SEU Fault Evaluation and Characteristics for SRAM-Based FPGA Architectures and Synthesis Algorithms

NAIFENG JING, Shanghai Jiao Tong University
JU-YUEH LEE and ZHE FENG, University of California, Los Angeles
WEIFENG HE and ZHIGANG MAO, Shanghai Jiao Tong University
LEI HE, University of California, Los Angeles

Reliability has become an increasingly important concern for SRAM-based field programmable gate arrays (FPGAs). Targeting SEU (single event upset) in SRAM-based FPGAs, this article first develops an SEU evaluation framework that can quantify the failure sensitivity for each configuration bit during design time. This framework considers detailed fault behavior and logic masking on a post-layout FPGA application and performs logic simulation on various circuit elements for fault evaluation. Applying this framework on MCNC benchmark circuits, we first characterize SEUs with respect to different FPGA circuits and architectures, for example, bidirectional routing and unidirectional routing. We show that in both routing architectures, interconnects not only contribute to the lion's share of the SEU-induced functional failures, but also present higher failure rates per configuration bits than LUTs. Particularly, local interconnect multiplexers in logic blocks have the highest failure rate per configuration bit. Then, we evaluate three recently proposed SEU mitigation algorithms, IPD, IPF, and IPV, which are all logic resynthesis-based with little or no overhead on placement and routing. Different fault mitigating capabilities at the chip level are revealed, and it demonstrates that algorithms with explicit consideration for interconnect significantly mitigate the SEU at the chip level, for example, IPV achieves 61% failure rate reduction on average against IPF with about 15%. In addition, the combination of the three algorithms delivers over 70% failure rate reduction on average at the chip level. The experiments also reveal that in order to improve fault tolerance at the chip level, it is necessary for future fault mitigation algorithms to concern not only LUT or interconnect faults, but also their interactions. We envision that our framework can be used to cast more useful insights for more robust FPGA circuits, architectures, and better synthesis algorithms.

Categories and Subject Descriptors: B.8.1 [**Performance and Reliability**]: Reliability, Testing, and Fault-Tolerance

General Terms: Reliability, Design, Measurement

Additional Key Words and Phrases: FPGA, soft error, SER, interconnect, routing, architecture, synthesis

ACM Reference Format:
Jing, N., Lee, J.-Y., Feng, Z., He, W., Mao, Z., and He, L. 2012. SEU fault evaluation and characteristics for SRAM-based FPGA architectures and synthesis algorithms. ACM Trans. Des. Autom. Electron. Syst. 18, 1, Article 13 (December 2012), 18 pages.
DOI = 10.1145/2390191.2390204 http://doi.acm.org/10.1145/2390191.2390204

This work is supported by the National Hi-Tech Research and Development Program (863) of China under Grant 2009AA01170.

Authors' addresses: N. Jing (corresponding author), W. He, and Z. Mao, School of Microelectronics, Shanghai Jiao Tong University; email: jingnaifeng@ic.sjtu.edu.cn; J.-Y. Lee, Z. Feng, and L. He, Department of Electrical Engineering, University of California, Los Angeles.

DOI 10.1145/2390191.2390204 http://doi.acm.org/10.1145/2390191.2390204

1. INTRODUCTION

Reliability has become an increasingly important design consideration for nanoscale SRAM-based field programmable gate arrays (FPGAs) in the past decade. Although modern FPGAs are capable of more powerful designs benefiting from the continuous technology scaling, they also become more sensitive to soft errors due to a large amount of SRAM cells used for their configurability. Suffering from the soft errors caused by cosmetic radiation or circuit internal noise [Sterpone et al. 2006], the fault sensitivity of these SRAM cells increases due to technology scaling by smaller feature size, higher logic density and lower operating voltage [Golshan et al. 2007]. The effect causes so called SEUs (single event upset) that change the logic state of a SRAM cell bit and may lead to functional failures of the implemented applications on FPGAs.

In general, SEUs in SRAM-based FPGAs may undermine either user memory cells or configuration memory cells. The former faults are usually acknowledged as transient because they can be overwritten during execution. In contrast, although faults in configuration memory cells can be corrected by reprogramming, they may still cause functional failures until the next reprogramming, and thus reduce the MTTF (mean time to failure), a measurement of system reliability. In addition, typically 98% of the FPGA SRAM cells are configuration memory cells [Asadi et al. 2005], which can be up to 160 million in a modern FPGA device [Xilinx Inc. 2010]. Termed as CRAM bits in this article, the configuration SRAM cells mostly reside in programmable logic blocks for functionality and routing elements for interconnect, controlling the behavior of the FPGA. Unfortunately, these CRAM bits are all subject to SEUs, which may change the circuit functionality or even circuit structure, resulting in system failures. As a result, SEUs in SRAM-based FPGAs have made reliability a major concern for FGPA users.

There have been extensive studies to mitigate the impact of SEU on CRAM bits. TMR (triple modular redundancy) is a classic technique using redundancy to reduce the fault-induced failures, but is known to have high overhead in area, power and performance. Recently, several logic resynthesis-based techniques have been proposed (e.g., [Hu et al. 2008; Feng et al. 2009, 2011; Lee et al. 2010; Jose et al. 2010; Jing et al. 2011a]). They apply different logic masking strategies to reduce failures either in LUT or interconnect and involve minimal overhead in area, power, and performance.

In order to design robust FPGA circuits, architectures, or synthesis algorithms with respect to SEU, ideally we should identify the most SEU-sensitive circuit element in FPGAs. Previous work estimating the fault sensitivity, or failure rate, is mainly hardware emulation-based, radiation-based, or a combination of both [Rebaudengo et al., 2002; Johnson et al. 2003; Graham et al. 2003; Bellato et al. 2004; Heron et al. 2005; Asadi et al. 2007]. However, it is hard for those methods to predict the failure on a specific CRAM bit during design time. Software-based simulation and analytical approaches are also proposed [Krishnaswamy et al. 2007; Asadi et al. 2005], but without explicit consideration for interconnect. In addition, these existing estimating approaches assume a bidirectional routing architecture. However, modern FPGA routing architecture has shifted from conventional bidirectional routing towards unidirectional routing, where fault sensitivity has not yet been evaluated. We will discuss the methods in detail in Section 2.1.

Targeting generic FPGA architectures and applying logic simulation on a post-layout FPGA application, this article first develops a comprehensive SEU fault evaluation framework for SRAM-based FPGAs. It can quantify the failure rate, that is, the probability of an SEU that causes a system failure on a basis of each CRAM bit. Then, the framework is applied to characterize the fault on different applications under several typical FPGA architectures. It demonstrates that different circuit elements present significantly different failure sensitivities and that the faults on interconnects are dominant. At the same time, local multiplexers (MUXes) in logic blocks present the highest averaged failure rate per bit. In addition, concerning the interconnect fault, several logic resyntheses-based fault mitigation algorithms are evaluated to see their improvements at the chip level. It shows that the mitigation algorithm with explicit consideration for interconnect leads to significantly better results. At the same time, interactions between these resynthesis algorithms are also investigated by various combinations of them. It reveals that future fault mitigation algorithms should be developed not only concerning the faults on LUT or interconnect, but also their interactions, in order to improve the fault tolerance to the greatest extent. We envision that our fault evaluation framework will be used to cast more useful insights for more robust FPGA circuits, architectures, and better synthesis algorithms.

The article is organized as follows. Section 2 briefly introduces the previous works on failure rate estimation and resynthesis-based fault mitigation algorithms. In Section 3, we introduce the preliminaries on FPGA architecture and the SEU behavior in various circuit elements. Then, our proposed fault evaluation framework is presented in Section 4. In Sections 5 and 6, our framework is applied to evaluate FPGA circuits, architectures, and synthesis algorithms. Section 7 concludes this article.

2. PREVIOUS WORKS

2.1. Failure Rate Estimation

Previous work estimating SEU-induced failure rate in SRAM-based FPGAs is mainly hardware emulation-based, radiation-based, or a combination of both [Rebaudengo et al. 2002; Johnson et al. 2003; Graham et al. 2003; Bellato et al. 2004; Heron et al. 2005; Asadi et al. 2007]. They use the fault injection strategy and perform hardware emulation on fabricated devices to collect probabilities that the faults can be sensitized by primary inputs and propagated to primary outputs. However, as these approaches usually target specific devices, like the Xilinx Virtex family, it is hard for these methods to predict the failure sensitivity in the perspective of FPGA structures and elements, for example, an exact CRAM bit. In addition, they cannot predict the failure sensitivity during design time. Therefore, the information collected is too late to change FPGA circuits and architectures for reliability.

A software-based simulation estimation approach is proposed in Krishnaswamy et al. [2007]. It applies bit-level logic simulation to predict the sensitivity on logic gate nodes but does not consider the faults on interconnects. Another software-based analytical approach is proposed in Asadi et al. [2005], which estimates the sensitivity of an FPGA circuit node affected by an SEU. However, their method cannot be practically used for larger circuits due to the reconvergent path problem. It is impossible to consider all signal reconvergence on both faulty and non-faulty paths according to their method. Although they consider the faults on interconnect, they treat multiple configuration bits on each net as a single node whose sensitivity is then simply estimated by the number of configuration bits on that net. This simplification neglects the variation of the failure sensitivities for different configuration bits on interconnects, and thus is improper for interconnect failure sensitivity estimation.

In addition, these existing fault estimation approaches and fault mitigation approaches (e.g., [Sterpone et al. 2006; Golshan et al. 2007; Asadi et al. 2005; Krishnaswamy et al. 2007]) all assume a bidirectional routing architecture. However, modern FPGA routing has shifted from the conventional bidirectional routing towards unidirectional routing architecture [Lemieux et al. 2004; Luu et al. 2009; Smith et al. 2009]. In our previous work [Jing et al. 2011a], we talked about the SEU evaluation in this unidirectional routing architecture. In this article, we extend that work by considering both the routing architectures to characterize SEU fault. We evaluate SEU faults on a post-layout circuit, which means we can analyze the fault sensitivity on each CRAM bit in a placed and routed FPGA. Our approach can be performed during design time such that the most failure-sensitive element can be identified as early as possible for more reliable FPGA circuits, architectures, and synthesis algorithm design.

2.2. Resynthesis-Based Fault Mitigation Algorithms

Recently, several logic resynthesis-based SEU mitigation techniques have been proposed, such as ROSE [Hu et al. 2008], IPR [Feng et al. 2009], IPD [Lee et at. 2010], R2 [Jose et al. 2010], IPF [Feng et al. 2011], and IPV [Jing et al. 2011b], which apply different logic masking strategies to mitigate the fault impact with minimum overhead in area, power, and performance. Applied on non-mission-critical FPGA applications, such as networking and communication, these algorithms can reduce SEU-induced failure rates on an LUT or interconnect significantly.

A common feature of the IPD, IPF, and IPV algorithms is that they can be in-place performed within LUTs without changing the placement and routing (IPD may slightly alter some local routing), and thus do not affect design closure. The IPD (In-Place Decomposition) algorithm decomposes a function in an LUT into two sub-functions and combines them by a converging logic that provides logic masking capability. The subfunctions can be in-place implemented by decomposable under-utilized LUTs, and the converging logic can be implemented by built-in carry chains. By leveraging the architectural redundancy in each logic block, IPD can improve reliability without changing global routing. The IPF (In-Place X-Filling) algorithm leverages logic redundancy, that is, don't care bits in LUTs, and proposes several heuristics to refill the logic values of these bits based on bit-level failure sensitivity. During the refilling, the logic functionality can be kept equivalent and the logic masking capability in the sink LUT can be enhanced to mitigate the faults in the fan-in LUTs. The IPV (In-Place inVersion) algorithm is based on the fact that routing CRAM bits contributes to the majority of the CRAM bits in FPGA and are more sensitive to functional failures. It explicitly considers the detailed fault behavior on routing elements in a post-layout FPGA application. For example, for MUX-based unidirectional routing, when an SEU occurs on a routing CRAM bit, the manifestation of the fault depends on the signal discrepancy at the faulty MUX and the propagation observability [Krishnaswamy et al. 2007; Lee et al. 2010] from the faulty MUX to primary outputs. By selectively inverting the logic polarity of the driving LUTs that has higher fault propagation observability, the failure rate from interconnect can be reduced.

The IPD, IPF, and IPV algorithms can be combined to deliver more fault mitigation capability, as later demonstrated in this article. In addition, as in-place resynthesis-based techniques, they are also orthogonal to existing fault mitigation techniques, which means they can be jointly applied with other fault mitigating methods, such as TMR. In this article, we will focus on the three algorithms and their combinations to evaluate their improvements on FPGA reliability with respect to SEU.

3. ANALYSIS OF FAULT BEHAVIOR IN FPGA ARCHITECTURE

In this section, we first introduce the generic architecture used in our framework for SEU analysis in FPGA. Three types of FPGA circuit elements are focused in our framework. Then, we analyze SEU behavior on CRAM-bit basis in the three different elements. These bits concerned in this article contribute to the majority of CRAM bits in FPGA. Although there are other CRAM bits, their number is relatively tiny and can be neglected [Graham et al. 2004].

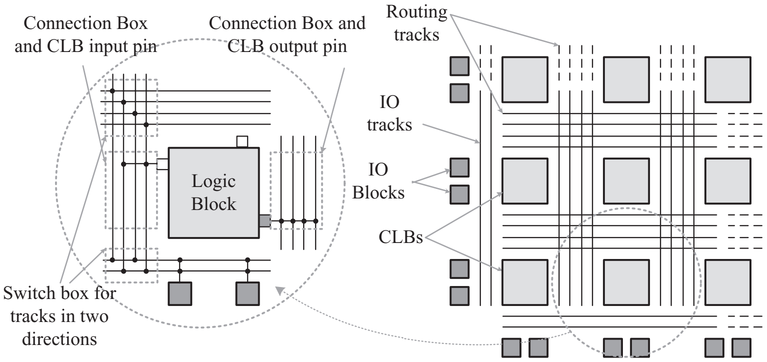

Connection Box and CLB input pin
Connection Box and CLB output pin
Routing tracks
IO tracks
IO Blocks
CLBs
Logic Block
Switch box for tracks in two directions

Fig. 1. Generic FPGA architectural model.

3.1. FPGA Architecture Overview

An FPGA architecture is mainly defined by CLBs (configurable logic blocks) and routing architectures. In this article, we take the cluster-based logic blocks like VPR [Betz and Rose 1997; Luu et al. 2009] as the architectural description to characterize the SEU fault in SRAM-based FPGAs. Figure 1 illustrates an FPGA consisting of a 2D array of the CLBs that are selectively connected by the inter-CLB routing architecture (exchangeably called global routing in this article). Each CLB can be parameterized by (k, N), that is, it consists of N LUTs, and each LUT has k inputs. The LUT inputs and outputs are fully connected by intra-CLB routing (also called local routing) architecture, which is mainly implemented by MUXes allowing signals to be routed respectively between CLB inputs (or outputs) and LUT inputs (or outputs). The CRAM bits stored in each LUT implement the desired functionality.

Interconnects are critical to FPGAs, since the routing structure contributes a large portion of the total FPGA area and CRAM bits. This article considers the island style routing that is wildly used in commercial FPGAs. The CLBs are connected via inter-CLB routing elements, that is, switch boxes and connection boxes connecting wires deployed in routing channels via the bidirectional or unidirectional PIPs (programmable interconnecting points) within switch boxes and connections boxes. Typically, bidirectional PIPs are implemented by pass transistors, while the unidirectional PIPs are the selection bits in MUXes. These CRAM bits that configure the PIPs contribute to most of the CRAM bits in FPGAs.

3.2. SEU Fault Overview in FPGA

Different circuit elements in an FPGA behave differently when affected by an SEU. In this section, we are going to study the fault behavior of the CRAM bits in LUTs, local routing MUXes, and global routing PIPs according to the generic architecture description preceding and their micro-architecture typically used in the FPGA circuit. Both the bidirectional and unidirectional routing architectures will be considered. Note that previous work treats multiple CRAM bits on one net just as a single node [Asadi et al. 2005]. In our framework, each bit is evaluated based on the detailed post-layout circuit for more realistic predication of its failure sensitivity.

3.2.1. SEU on LUTs. A typical implementation of an LUT can be illustrated in Figure 2. For a k input LUT, there are 2^k CRAM bits for the desired logic function. The k inputs make a cascading selection on these bits and provide one bit as the final LUT output.

The behavior of an SEU on an LUT CRAM bit is straightforward. An SEU on any CRAM bit of the 2^k bits may flip the LUT output when the affected bit happens to

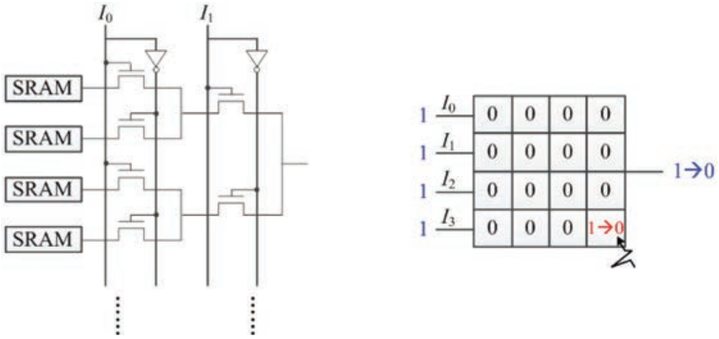


Fig. 2. SEU on an LUT.

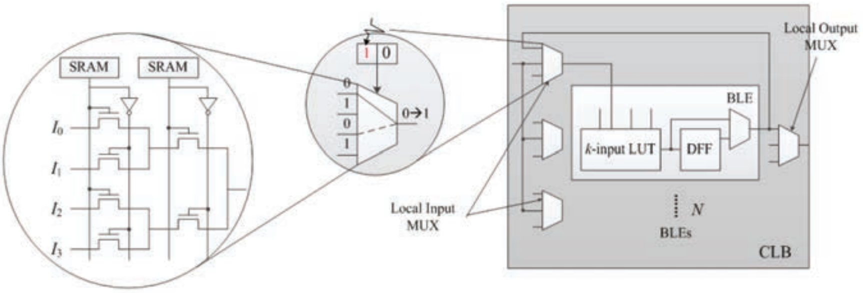


Fig. 3. SEU on an input MUX inside a logic block.

be accessed under certain input patterns. The affected value may further propagate throughout the logic network and finally result in a functional failure when it reaches the primary outputs of the circuit.

3.2.2. SEU on Intra-CLB Routing. The intra-CLB routing connects the CLB inputs (and outputs) to LUT inputs (and outputs) within each CLB. In the generic architecture to be evaluated in this article, we allow for a full connection for all the inputs and outputs. Local routing primarily uses MUXes for signal selection. Therefore, for the typical implementation of a local routing network, as illustrated in Figure 3, each k input pin of the N LUTs has its own input MUX with several CRAM bits, which are programmed to select any of the CLB inputs, as well as the N outputs of the LUTs within the same CLB. At the same time, the N LUT outputs can also be connected to any of the CLB outputs via the local routing MUXes. The MUXes enable the arbitrary interconnecting capability within each CLB.

Figure 3 further shows a typical structure for an encoded MUX under an SEU. The CRAM bits controlling this MUX cascadingly select one signal to drive the MUX output. Once affected by the SEU, one of the encoded CRAM bits will flip its state and thus select an erroneous input signal onto the output. Different from the SEU on an LUT, the affected configuration bit will always mistakenly select an irrelevant signal until the next reprogramming. The irrelevant signal has the chance of being propagated to primary outputs and raising an unwanted functional failure.

3.2.3. SEU on Bidirectional Routing. Conventionally, inter-CLB routing is typically interconnected via bidirectional pass transistors [Sterpone et al. 2006; Golshan et al. 2007],

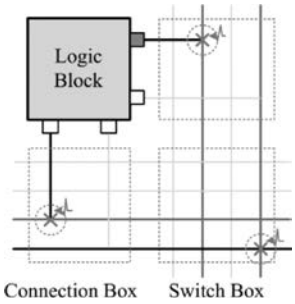


Fig. 4. Open fault in connection box or switch box in a schematic view.

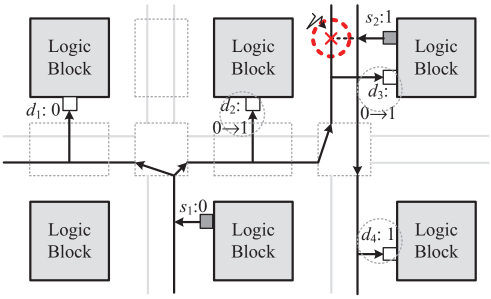


Fig. 5. Short fault in a schematic view (bridging at the crossing point).

and its connectivity within a connection box or a switch box is configured by CRAM bits. Once affected by an SEU, these bits either Temporarily Stuct-At-0 (TSA0) or Stuct-At-1 (TSA1). Then the signal they carry are undermined, and faults are injected in the circuit until it can be corrected by reprogramming.

Figure 4 illustrates SEU open fault, that is, TSA0, which breaks the originally connected wires at the faulty point in the connection box or switch box. The outgoing wire from the open point carries an unknown signal whose value depends on the FPGA circuit. To be practical, we assume that the broken net will be tied off to either *Vdd* or *Gnd* [Reddy et al. 2005] such that the following transistors can be prevented from being conducted as short circuits, which should be avoided in CMOS designs. However, the tied-off value may bring an input fault to the immediate fan-outs of the faulty point if it is different from the desired value. The fault may propagate through the fan-out network and finally be observed at primary outputs as a circuit failure.

SEU short fault, that is, TSA1, bridges two adjacent wires when they both pass through the same connection box or switch box. Figure 5 illustrates an example, where two nets are bridged due to an SEU in the top-right connection box. In fact, bridging may not always inject faults into the circuit. It depends on the driving logic and strengths along the two nets. If both the driving signals are the same, the signal at the faulty point is forwarded without fault. Only when the two nets are driven by opposite logic values is the net bridging likely to cause circuit failures.

For the net bridging, our concern is what logic values will be forwarded to the following logic from the bridging point. That is, the steady logic values of $d_1 \sim d_4$ are concerned, as the example in Figure 5 shows. To get their values, we derive the equivalent interconnect circuit as in Figure 6(a), where the resistance and capacitance are respectively modeled according to the physical layout after placement and routing. Although the driving signals of the two nets s_1 and s_2 change at circuit frequency, the

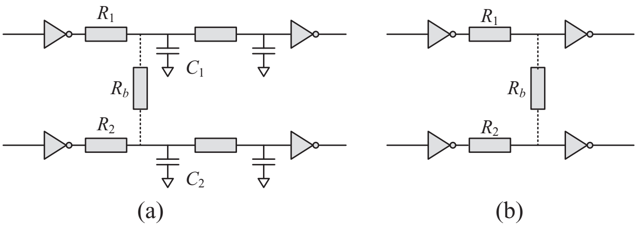


Fig. 6. The bridged circuit model [Gao et al. 2005].

transition time of the faulty signal is typically small compared to that of the clock period. As a result, we can statically analyze the circuit by ignoring the interconnecting and sink capacitances from Figure 6(a), according to the study in Gao et al. [2005], on a bridged circuit. Then, a resistance network with R_1, R_2, and R_b is left, as in Figure 6(b), whose resistance values are calculated by the physical architectural parameters and routing distances of the concerned wires from their respective driving blocks. Hence, the signal values at the faulty point can be calculated by a voltage dividing between Vdd and Gnd along the wires, and the logic values of $d_1 \sim d_4$ can be obtained accordingly.

In addition, the impact of bridging may vary along the affected wires. Still considering net s_1 in Figure 5, the logic values on points d_2 and d_3 are possible to flip due to voltage dividing, while d_1 may remain its value because it is nearer to its driving source at s_1. Similarly, d_4 may also be flipped depending on its driving distance to its source s_2. This behavior makes the bridging fault quite different from that of the LUTs. It is likely that a single wire is decomposed to carry different logic values, and multiple faults may be injected into the circuit at one time.

3.2.4. SEU on Unidirectional Routing. For inter-CLB routing, modern FPGAs have shifted from bidirectional routing towards unidirectional routing architecture. In this new routing architecture, connection boxes and switch boxes employ directional wires to route signals and use MUXes for signal interconnection. As a result, the fault behavior in this unidirectional routing is different from that of bidirectional pass transistors. In this article, we consider the strict use of single-driver directional routing [Lemieux et al. 2004], which is briefly explained as follows.

The evolution towards single-driver directional routing architecture changes the micro-architecture of connection boxes and switch boxes. That is, the CLB outputs can only be connected onto the input MUXes of the wires that begin nearby. The CLB inputs receive signals from tracks passing through the neighboring connection boxes. Accordingly, input MUXes of wires in switch boxes select candidate drivers, including both interconnects within the switch box and CLB outputs nearby. Figure 7 briefly illustrates the single-driver directional routing architecture.

Once affected by the SEU, one of the encoded CRAM bits for routing MUXes will flip its value and thus select an erroneous input pin onto the MUX output, similar to the affected local routing seen in Figure 7. The erroneous signal may be further propagated to primary outputs to be finally observed.

The unidirectional routing architecture is mainly made up of MUXes, which raises the signal selection fault instead of the open fault or short (bridging) fault in the conventional bidirectional routing when an SEU occurs. We will investigate the fault characteristics in detail in Section 5.

4. PROPOSED SEU EVALUATION FRAMEWORK

We now present our SEU fault evaluation framework in this article, based on the parameterized architectural description given in Section 3. Our framework performs

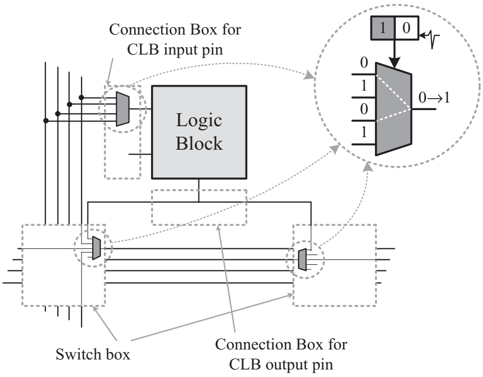


Fig. 7. An SEU on a connection box and switch box in a unidirectional routing architecture.

the fault analysis on each CRAM bit under the single fault assumption, that is, at any time, at most one SEU fault exists in the FPGA. This is reasonable, because compared to SEUs, simultaneous multiple-bit SEUs (MBU) have less chances to happen in current FPGAs [Chapman, 2009].

4.1. Fault Sensitivity Evaluation

Previous work has studied the failure sensitivity due to an SEU on an LUT CRAM bit by introducing the metric of *criticality*. In this article, we leverage the concept and extend it onto interconnects, presenting a unified metric of criticality defined as follows.

Definition. Given a circuit C with n primary inputs and a set of input vectors X, the criticality c_b of one CRAM bit b, which configures an FPGA element like an LUT or a routing element, is the probability that one or more errors can be observed at the primary outputs due to an SEU on that bit.

$$c_b(x) = \frac{1}{2^n} \left| \left\{ x | \mathrm{C}_b(x) \neq \mathrm{C}_{\bar{b}}(x) \right\} \right| , \tag{1}$$

where $x \in (0, 1)^n$ is one of the vectors in the exhaustive input set X. $\mathrm{C}_b(x)$ is the circuit output without SEU fault under x, and $\mathrm{C}_{\bar{b}}(x)$ is the circuit output when bit b is flipped. When $\mathrm{C}_{\bar{b}}(x)$ and $\mathrm{C}_b(x)$ mismatches, the system is said to encounter failures which should be attributed to bit b. So, by identifying visible errors by applying input set X on the circuit, the metric of criticality reveals the possibility of an SEU on a CRAM bit that results in FPGA failures, which is generally acknowledged as the failure rate by an SEU.

Ideally, the criticality of bit b should be obtained by exhausting all the 2^n permissible vectors in a complete input set X, which is very time consuming. In practice, it can be approximated by Monte Carlo-based simulation of as many as K times, which can provide good accuracy, as implied in Luckenbill et al. [2010]. In addition, it applies to any circuit element as long as it has CRAM bits in it, as in Equation (2). For example, we can quantify the chip failure rate from each CRAM bit in the circuit by aggregating the criticality of one bit and the probability of SEU that happens on that bit.

$$\sum_{b \in elem} c_b \cdot Pr(b \xrightarrow{SEU} \bar{b}) \tag{2}$$

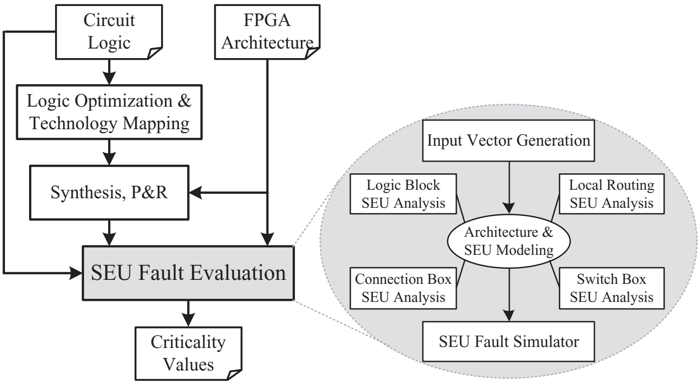


Fig. 8. Overview of our SEU evaluation framework.

4.2. Framework Overview

Figure 8 illustrates the flow of our proposed SEU fault evaluation framework for SRAM-based FPGAs. Starting from the given circuit netlist, it first applies logic optimization and technology mapping onto the LUTs. The mapped circuit is packed into logic blocks then placed and routed by physical design tools. Our fault analysis starts right after the placement and routing, taking the post-layout circuit, FPGA architectural file, and circuit logical function as inputs. Based on the metric of criticality, our framework evaluates the failure rate or sensitivity of each CRAM bit in various elements according to their fault behavior, as described in Section 3, for example, the open and short faults in bidirectional routing or the selection faults in unidirectional routing. After the fault analysis, SEU-induced faults are injected into the simulator, which then performs logic-level simulation on the faulty circuit and calculates the criticality for each bit automatically.

As an important evaluation step towards robust FPGA design, our framework can identify the most failure-sensitive circuit element and evaluate the applicability of various fault mitigation schemes. This evaluation can be applied as early as possible to be helpful during design time. Based on physical layout information, our framework is able to reveal the failure sensitivity for each CRAM bit in the FPGA circuit that is vulnerable to SEUs. In addition, the framework is universal and flexible to different FPGA architectures by adding new micro-architectural descriptions of the circuit element concerned. As a result, we envision that by shedding light on the hidden relation between CRAM bits and FPGA functional failures, our proposed framework will be helpful in designing more robust FPGA circuits, architectures, and synthesis algorithms.

5. SEU CHARACTERISTICS ON ARCHITECTURES

In the experiments, the ten largest MCNC combinational circuits are used as our test benchmark circuits with their statistics shown in Table I. For these circuits, we first apply logic optimization and technology mapping to 4- and 6-input LUTs (LUT size k) using the Berkeley ABC tool to represent the most popular used LUT input sizes in practice. The mapped circuits are packed by different logic block sizes (cluster size N) of 4, 6, 8, and 12 by the T-VPack tool. As a result, their combinations cover eight different architectural settings representing different cases, like smaller LUT with larger cluster, or larger LUT with smaller cluster, and some other common settings.

Table I. Size Statistics for the 10 MCNC Benchmark Circuits

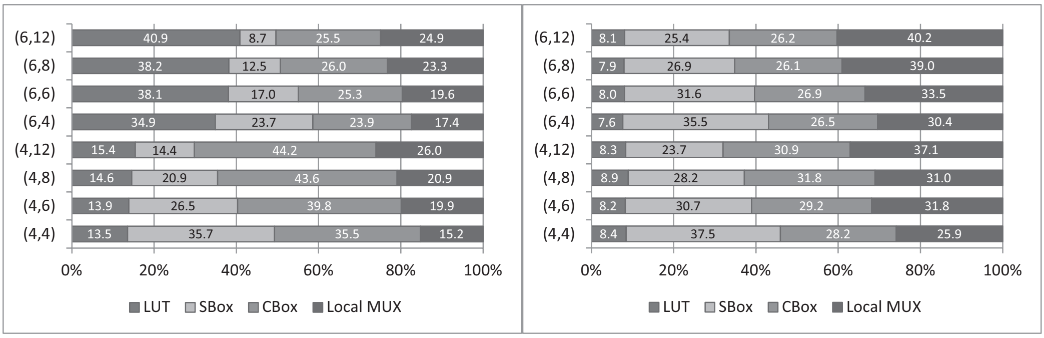
Circuit	# Gates	# Inputs	# Outputs
ex5p	527	8	63
apex4	722	9	19
misex3	735	14	14
alu4	730	14	8
ex1010	850	10	10
apex2	942	39	3
seq	1,020	41	35
des	1,498	256	245
spla	2,237	16	46
pdc	2,326	16	40

Then, the circuit under each case is placed and routed by the VPR tool [Luu et al. 2009] for minimum dimension and routing channel width. That means it generates the FPGA array as compactly as possible without involving extra unused bits that exceed the actual need of the circuit.

In this experiment, we characterized the SEU fault with respect to different circuits and CLB architectures. The hardware model and detailed SEU behavior on them have been discussed in Section 3. For the criticality calculation, we performed Monte Carlo simulation of 10K vectors, which consumes an acceptable runtime and provided relatively accurate estimations of the criticality values, according to the study in Luckenbill et al. [2010]. The CRAM bits in all logic and routing resources are evaluated based on their physical information obtained after placement and routing, which contribute to the majority of the configuration bits in an FPGA. There are other CRAM bits with much smaller numbers, but may configure clocks, resets, or other modules for control. It will be our future work to model their diversified behavior when affected by an SEU. We suppose a uniform distribution of the probabilities for each bit to go faulty, that is, $Pr(b \xrightarrow{SEU} \bar{b})$ are the same in Equation (2) for all the CRAM bits. In this way, we can simplify our SEU evaluation by focusing on their sensitivity to failure. In this section, we will first report the SEU characterization from different perspectives. Note that the experimental results are based on the averaged data of the ten benchmark circuits if no circuit name is explicitly specified.

5.1. Criticalities under Different CLB Architectures

Under bidirectional routing with different CLB architectures, Figure 9(a) shows the proportion of CRAM bit numbers in different circuit elements, and Figure 9(b) shows the proportion of their criticality values. As a brief overview of the two plots, several observations can be made. (1) The routing resources hold the majority of the CRAM bits, from about 61% to 87%, while contributing even more in total criticality, over 90% in these cases. This means that functional failure is most likely due to routing rather than LUT. (2) In terms of criticality, there is no single circuit element dominating the overall criticality. However, local routing MUXes have a larger proportion, while the proportion of their CRAM bits is relatively small compared to other elements. This indicates that they are the most failure-sensitive elements in an FPGA. (3) Increasing LUT size k and cluster size N increases LUT bits, but their criticality proportion is nearly the same, that is, less than 10% for all the cases. (4) With the increasing LUT size k and cluster size N, local routing MUXes contribute more in total criticality, because both k and N enlarge the number and the size of local routing MUXes and shrink the global routing network at the same time.



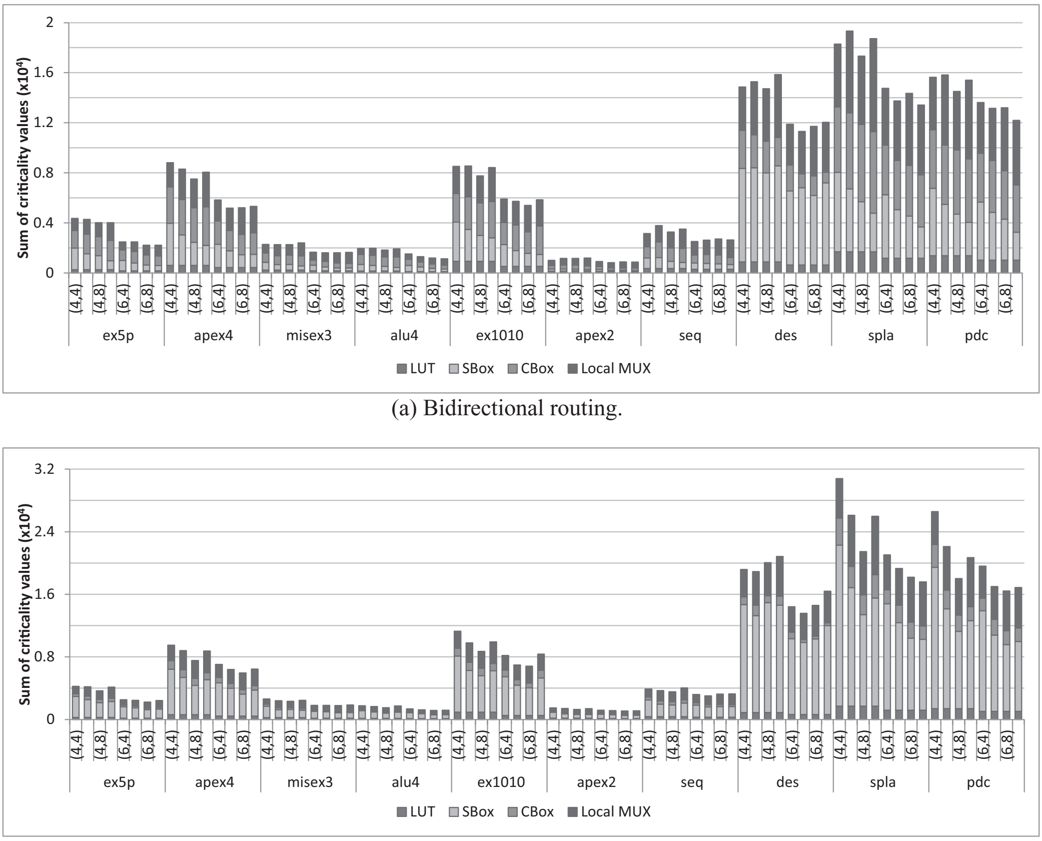
(a) (b)

Fig. 9. In bidirectional routing, (a) CRAM bits and (b) total criticality proportions from different circuit elements under different CLB architectural settings.

We further show Figure 10(a) for a detailed view of the criticality values from different circuit elements. The x-axis lists all the circuits under test in different architectural settings, and the y-axis gives the summed criticality value for each case. From the figure, one can see that each circuit presents significantly different failure sensitivities due to their inherent logic. Moreover, failure sensitivities of the same circuit under different settings may also vary. It is interesting to note that a larger LUT input size k provides a notable reduction of the total criticality value, because a larger LUT input size k shrinks the network dimension, which helps to reduce the routing CRAM bits that are more vulnerable to failures. In contrast, cluster size N balances the impacts of switch boxes versus connection boxes and local routing MUXes, while the total criticality values of cases with medium cluster size N are generally lower than other cases of the same LUT size, as seen from the plot.

We also report detailed criticality values from different elements under the unidirectional routing in Figure 10(b). One can see that it presents similar patterns to those of bidirectional routing. A most significant difference is that in unidirectional routing, the switch boxes hold the largest number of CRAM bits among the three routing elements and dominate the overall criticality as well. The reason (discussed in Section 3.2.4) is due to the micro-architecture in unidirectional routing, where CLB outputs go directly into switch box MUXes nearby and only CLB inputs are multiplexed in connection boxes.

Further, Figure 11 reports the SEU evaluation time for the unidirectional routing architecture to provide a sketch of the efficiency of our evaluation framework. The experiments are performed on a personal desktop with an Intel i3-CPU with 3G RAM. One can see that a smaller LUT input size k generally results in a longer evaluation time, because smaller LUT sizes require more interconnect and involve more CRAM bits on routing for evaluation. Note that in our evaluation framework, all the CRAM bits will be evaluated, each with 10K input vectors. Therefore it will be time consuming when a circuit has millions of CRAM bits. To accelerate the evaluation, several techniques can be applied on our framework in the future, such as packing the bit-wise input signals for parallel simulation and packing circuit nodes into larger blocks to avoid unnecessary node traversal when a block is fault free. In addition, due to the inherent parallelism in fault simulation, the framework can be easily deployed across different machines to gain further boost-up.



(a) Bidirectional routing.

(b) Unidirectional routing.

Fig. 10. Detailed criticality values for all the circuits under (a) bidirectional routing (b) unidirectional routing.

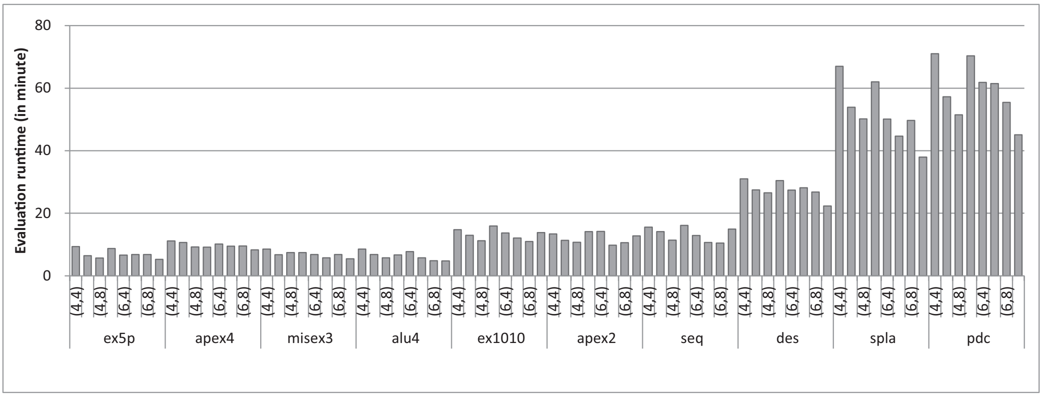


Fig. 11. Runtime for the SEU evaluation by our framework.

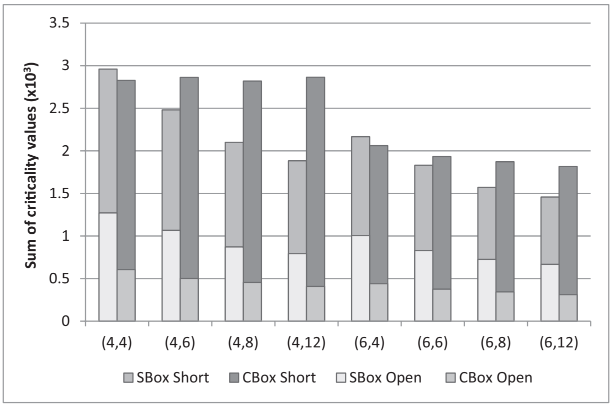


Fig. 12. SEU-induced open and short fault breakdown in switch boxes and connection boxes in bidirectional routing.

5.2. Criticality Breakdown in Bidirectional Routing

As discussed in Section 3.2, SEUs on the routing CRAM bits in bidirectional routing induce both open and short faults. Figure 12 shows their criticality breakdown. One can see that an SEU-induced short fault is more sensitive to functional failure than that of on open fault, almost 1.3x in switch boxes and 4.5x in connection boxes on average, in terms of their summed criticality values. This is because most of the time, switch boxes have utilization rates lower than 30%, and the rates of connection boxes are even lower. Typically, a lower utilization rate in a switch box or a connection box provides more possibility for a short fault. At the same time, the sum of short and open criticality values in switch boxes is notably reduced when the LUT input size k and CLB size N increase, because the sensitive CRAM bits in switch boxes rely completely on the dimension of the global routing network, which shrinks with an increasing LUT input size k and CLB size N.

6. SEU CHARACTERISTICS OF SYNTHESIS ALGORITHMS

Finally, we applied our fault evaluation framework on several resynthesis-based fault mitigation algorithms (as mentioned in Section 2.2), for example, IPD, IPF, and IPV, to see their improvements at the chip level. As resynthesis-based techniques, the three algorithms all can be performed within LUTs after placement and routing and preserve the circuit functionality without invoking physical resyntheses.

6.1. Failure Rate Reduction of Individual Algorithm

A brief introduction to the IPF, IPD, and IPV algorithms has been provided in Section 2.2. Here, Table II demonstrates the failure rate reductions by the three algorithms individually: on the LUT level for IPD and IPF and on interconnect for IPV. We also evaluated the chip-level failure rate reduction by taking the LUT and interconnect fault into account for the three algorithms. From the table, one can see that the three algorithms present different characteristics on fault mitigation. IPD significantly reduces the failure rate on LUTs (by around 75% on average), but the reduction on the chip level is limited (merely by 6%), since it only masks the fault within each LUT to prevent its propagation out of the LUTs. In contrast, IPF reduces the fault in fan-in cones by enhancing the logic masking capability in its sink LUT. As the don't care bits are not always available for logic masking, the reduction on LUTs is limited (by around 15% on average). Finally, IPF implicitly helps to reduce interconnect faults con-

Table II. Failure Rate Reduction by Individual IPD, IPF, and IPV Algorithms

Circuit	IPD Reduction		IPF Reduction		IPV Reduction	
	LUT (%)	Chip (%)	LUT (%)	Chip (%)	Int. (%)	Chip (%)
ex5p	66.47	3.26	6.80	14.86	47.34	43.82
apex4	78.21	8.40	13.32	12.31	79.62	45.05
misex3	82.07	7.55	26.09	27.61	66.49	60.00
alu4	68.78	5.14	22.38	17.80	49.80	74.09
ex1010	76.57	6.44	9.29	12.39	86.59	64.54
apex2	85.69	7.77	13.00	9.85	73.36	64.03
seq	82.51	8.26	18.88	14.52	70.60	79.76
des	28.85	1.75	10.50	6.25	8.37	8.26
spla	85.80	6.36	14.57	17.53	92.88	87.13
pdc	88.41	6.51	13.13	15.17	92.01	86.32
Avg.	74.34	6.14	14.79	14.83	66.71	61.30

Table III. Failure Rate Reduction by Combined Algorithms of IPD, IPF, and IPV

Circuit	IPF+IPD Reduction		IPF+IPV Reduction		IPD+IPV Reduction		IPF+IPD+IPV Reduction	
	LUT(%)	Chip(%)	LUT(%)	Chip(%)	LUT(%)	Chip(%)	LUT(%)	Chip(%)
ex5p	62.00	17.95	7.00	48.44	66.47	48.68	62.00	52.55
apex4	66.63	15.60	13.29	75.17	78.21	79.49	66.63	79.79
misex3	75.82	33.02	26.12	68.59	82.07	67.93	75.82	74.12
alu4	63.81	20.58	22.53	55.36	68.78	51.53	63.81	59.87
ex1010	59.03	14.59	9.24	80.51	76.57	85.84	59.03	86.37
apex2	85.26	15.76	12.52	70.41	85.69	74.68	85.26	78.29
seq	79.31	25.59	19.26	69.47	82.51	71.67	79.31	76.78
des	31.87	7.80	10.13	10.83	28.85	9.26	31.87	12.49
spla	70.94	20.49	14.48	87.69	85.80	92.44	70.94	92.71
pdc	70.64	24.90	13.05	86.51	88.41	91.79	70.64	92.38
Avg.	66.53	19.63	14.76	65.30	74.34	67.33	66.53	70.53

currently in the fan-in cones, thus obtaining a chip-level failure rate reduction around 15% on average. This implies that an LUT fault mitigation technique may also help to improve interconnect due to their interaction, but a more effective technique still needs further investigation. In contrast, IPV focuses on interconnect fault and significantly reduces the interconnect failure rate by about 67%. Although it does not improve faults in LUTs, it achieves more failure rate reduction on the chip level, as we have seen that interconnect contributes to the majority of the faults in FPGA.

6.2. Combined Algorithms and Interaction

Then, several combinations of the three algorithms are evaluated to investigate the interactions between them to further boost their fault mitigation capability. We evaluated the combinations of IPF+IPD, IPF+IPV, IPD+IPV, and IPF+IPD+IPV, where the algorithms are applied on the circuit as indicated by their order, and the results are shown in Table III.

For IPF+IPD, one can see that the two algorithms are not orthogonal. First, the interaction between them degrades the failure rate reduction on LUTs more than with an individual IPD, from about 74% down to 67% on average. This is because IPF may reduce the on/off set criticality difference on LUTs, which is an indicator provided in Lee et al. [2010] to show how much IPD can reduce LUT fault. The "on" (resp. "off") set is the CRAM bit set with logic "1" (resp. "0"). In general, a higher on/off set criticality

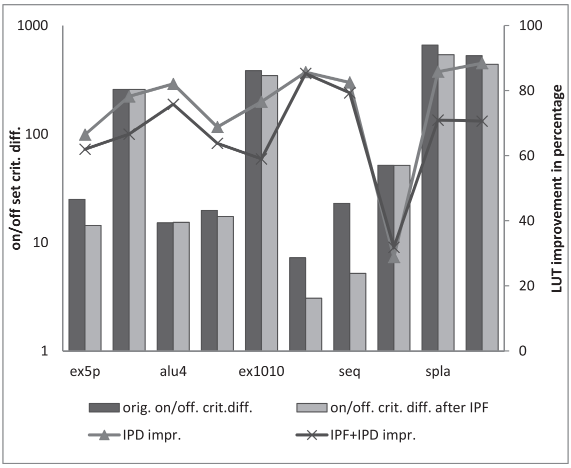


Fig. 13. The on/off set criticality differences versus LUT criticality improvements by IPD and IPF+IPD.

difference indicates more potential improvement that IPD can provide. We further plotted the on/off set criticality differences for the ten circuits in Figure 13, where most of the differences are reduced after applying IPF, and thus the failure rate reductions by IPF+IPD are degraded. Second, compared with individual IPF, an extra failure rate reduction of several percent (about 5% on average) is observed on the chip level after applying IPD on IPF. This is due to IPD further reducing the fault on LUTs, which in turn improves chip reliability. Third, both algorithms present limited improvement on the chip level (less than 20% on average), since neither of them considers interconnect fault explicitly. This experiment reveals that interconnect is more important in fault mitigation, and in order to develop more advanced fault mitigation techniques in the future, the faults on LUTs and interconnects should be tuned together to improve the circuit fault tolerance to the greatest extent.

For the combined algorithms of IPF+IPV and IPD+IPV, one can see that the failure rate on the chip level can be reduced respectively by around 65% and 67% on average, which means reductions of 2.88x and 3.06x can be achieved. This is due to IPV explicitly considering interconnect fault. In addition, the IPV algorithm is completely orthogonal with IPD. That is, the failure rate reduction of LUT comes from IPD, while reduction of interconnect comes from IPV, because there is no interaction or coupling between the two algorithms. For the combination of IPF+IPV, IPV keeps the LUT fault reduction by IPF, while IPF helps to reduce the interconnect fault by another 4% on average (from 61.3% to 65.3%). This slight improvement is due to the implicit fault reduction by IPF, as previously explained.

For the combination of IPF+IPD+IPV, the experiment results also confirm our understanding of the three algorithms. That is, (1) since IPV has no improvement on LUT fault, the failure rate reduction on LUT is the same as that of IPF+IPD; (2) since the interconnect fault is explicitly considered by IPV, the failure rate reduction on the chip level is higher than IPF+IPD; (3) as IPF implicitly helps to reduce interconnect fault in fan-in cones for each LUT, the failure rate reduction is higher than IPD+IPV in all cases. Although this combination covers fault mitigation both on LUTs and interconnects, the improvement is not orthogonal, because the current combination simply neglects the interaction between them, for example, IPF with IPD, IPF with IPV, which overlaps in optimization. This reveals that in order to improve the fault tolerance on the chip level, future fault mitigation algorithms should be concerned not only about the fault in LUTs and interconnects, but also their interactions. We will further investigate the interactions and make more intelligent integration of the in-place, resynthesis-based fault mitigation algorithms, for example, IPD, IPF, IPV, and their variants, to jointly improve the chip fault tolerance.

7. CONCLUSIONS AND FUTURE WORK

A comprehensive SEU fault evaluation framework for SRAM-based FPGAs has been proposed in this article. Based on the post-layout FPGA application, the proposed framework is capable of quantifying the SEU fault-induced functional failures for exact configuration bits in various circuit elements, such as LUTs, connection boxes, switch boxes, and local routing multiplexers. In this article, the SEU fault was characterized by several existing FPGA architectures differentiated by CLB sizes, LUT sizes, and routing structures. At the same time, several logic resynthesis-based fault mitigation algorithms and their combinations were evaluated to see the improvement on the chip level. Detailed fault characteristics from various perspectives can be found in our experiments.

In the future, by identifying sequential feedbacks, we can also apply our approach to sequential circuits. Besides, commercial architectures will be modeled to make this framework more general for architectural and synthesis algorithm evaluation with respect to SEU fault in FPGAs.

Our SEU fault evaluation framework provides detailed information for identifying the most critical configuration bits or circuit elements to develop new fault mitigation algorithms. We envision that our fault evaluation framework will be used to cast more useful insights for the design of more robust FPGA circuits, architectures, and better synthesis algorithms.

REFERENCES

Asadi, G. and Tahoori, M. B. 2005. Soft error rate estimation and mitigation for SRAM-based FPGAs. In *Proceedings of the ACM/SIGDA 13th International Symposium on Field-Programmable Gate Arrays*. 149–160.

Asadi, H., Tahoori, M. B., Mullins, B., Kaeli, D., and Granlund, K. 2007. Soft error susceptibility analysis of SRAM-based FPGAs in high-performance information systems. *IEEE Trans. Nucl. Sci.*, 54, 2714–2726.

Bellato, M., Bernardi, P., Bortolato, D., Candelori, A., Ceschia, M., Paccagnella, A., Rebaudengo, M., Sonza Reorda, M., Violante, M., and Zambolin, P. 2004. Evaluating the effects of SEUs affecting the configuration memory of an SRAM-based FPGA. In *Proceedings of the Conference on Design, Automation and Test in Europe*. 584–589.

Betz, V., and Rose, J. 1997. VPR: A new packing, placement and routing tool for FPGA research. In *Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications*. 213–222.

Chapman, K. 2010. SEU strategies for Virtex-5 devices.

Feng, Z., Hu, Y., He, L., and Majumdar, R. 2009. IPR: In-place reconfiguration for FPGA fault tolerance. In *Proceedings of the International Conference on Computer-Aided Design*. 105–108.

Feng, Z., Jing, N., Hu, Y., and He, L. 2011. IPF: In-place x-filling to mitigate soft errors in SRAM-based FPGAs. In *Proceedings of the 21th International Workshop on Field-Programmable Logic and Applications*. 482–485.

Gao, H., Yang, Y., Ma, X., and Dong, G. 2005. Testing for resistive shorts in FPGA interconnects. In *Proceedings of the 6th International Symposium on Quality of Electronic Design.* 159–163.

Golshan, S. and Bozorgzadeh, E. 2007. Single-event-upset (SEU) awareness in FPGA routing. In *Proceedings of the 44th Annual Design Automation Conference.* 330–333.

Graham, P., Caffrey, M., Zimmerman, J., Sundararajan, P., Johnson, E., and Patterson, C. 2003. Consequences and categories of SRAM FPGA configuration SEUs. In *Proceedings of the International Conference on Military and Aerospace Programmable Logic Devices*.

Héron, O., Arnaout, T., and Wunderlich, H.-J. 2005. On the reliability evaluation of SRAM-based FPGA designs. In *Proceedings of the International Conference on Field Programmable Logic and Applications*.

Hu, Y., Feng, Z., He, L., and Majumdar, R. 2008. Robust FPGA resynthesis based on fault-tolerant boolean matching. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. 706–713.

Jing, N., Lee, J.-Y., Feng, Z., He, W., Mao, Z., Wen, S.-J., Wong, R., and He, L. 2011a. Quantitative SEU fault evaluation for SRAM-based FPGA architectures and synthesis algorithms. In *Proceeding of the 21th International Workshop on Field-Programmable Logic and Applications*. 282–285.

Jing, N., Lee, J.-Y., He, W., Mao, Z., and He, L. 2011b. Mitigating FPGA interconnect soft errors by in-place LUT inversion. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. 582–586.

Johnson, E., Caffrey, M., Graham, P., Rollins, N., and Wirthlin, M. 2003. Accelerator validation of an FPGA SEU simulator. *IEEE Trans. Nucl. Sci.*, 50, 2147–2157.

Jose, M., Hu, Y., Majumdar, R., and He, L. 2010. Rewiring for robustness. In *Proceedings of the 47th Design Automation Conference*. 469–474.

Krishnaswamy, S., Plaza, S. M., Markov, I. L., and Hayes, J. P. 2007. Enhancing design robustness with reliability-aware resynthesis and logic simulation. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. 149–154.

Lee, J.-Y., Feng, Z., and He, L. 2010. In-place decomposition for robustness in FPGA. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*. 143–148.

Lemieux, G., Lee, E., Tom, M., and Yu, A. 2004. Directional and single-driver wires in FPGA interconnect. In *Proceedings of the Conference on Field-Programmable Technology*. 41–48.

Luckenbill, S., Lee, J.-Y., Hu, Y., Majumdar, R., and He, L. 2010. RALF: Reliability analysis for logic faults: an exact algorithm and its applications. In *Proceedings of the Conference on Design, Automation and Test in Europe*. 783–788.

Luu, J., Kuon, I., Jamieson, P., Campbell, T., Ye, A., Fang, W. M., and Rose, J. 2009. VPR 5.0: FPGA cad and architecture exploration tools with single-driver routing, heterogeneity and process scaling. In *Proceeding of the ACM/SIGDA 17th International Symposium on Field Programmable Gate Arrays*. 133–142.

Rebaudengo, M., Reorda, M. S., and Violante, M. 2002. Simulation-based analysis of SEU effects on SRAM-based FPGAs. In *Proceedings of the 12th International Conference on Field-Programmable Logic and Applications*. 607–615.

Reddy, E. S. S., Chandrasekhar, V., Sashikanth, M., Kamakoti, V., and Vijaykrishnan, N. 2005. Detecting SEU-caused routing errors in SRAM-based FPGAs. In *Proceedings of the 18th International Conference on VLSI Design*. 736–741.

Smith, A. M., Constantinides, G. A., Wilton, S., and Cheung, P. 2009. Concurrently optimizing FPGA architecture parameters and transistor sizing: Implications for FPGA design. In *Proceedings of the Conference on Field-Programmable Technology*. 54–61.

Sterpone, L. and Violante, M. 2006. A new reliability-oriented place and route algorithm for SRAM-based FPGAs. *IEEE Trans. Comput.*, 55, 732–744.

Xilinx Inc. 2010. Virtex-6 Family Overview.

Received October 2011; revised May 2012; accepted July 2012