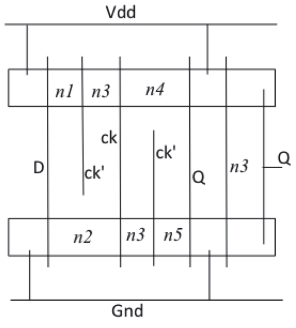
These two latch designs, Latch_H and Latch_L, have the same area. They also have very similar delay and power, as the capacitance loading for clock and input pins are similar. So without changing the schematics, each of the master and slave latches can be altered to have SER asymmetry. Since these layout changes do not affect the area, the FF cells can have the same footprint and I/O pin locations. This makes them easier for replacement during ECO process.

B. Asymmetry through Schematic Changes

Other than the layout styles of the latch, the schematics of the FF, e.g., how latches are connected, can also introduce SER asymmetry. Two examples of designing master-slave FF are illustrated in Fig. 4.

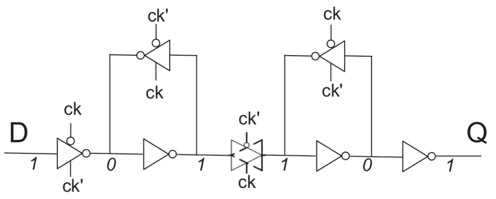


(a) Latch_L Stick Diagram

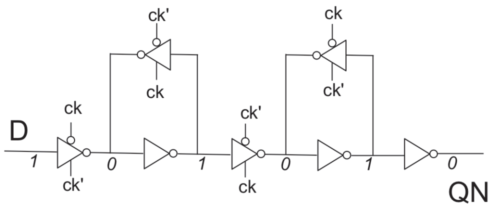


(b) Latch_H Stick Diagram

Fig. 3. Two different layout styles of implementing the D latch in Fig. 1(a).



(a) DFFQ with latches connected through transmission gate



(b) DFFQN with latches connected through tri-state inverter

Fig. 4. FF designs with different gates connecting between the master and slave latches.

If the two latches are connected through a transmission gate as shown in Fig. 4(a), the master latch and the slave latch will be storing complementary logic values. In the example shown in Fig. 4(a), if D=1, the master latch, i.e., the left latch, will store a logic value 1. However, the slave latch will be storing a logic value 0.

The two latches can also be connected through a tri-state inverter, as shown in Fig. 4(b). In this case, the two latches will be storing the same logic value. In the example shown in Fig. 4(b), if D=1, both the master and slave latches will be storing logic value 1. Due to the additional invertion of the tri-state inverter, the cell's output is QN instead of Q. Additional inverter can be used to construct normal DFFQ function with this cell. The invertion can also be inserted during logic synthesis to avoid the overhead.

The options between transmission gate and tri-state inverter offer another way to design FF with different asymmetry. The tri-state inverter has two more transistors and no longer zero overhead. But this is particularly important when the physical characteristics of P and N diffusion is very different. For example, if the P-type diffusion is more sensitive to soft error than N-type diffusion, the layout changes may not give enough asymmetry if the latches store complementary logic values.

C. *FF Asymmetry Results*

We have demonstrated different layout and schematic techniques to introduce SER asymmetry in FF. We will verify these techniques in this section by designing different styles of FFs using a sub-32nm commercial process technology. The process models, including device models and soft error models are used with commercial tools to generate the delay and SER at sea level.

The FFs design information is summarized in Table I. In the experiments, we design four different FFs. The L and H in the FF names are use to describe the master and slave latches. L and H means the latch is designed for Q=0 or Q=1, respectively. The comparison with DFFQ_LH, DFFQ_LL, and DFFQ_HH is used to evaluate the effectiveness of layout changes described in Section IV-A. Due to the complementary effects of master and slave latches when connected using transmission gate, DFFQ_LH contains Latch_L for both master and slave latches. The comparison between DFFQ_LH and DFFQ_I is used to evaluate the schematic changes described in Section IV-B.

The experimental results are also summarized in Table I. All SER results are normalized with respect to the SER of DFFQ_LH when Q=1. All delay and area results are also normalized with respect to DFFQ_LH. The SER ratio, i.e., SER(Q=1) over SER(Q=0), of DFFQ_LH is 0.52, which is inline with the ratio reported in [7], [6]. The other two FFs with layout changes have SER ratio of 0.39 and 0.70. As mentioned earlier, the layout changes are limited by the complementary property of the transmission-gate-based FF. The FF with tri-state inverter, i.e., DFFQN_I can further improve the asymmetry and have SER ratio of 5.11.

V. FF Matching Methodology

In this section, we describe proposed methodology to match different FF cells with corresponding registers in the design. We will first perform the analysis of register SP in Section V-A. We present the FF-register matching methodology in Section V-B.

A. *Register SP Analysis*

Registers in a design RTL are usually implemented as FFs. Depending on the signal stored, different registers can have different probabilities of storing different logic values (i.e., different SP). The SP values can be obtained from running representative workloads. Since the register-FF matching is done at design time, it is a static method that cannot be adaptive based on the application running. So we should obtain the SP considering different application types.

The key question is whether the SP is stable under different types of applications. To validate this, we perform RTL simulation on a commercial processor. We pick four benchmarks, dhrystone, dhrystone64, max_power64 and saxpy64 that covers different and even extreme use cases. Dhrystone is a benchmark designed specifically for representing system software. The difference between dhrystone and dhrystone64 are the use of different compiler and instruction bit-width. Max_power64 is a benchmark specific written to stimulate the processor as much as possible. It is written in assembly with extensive use of loops and branch instructions. Saxpy is a benchmark for floating-point based computation. These four benchmarks are used to evaluate different application characteristics, e.g., 64-bit vs. 32-bit, integer vs. floating-point, computation vs. branches.

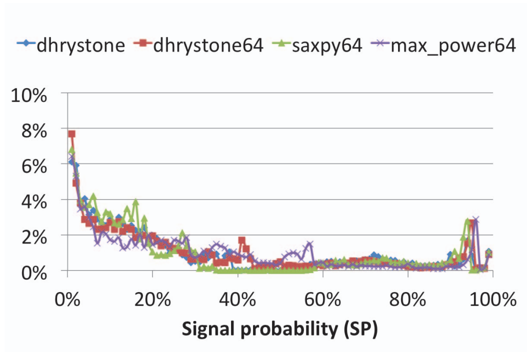


Fig. 5. Register signal probability histogram of different benchmarks. For example, 5% at 10% SP means there are 5% of registers in the design with SP=10%.

During the simulation, we record the total number of cycles that each register stores the logic value 1. So SP equals the total number of cycles that a particular register stores the logic value 1 divided by the total number of simulation cycles. The histogram of SP is plotted in Fig. 5. The SP histogram is very consistent across different benchmarks. Overall, the SP for max_power64 is slightly higher than other benchmarks, which is expected as it is used to stimulate as many nodes as possible.

For each of the register, we also record the maximum and minimum SP among these four benchmarks. The delta between maximum and minimum represents the span of the SP among different types of benchmarks. The results of the span of SP are shown in Fig. 6. Most of registers have very stable SP with span less than 20%. There are two groups of registers have span around 50% and 90%.

In the matching process, the average SP among different representative workloads should be used. As discussed earlier, if representative workloads are not available, logic analysis can also be applied. The other alternative is that the RTL designer can annotate the expected SP for some key registers, especially for control registers whose errors are more likely to affect the program execution.

B. *FF Matching Methodology*

With the two pre-processing steps to identify the FF SER asymmetry and register SP, we can match them so that the overall SER for each register is minimized. The average SER of a register can be calculated with Equation (1).

$$SER_{avg} = SP * SER_{Q=1} + (1 - SP) * SER_{Q=0} \quad (1)$$

TABLE I
THE FF DESIGN INFORMATION AND EXPERIMENTAL RESULTS. RESULTS ARE NORMALIZED WITH THAT OF DFFQ_LH.

	Master latch	Slave latch	Connecting gate	SER(Q=1)	SER(Q=0)	SER ratio(Q=1/Q=0)	Setup time	Clk-to-Q delay	Area
DFFQ_LH	Latch_L	Latch_L	transmission gate	1	1.92	0.52	1	1	1
DFFQ_LL	Latch_L	Latch_H	transmission gate	0.80	2.03	0.39	0.97	0.99	1
DFFQ_HH	Latch_H	Latch_L	transmission gate	1.09	1.56	0.70	1.07	0.99	1
DFFQN_I	Latch_L	Latch_L	tri-state inverter	2.30	0.45	5.11	0.93	1.08	1.14

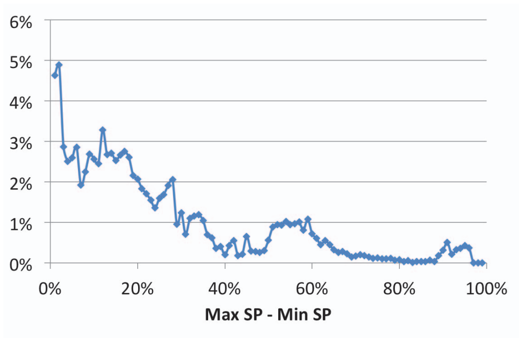


Fig. 6. The histogram of register signal probability span across different benchmarks. For example, 2% at 20% point means that there are 2% of register in the design have SP span of 20%.

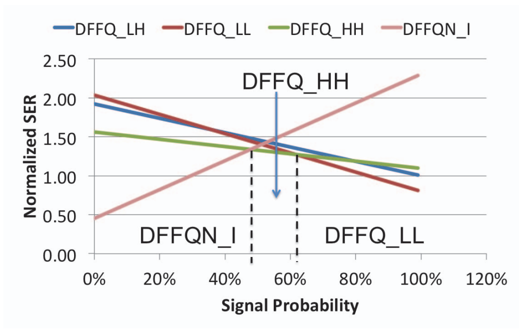


Fig. 7. A graph to illustrate the register-FF matching method.

Since the SER is linear with respect to SP, there can be a simple method to identify the matching criteria. For example, with the four FFs with known SER shown in Table I, we can plot the relationship between FF SER and SP, as shown in Fig. 7. For registers at each SP, we want to pick the FF that minimize the SER. Similar to linear programming, based on the crossing points of lines representing different FFs, we can divide the space into different regions. For the example shown in Fig. 7, there are two crossing points which divide the SP space into three regions. The registers within each region will be matched to the type of FF that represents the line on top of the region.

Since all FFs in this work have same or similar specifications, the matching objective is to optimize for the SER only. In some scenarios when the SER requirement for the design is higher, the design may need to incorporate hardened FFs as well. In that case, the hardened FF replacement can be applied after this register-FF matching. Alternatively, the flip-flop selection and matching problem can be formulated as a linear programming problem.

VI. EXPERIMENTAL RESULTS AND DISCUSSIONS

The FF asymmetry results are already described in Section IV-C, and design SP results are already described in Section V-A. In this section, we will present the experimental results of the register-FF matching method. The experimental setup is described in Section VI-A. The system SER results are presented in Section VI-B. Some discussions about the methodology and results are included in Section VI-C.

A. Experimental Setup

The FFs used in the experiments are the ones described in Table I. As described in Section IV-C, these cells are implemented based on the sub-32nm commercial process technology. All cells are designed with corresponding transistor netlist and layout GDS. A liberty (.lib) file is generated using commercial library characterization tool to obtain the delay values, including setup time and Clock-to-Q delay. The SER values are evaluated using the soft error model calibrated with the process.

The design benchmark is a commercial processor. As described in Section V-A, four benchmarks are used in the simulation. The average SP among these four benchmarks is used for the matching method described in Section V-B.

B. System SER Saving

We apply the matching method described in V-B. For comparison purpose, we assume that the baseline design uses DFFQ_LH for all of the registers. We apply the matching method with two sets of FFs. The zero-overhead one uses only FFs with layout changes, i.e., DFFQ_LH, DFFQ_LL, and DFFQ_HH. The low-overhead one uses all four FFs from Table I.

The percentage of SER improvement for register of different SP is plotted in Fig. 8. By using the FFs with layout changes only, the per FF SER improvement varies between 10% to 20%. With the additional FF with schematic changes, the per FF SER improvement can be further improved to about 75% for low SP registers.

The improvement of system SER for different register SP is plotted in Fig. 9. The values show which types of registers contribute more to the overall system SER reduction. Most system SER reduction emanates from registers with smaller SP. There are two reasons for this. The first reason is that

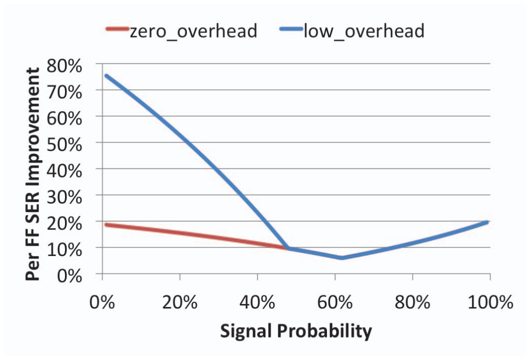


Fig. 8. The per FF SER improvement for registers with different SP.

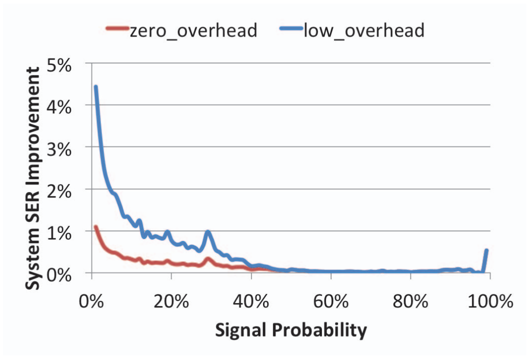


Fig. 9. The improvement on the system SER for registers with different SP.

there are more registers in the design with smaller SP, as shown in Fig. 5. The other reason is that FFs with smaller SP have much better per FF SER reduction, as shown in Fig. 8.

Overall, the improvement in system SER is 16% using the FFs with zero-overhead. By allowing schematic changes, the system SER reduction can be improved to 48%. The schematic changes add 6.7% overhead to the FF area.

C. Discussion

FF SER asymmetry has been reported with experiments [7], [6]. This work exploits this circuit-level property with the system-level information about the design and data-dependence. We have demonstrated that the high-level usage information can be very helpful in making lower-level design decisions and optimizations.

In this work, we obtain the usage information (i.e., SP) through simulation with representative benchmarks. In some scenarios, this may not be possible or can take a lot of computation time. There may be some pathological workloads that can give different or even biased results. Design annotation can be an alternative to the simulation-based methods.

For the implementation process, the proposed register-FF matching can be applied at different design stages. Since we are only modifying the FFs, it can also be applied as part of the ECO process, especially using the FFs with different layout modifications. These FFs have same footprint and pin locations, making it compatible without additional placement/routing changes.

VII. Conclusion

In this paper, we proposed a methodology to mitigate system soft error through exploiting register data-dependence and FF SER asymmetry. We proposed a flow to design FF cells with different SER asymmetry and match them with registers of different SP. We demonstrate on a commercial processor that this methodology can be applied with layout changes only and achieve 16% system SER reduction with zero overhead. With 6.7% increase in FF area, the system SER reduction can be improved to 48%. Future work will seek to combine this method with conventional FF hardening methods to meet varied SER requirements.

References

[1] D. Alexandrescu. A comprehensive soft error analysis methodology for socs/asics memory instances. In *On-Line Testing Symposium (IOLTS), 2011 IEEE 17th International*, pages 175–176. IEEE, 2011.

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

[3] V. Chandra and R. Aitken. Impact of technology and voltage scaling on the soft error susceptibility in nanoscale cmos. In *Defect and Fault Tolerance of VLSI Systems, 2008. DFTVS'08. IEEE International Symposium on*, pages 114–122. IEEE, 2008.

[4] M. Ebrahimi et al. Comprehensive analysis of alpha and neutron particle-induced soft errors in an embedded processor at nanoscales. In *Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014*, pages 1–6. IEEE, 2014.

[5] K. Eshraghian and N. Weste. Principles of cmos vlsi design. *Addiso-Wesley Pub. Company*, 1993.

[6] J. Furuta et al. A 65nm flip-flop array to measure soft error resiliency against high-energy neutron and alpha particles. In *16th Asia and South Pacific Design Automation Conference (ASP-DAC 2011)*, pages 83–84. IEEE, 2011.

[7] T. Heijmen et al. A comprehensive study on the soft-error rate of flip-flops from 90-nm production libraries. *IEEE Transactions on Device and Materials Reliability*, 7(1):84–96, 2007.

[8] L. H.-H. Kelin, L. Klas, B. Mounaim, R. Prasanthi, I. R. Linscott, U. S. Inan, and M. Subhasish. Leap: Layout design through error-aware transistor positioning for soft-error resilient sequential cell design. In *Reliability Physics Symposium (IRPS), 2010 IEEE International*, pages 203–212. IEEE, 2010.

[9] L. Lai et al. Evaluating and exploiting impacts of dynamic power management schemes on system reliability. In *International Conference on Compilers, Architecture and Synthesis for Embedded Systems*, 2015.

[10] K. Lilja et al. Single-event performance and layout optimization of flip-flops in a 28-nm bulk technology. *IEEE Trans Nucl Sci*, 60(4):2782–2788, 2013.

[11] S. Mitra et al. Built-in soft error resilience for robust system design. In *Integrated Circuit Design and Technology, 2007. ICICDT '07. IEEE International Conference on*, pages 1–6, May 2007.

[12] N. Seifert et al. Soft error susceptibilities of 22 nm tri-gate devices. *IEEE Transactions on Nuclear Science*, 59(6):2666–2673, 2012.

[13] N. Seifert et al. Soft error rate improvements in 14-nm technology featuring second-generation 3d tri-gate transistors. *IEEE Transactions on Nuclear Science*, 62(6):2570–2577, 2015.

[14] P. Shivakumar et al. Modeling the effect of technology trends on the soft error rate of combinational logic. In *Dependable Systems and Networks, 2002. DSN 2002. Proceedings. International Conference on*, pages 389–398. IEEE, 2002.

[15] D. Zhu et al. Reliability-aware energy management for periodic real-time tasks. *Computers, IEEE Transactions on*, 58(10):1382–1397, 2009.