This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A Scalable Dependability Scheme for Routing Fabric of SRAM-Based Reconfigurable Devices

Sadegh Yazdanshenas, Hossein Asadi, *Member, IEEE*, and Behnam Khaleghi

***Abstract*— With the continual scaling of feature size, system failure due to soft errors is getting more frequent in CMOS technology. Soft errors have particularly severe effects in static random-access memory (SRAM)-based reconfigurable devices (SRDs) since an error in SRD configuration bits can permanently change the functionality of the system. Since interconnect resources are the dominant contributor to the overall configuration memory upsets in SRD-based designs, the system failure rate can be significantly reduced by mitigating soft errors in routing fabric. This paper first presents a comprehensive analysis of SRD switch box susceptibility to short and open faults. Based on this analysis, we present a dependable routing fabric by efficiently employing asymmetric SRAM cells in configuration memory of SRDs. The proposed scheme is highly scalable and capable of achieving any desired level of dependability. In the proposed scheme, we also present a fault masking mechanism to mitigate the effect of soft errors in the routing circuitry. A routing algorithm is also proposed to take the advantage of the proposed routing fabric. Experimental results over the Microelectronics Center of North Carolina benchmarks show that the proposed scheme can mitigate both single and multiple event upsets in the routing fabric and can reduce system failure rate orders of magnitude as compared with the conventional protection techniques.**

***Index Terms*— Asymmetric static random-access memory (SRAM), dependability, routing fabric, soft errors, SRAM-based reconfigurable devices (SRDs).**

I. Introduction

STATIC random-access memory (SRAM)-based reconfigurable devices (SRDs) are a popular platform for fast prototyping of digital systems in a wide range of application domains due to their prominent features such as flexibility, high performance, nonrecurring engineering cost, and fast time-to-market [1]. In application domains where human intervention in system maintenance is very difficult or impossible, SRDs can be used to remotely reconfigure the entire system making system upgrade/alteration very flexible. Such flexibility, however, comes at several costs including power, area, performance, and reliability penalties as compared with application-specific integrated circuit (ASIC) counterparts.

One major concern limiting the widespread usage of SRDs in enterprise and safety-critical applications is susceptibility of these devices to soft errors. These errors are transient errors occurring in combinational and sequential elements attributed to alpha particles and atmospheric neutrons. Soft errors have become a major concern in deep-submicrometer technologies due to extremely low switching charges. SRDs are especially more sensitive to soft errors than their ASIC counterparts since a soft error in configuration memory can permanently invalidate the device state unless it is reconfigured by error-free configuration bits [1], [2]. This type of errors is the most common error type in SRDs since the majority of memory cells are the configuration bits.

Traditional fault-tolerant methods such as triple modular redundancy (TMR) [3]–[5] or fine-grained redundancy [6] techniques can effectively mitigate the effect of soft errors in logic blocks (LBs) used in SRDs. However, routing faults that are the cause of over 80% of the total configuration memory errors [1] cannot simply be addressed by the traditional methods such as TMR [4], [7], [8] unless placement and routing algorithms are modified such that no common routing resources are shared among replica modules [9]. In addition, since interconnect is the major contributor to the device area and performance, redundancies applied to routing are not suitable for cost-sensitive applications. This raises the need for low-overhead customized fault-tolerant methods for SRDs. For application domains where cost is not a main priority such as safety-critical or enterprise applications, such low-overhead fault-tolerant methods can still be useful and can be additive to high overhead error detection/correction schemes to further increase dependability.

This paper first presents a comprehensive study to investigate routing fabric susceptibility to open and short faults. The proposed study characterizes the susceptibility of different patterns of switch boxes (SBs) to soft errors by extracting the expected number of sensitive-one/zero configuration bits. A particle strike on a sensitive-one/zero configuration bit can result in open and short faults, respectively. The proposed study reveals that the expected number of sensitive zero configuration bits for different SB patterns has significant variation (ranging from 0 up to 4 bits per SB). It is also demonstrated that some SB patterns with the highest frequency in designs have the least susceptibility to soft errors. Finally, this paper shows that unlike the assumption made in previous studies [10]–[12], optimizing the configuration memory bits for logical value of zero does not efficiently reduce the circuit failure rate.

Based on the proposed study, we present a scalable dependability scheme for routing fabric of SRDs using a combination of zero-optimized and one-optimized SRAM cells.

Manuscript received August 11, 2013; revised April 16, 2014; accepted July 9, 2014.

The authors are with the Department of Computer Engineering, Sharif University of Technology, Tehran 14588-89694, Iran (e-mail: syazdanshenas@ce.sharif.edu; asadi@sharif.edu; behnam_khaleghi@ce.sharif.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2014.2344051

1063-8210 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A zero- or one-optimized SRAM cell provides significant immunity against particle strikes preventing SRAM bit-flip to the logical value of one or zero, respectively. In the proposed scheme, a main channel employing zero-optimized SRAM cells is used to route the original nets, while a redundant channel employing one-optimized SRAM cells is used to route the replica routing nets. Using zero-optimized SRAM cells in the main channel, the probability of short faults is minimized. Hence, nets in the main channel can be highly congested without any concern regarding short faults. On the other hand, by employing one-optimized SRAM cells in the redundant channel, the probability of open faults is significantly reduced. To avoid short faults, nets are routed such that no zero-sensitive configuration bit exists in the redundant channel. To mitigate the effect of soft errors in the main and redundant channels, we also propose a fault masking mechanism such that an erroneous value is masked and no error recovery will be required. Finally, a routing algorithm to take advantage of the proposed study and the proposed routing fabric is proposed.

The proposed routing scheme has been implemented over Microelectronics Center of North Carolina (MCNC) benchmarks and it is compared with the conventional mitigation techniques. The results show that the proposed scheme can mitigate both single and multiple event upsets in the routing fabric at a proportional increase in the area of routing fabric while performance remains intact. The experimental results also demonstrate that the proposed scheme reduces system failure rate up to six and eight orders of magnitude as compared with full TMR and five modular redundancy system (5MR) schemes, respectively, with the same area overhead.

The organization of the paper is as follows. In Section II, we briefly review background on SRDs interconnect. In Section III, related work is discussed. In Section IV, we propose our routing fabric soft error susceptibility analysis. In Section V, the proposed routing scheme is presented. In Section VI, experimental results are detailed. Finally, Section VII concludes the paper and offers future work.

II. Background

Island style SRD architecture is a popular architecture used in many of the commercial SRDs such as Xilinx Virtex 4 [13] and Altera Stratix II [14]. This architecture, named after its routing style, is also commonly used in academic SRD computer-aided design tools such as versatile place and route (VPR) [15]. This architecture mainly consists of a pool of interconnect resources surrounding a 2-D array of LBs. In the state-of-the art SRDs, there are also various full-custom blocks available as logic resources such as digital signal processing processors, block random-access memories, and multipliers.

The simplified diagram of island style SRDs is shown in Fig. 1. As shown in Fig. 1, there are three main programmable resources in island style SRDs: LBs, connection blocks (CBs), and switch matrices (SMs). A LB consists of either multiple reconfigurable logic elements (LEs) or a prefabricated full-custom complex block performing a specific operation

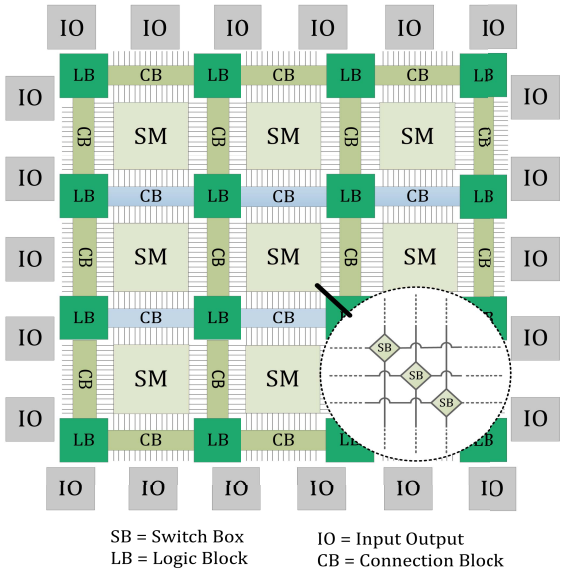


Fig. 1. Island style SRD architecture.

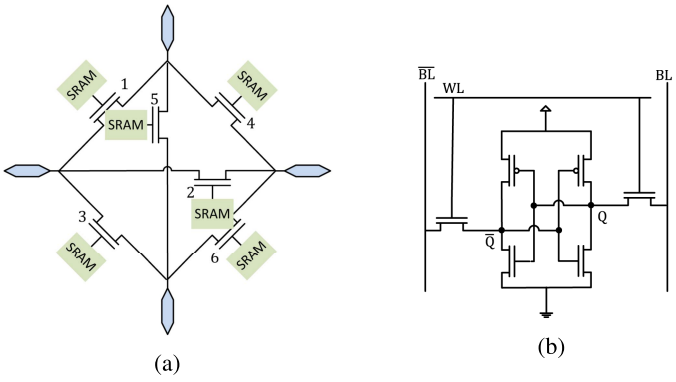


Fig. 2. SB architecture. (a) Typical SB with flexibility of three programmed by six SRAMs. (b) Typical SRAM used as configuration memory in SRDs.

such as multiplication. A typical LE is made of a look-up table (LUT), a reconfigurable multiplexer, and a sequential element such as flip-flop (FF). LBs are connected to each other and to the input/output pins via the interconnect resources. They are also connected to the routing fabric via CBs. At the intersection of horizontal and vertical routing channels, SMs perform signal routing, providing a flexible connectivity between these channels. As shown in Fig. 1, a SM consists of multiple SBs that come with different flexibilities, topologies, and directions. As an example, a SB with flexibility of three is shown in Fig. 2(a). The SRAM cells shown in Fig. 2 can be programmed to change the configuration (or pattern) of the SB. We have numbered the pass transistors in Fig. 2 for further references regarding SB patterns. A typical implementation of a SRAM configuration memory is shown in Fig. 2(b).

When a SB gets affected by a soft error, the corresponding consequence depends on the pattern of the SB and the surrounding nets. Either two connected nets will open, which is called an *open fault* or two unconnected nets will connect, leading to a *short circuit*. In the latter case, there are two possible outcomes: 1) it is possible that these two nets are both

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

used nets and therefore a *short fault* will occur and 2) it is also likely that at least one of these two nets are not used, as a result, no signal value is affected. However, the net delay will be affected due to the capacitance effect of shortening two wires. This case is considered as a fault only if it causes a timing violation that is very unlikely for the capacitance of a single unused net to cause a timing violation by itself due to high number of used resources along the target net. However, this effect still has to be considered for timing critical nets. It is evident that the other two cases, i.e., open and short faults, can potentially affect the circuit functionality and cause a system failure.

III. Related Works

Previous work on dependability of SRDs can be broadly classified into two groups. The first group tries to improve device dependability by orders of magnitude using aggressive redundancy schemes such as TMR [3]–[5]. The second group provides enhanced dependability by low-overhead fault avoidance schemes [11], [16]. The first category is only applicable to safety-critical or enterprise applications where either cost is not a major concern or high levels of dependability are required. The second category, however, is suitable for cost-sensitive applications where the device dependability is of a secondary concern. Next we detail this classification.

A. High-Cost Dependable Architectures

An effective architecture to provide a highly dependable SRD is TMR equipped with scrubbing technique. TMR and scrubbing are two different schemes that can be used independently. However, these two are typically employed together to benefit from both schemes. In TMR, three replica modules work in parallel and a voter is used to route an error-free value to the system output. A scrubbing circuitry is also used to periodically read configuration bits and overwrite possible faulty configuration bits from an external soft-error-immune memory. Error correction of erroneous configuration bits in scrubbing can be also achieved by applying information redundancy to configuration bits [17], [18]. TMR equipped with scrubbing can significantly improve system dependability, however, it imposes more than 250% area overhead to the system [19]. In addition, the dependability of voters remain questionable as addressed in previous studies [20]. TMR, however, can still be improved by employing placement and routing mechanisms such that the common routing resources among replica modules are minimized [9]. To alleviate significant area overhead by the TMR scheme, it can be selectively applied to the most critical parts of a design. Such scheme, called partial TMR [21], [22], reduces the area overhead, but still some design parts remain unprotected.

Duplication With Comparison is another technique in this category that can detect errors by comparing the outputs of the main module with a redundant module. This technique has no recovery mechanism and additionally imposes at least 100% area overhead to the system. Using such technique, a recent study proposes relatively low-overhead correction scheme as compared with TMR [19]. This paper, however, uses backward recovery and may not be applicable to some real-time applications.

The main shortcoming of the techniques proposed in this category is that to achieve a desired level of dependability, one might have to pay significantly more overhead in terms of area and power consumption that is actually needed for the desired level of dependability resulting in over-designed system. For example, if the required dependability level relies between the TMR and 5MR, the latter has to be employed paying significantly additional unwanted overhead. This shortcoming will be addressed in this paper by providing a scalable dependability scheme that is capable of having adjustable overhead proportionate to the required dependability.

B. Low-Cost Dependable Architectures

This category is mainly concerned with low-cost techniques to reduce system failure rate. There are several techniques that try to reduce error rate by modifying placement and routing such that less configuration bits are vulnerable to soft errors [23]–[28]. The applicability of these techniques, however, is limited to LBs since majority of these techniques ignore the interconnect that is the most important part of SRDs in terms of error rate. LB modification can still improve the reliability of routing fabric to some extent. As an example, [29] tries to reduce interconnect errors through LUT manipulation.

There are also various techniques proposing a SB design to improve device dependability. The technique in [30] has proposed to reduce the flexibility of SB to mitigate short faults. The main drawbacks of this technique are increased delay, longer nets, increased channel width, and higher number of open faults, while routing flexibility is also reduced. Another technique is to reduce the number of vulnerable SRAM cells by decoding the configuration information [16]. This technique, however, affects both the total delay and area. Area overhead also results in increased soft error rate. However, since this increase is not due to configuration bits, such faults will be transient and will not invalidate the configuration memory. This results in a higher availability of the design since the error rate of the configuration bits is reduced. However, many erroneous momentarily results may be produced due to higher transient error rate and reliability may decline.

There are also few techniques that use asymmetric SRAM cells optimized for the logical value of zero in SRD configuration bits to reduce error rate [10]–[12]. The main motivation behind these techniques is the fact that the majority of configuration bits in SRDs are zero. This is mainly due to abundance of unused routing resources. These techniques, however, have neglected the fact that not all unused resources are critical in the design. As an example, an unused SB far from an active net is very unlikely to affect the functionality of the implemented design due to soft errors. On the other hand, all the ones in SBs are sensitive to single-event upsets (SEUs) and will cause an open fault in case of being affected by soft errors.

IV. Proposed Soft Error Susceptibility Analysis in SBs

As discussed earlier, the main motivation to employ asymmetric SRAM cell in SRD configuration bits is to use

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

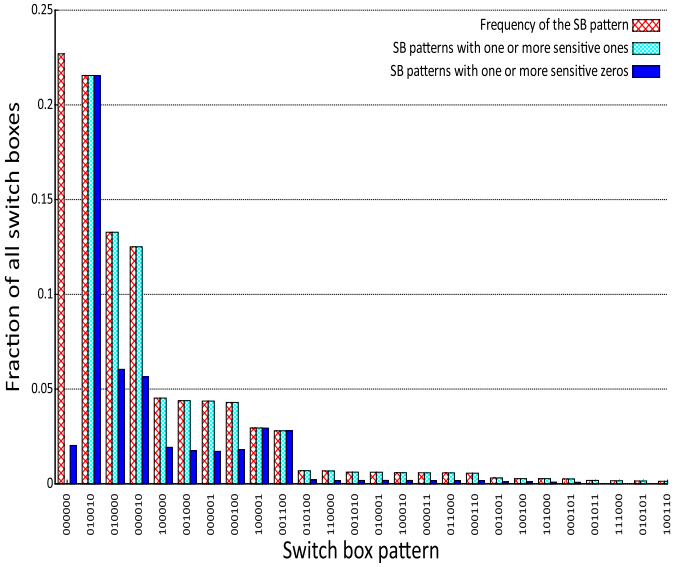


Fig. 3. Frequency of different SB patterns in MCNC benchmark and their susceptibility to soft errors.

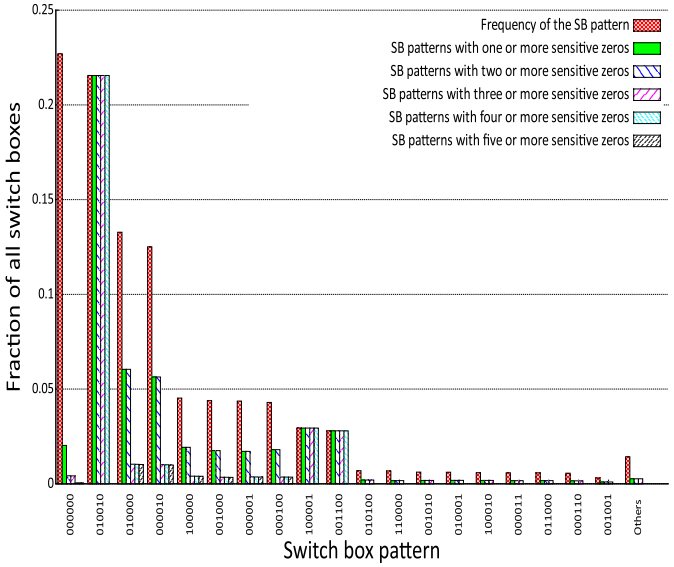


Fig. 4. Frequency of SBs with a minimum number of sensitive bits.

zero-optimized SRAM cells against soft errors since majority of configuration bits in SRDs have logical value of zero. This is especially effective for unused resources in routing and logic fabric. However, most of these unused memory cells even if affected by particle strikes, will not introduce any error in circuit functionality and hence such protection does not enhance the system dependability. We have carried out a set of experiments on 20 largest circuits of the MCNC benchmark and extracted the frequency of different SB patterns to see whether a soft error in these patterns will cause a malfunction in the circuit or not.

Fig. 3 reports the frequency of different patterns averaged over the MCNC benchmark circuitsthat are placed and routed on a minimum-size SRD with minimum channel width. The x-axis in Fig. 3 lists the most frequent patterns that have been numbered according to Fig. 2(a). Fig. 3 also reports how many of these SBs include at least one configuration bit that is sensitive to soft errors, leading to either an open or short fault. The middle and the right bars in Fig. 3 report the number of SBs with at least one *sensitive one* and one *sensitive zero* configuration bit for different SB patterns, respectively. A sensitive one (sensitive zero) configuration bit is a bit with the logical value of one (zero), in which a bit flip caused by a particle strike will change the system functionality. Sensitive one and sensitive zero configuration bits will lead to open and short faults, respectively.

The results reported in Fig. 3 reveal that although the majority of configuration bits are zero, the probability of the circuit being affected by soft errors could be very low depending on the SB pattern. For example, consider the 000000 pattern that is an all open SB. Although this pattern contributes to more than 22% of the total patterns, the probability that the system be affected by a bit-flip in this pattern is less than 2%. This is especially true for circuits not mapped to minimal area size, which is a typical case for SRDs. However, the results reported in Fig. 3 have been extracted for the minimum area size to avoid any over estimation of the above-mentioned observation.

To further investigate the susceptibility of different patterns of SBs to soft errors, we have examined how many bits of a SB pattern are sensitive zero. Fig. 4 reports the number of SBs with a minimum number of n sensitive zero bits, where n varies from one to five. Note here that we do not report sensitive one bits since by definition any configuration bit with a logical value of one is sensitive one. As can be observed in Fig. 4, patterns such as 010010 have considerable minimum number of n sensitive zero configuration bits, while a pattern such as 000011 has small number of n sensitive zero configuration bits ($n >= 1$). Therefore, those nets passing through SBs with patterns such as 010010 have much higher susceptibility to soft errors than those nets which are routed through SBs with a pattern such as 000011. Another observation from the results reported in Fig. 4 is that there are several patterns that have high frequency in SB such as 000000 but have less susceptibility to soft errors.

By investigating the soft error susceptibility of different patterns reported in Fig. 4, we have classified SBs into four different pattern types, as can be seen in Fig. 5. The first type of patterns is the case where two different nets are routed within the same SB. This SB pattern is very sensitive to soft errors since all configuration bits of the SB are either sensitive one or sensitive zero bits. Example patterns in the first type are 010010 and 100001. The second type of SB patterns is when a single net is routed in the SB with multiple outputs (also called multiple fan-outs). In case a bit-flip in an unused switch folds a net to itself, the unused switch is not sensitive zero. It is also possible that an unused switch connects a net to an unused pin. In this case when the unused bit is affected by a soft error, the outcome depends on the wire that the net is connected to. If that wire is used by another net, it will result in a short fault. If not, it will only add some capacitance to the net. Sample patterns in this category are 001010 and 000011.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

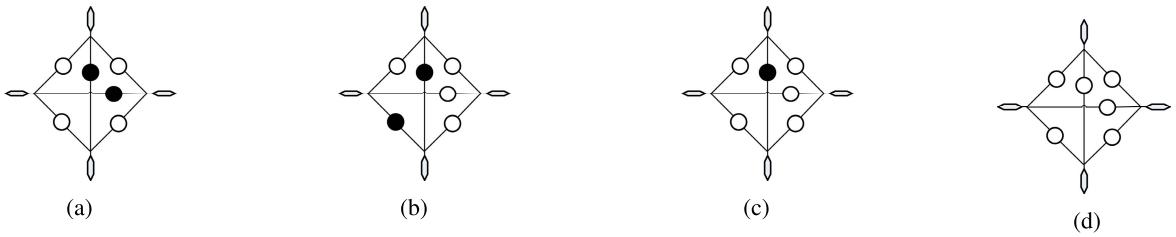
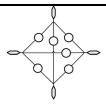
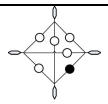
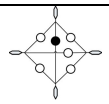
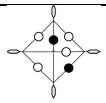
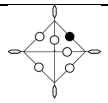
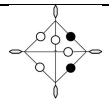
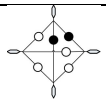
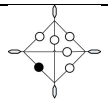
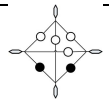
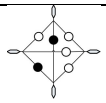
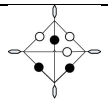
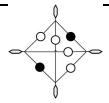
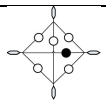
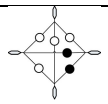
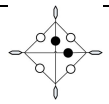
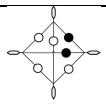
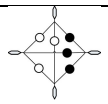
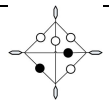
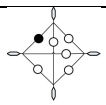
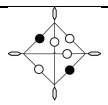
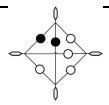
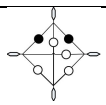
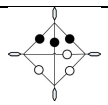
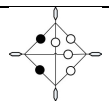
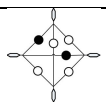
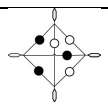


Fig. 5. Different types of SB patterns classified according to sensitivity to soft errors. (a) First type: two independent nets. (b) Second type: a single net with multiple fanouts. (c) Third type: a single net with one fanout. (d) Forth type: no routed net.

TABLE I

EXPECTED NUMBER OF SENSITIVE ZERO/ONE CONFIGURATION BITS PER NET PASSING THROUGH A SB (NSZ: EXPECTED NUMBER OF SENSITIVE ZERO CONFIGURATION BITS, NSO: EXPECTED NUMBER OF SENSITIVE ONE CONFIGURATION BITS)

Pattern	Pattern Code	NSZ	NSO	Type	Pattern	Pattern Code	NSZ	NSO	Type	Pattern	Pattern Code	NSZ	NSO	Type
	000000	0.066	0	4		000001	0.826	1	3		000010	0.913	1	3
	000011	0.856	2	2		000100	0.870	1	3		000101	0.950	2	2
	000110	0.839	2	2		001000	0.826	1	3		001001	0.935	2	2
	001010	0.879	2	2		001011	0	3	2		001100	4	2	1
	010000	0.913	1	3		010001	0.896	2	2		010010	4	2	1
	010100	0.865	2	3		010101	0	3	2		011000	0.875	2	2
	100000	0.893	1	3		100001	4	2	1		100010	0.919	2	2
	100100	1.041	2	2		100110	0	3	2		101000	0.987	2	2
	110000	0.745	2	2		111000	0	3	2					

The third type of SB patterns is when only one single net having a single fan-out is routed through a SB. In this case when a zero bit is affected by a soft error, similar to the discussion provided for the second pattern type, the outcome depends on the wire that the net is connected to. The fourth type of SB patterns is the all-zero pattern that is the most common pattern in designs due to unused routing resources. This type of SB is not very sensitive to soft errors since a short fault occurs only when two used nets in the neighbor SBs are connected to each other using this unused SB. This SB pattern can be found pretty much in areas of the design where routing resources are unused.

To validate our analytical characterization of SB patterns, we have experimentally investigated the number of sensitive one and sensitive zero bits for different SB patterns. To this end, we have extracted the sensitive bits of the MCNC benchmark circuits for different SB patterns. The experimental results have been reported in Table I. In Table I, we report the expected number of sensitive one and sensitive zero bits for different SB patterns. By the expected number of sensitive one and sensitive zero bits, we refer to the average number of sensitive one and sensitive zero bits, respectively. As can be observed in Table I, the first type of SB pattern has the largest number of sensitive bits,

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE II
EXPECTED NUMBER OF SENSITIVE CELLS AVERAGED OVER NETS IN MCNC BENCHMARK

Circuit	Zero Cells	One Cells	Circuit	Zero Cells	One Cells	Circuit	Zero Cells	One Cells	Circuit	Zero Cells	One Cells
Alu4	40.34	27.83	Des	34.42	24.28	Ex5p	46.63	31.72	S38417	30.31	20.78
Apex2	54.89	37.96	Diffeq	28.25	19.44	Frisc	61.66	41.25	S38584	33.85	22.71
Apex4	58.75	40.19	Dsip	30.68	22.44	Misex3	44.41	30.77	Seq	56.71	38.29
Bigkey	38.87	26.36	Elliptic	42.15	29.27	Pdc	84.59	56.53	Spla	72.68	48.77
Clma	67.83	45.55	Ex1010	70.39	47.74	S298	44.24	30.68	Tseng	20.63	14.62
Average	**48.12**	**32.86**									

while the fourth type of SB pattern has the least number of sensitive bits. It can also be observed from Table I that a single net with three fan-outs has zero number of sensitive zero configuration bits.

V. PROPOSED ROUTING FABRIC

The main aim of the proposed routing fabric is to use combination of zero- and one-optimized asymmetric SRAM cells in the routing fabric to enhance system dependability. Zero-optimized SRAM cells that offer significant immunity against particle strikes causing zero-to-one bit-flips are best suited for nets in congested areas where short faults are very likely to occur. The sensitivity of nets to short faults is mainly due to routing two or multiple nets in a shared channel or a shared SM where an unused switch can unwantedly connect the used nets together in the SM or in the routing channel. Hence, congested areas result in more short sensitive cells than noncongested areas since such scenario can occur more frequently in the congested areas. On the other hand, one-optimized SRAM cells are suitable for long nets that are more susceptible to open faults. The sensitivity of a net to open faults is mainly affected by the length of the net and the number of net fan-outs. If two nets are routed in a shared channel or routed individually in two different channels, the number of open-sensitive cells would be equal. That is congestion does not necessarily affect the number of open-sensitive cells.

We have obtained the expected number of sensitive cells in MCNC benchmark by analyzing all nets in each circuit, as reported in Table I. The expected number of open and short faults for MCNC benchmark circuits have been summarized in Table II. The summary results reveal that short faults contribute to more than 59% ($= 48.12/48.12 + 32.86$) of the total sensitive configuration bits, while the number of open faults is not negligible at all. Using the observations demonstrated in the previous section, we first propose a dependable routing architecture to protect both zero- and one-sensitive configuration bits. Then, a routing circuitry is proposed to mitigate the effect of soft errors in the routing fabric. Finally, we propose a routing algorithm to take advantage of the proposed routing fabric such that routing sensitivity is minimized.

A. Proposed Routing Architecture

In the proposed routing architecture, we divide the routing resources into two different channels. The main routing channel is used to route the entire nets and the other one is used as a redundant channel for rerouting the most sensitive nets of the design. The main routing channel is optimized for the logical value of zero to avoid short faults in congested routing areas. Existing placement and routing tools (such as VPR [15]) can perform compact routing. This provides the main routing channel the opportunity of being as compact as possible while having no susceptibility to short faults. Note the patterns with high number of zero-sensitive bits are widely present in the congested areas of the design, making them the hot spots of susceptibility to soft errors. Using zero-optimized SRAM cells in the main routing channel removes such susceptibility in congested areas. On the other hand, the compact routing can reduce the routing wire length, resulting in significant reduction in the number of sensitive one configuration bits.

The redundant routing channel in the proposed architecture is optimized for the logical value of one to avoid open faults. We use this channel to reroute sensitive nets to further enhance dependability. The size of this channel can be set at the design time of the routing fabric to obtain any desired level of dependability. This will be detailed in the next section. Nets passing through the redundant channel are routed such that any SB pattern that can lead to a short fault is avoided. Such limitation can potentially impose significant area overhead to the routing fabric, however, this limitation makes the routed nets in the redundant channel immune to both short and open faults. In addition, the area overhead is imposed only for the most sensitive nets of a design, which is typically a small percentage of the total design nets. The general perspective of the proposed architecture for a sample sensitive net and its redundant net in the design can be observed in Fig. 6. As can be observed in Fig. 6, the original nets are routed through the main routing channel, while the replica nets are routed through the redundant channel.

B. Fault Masking Circuitry

One major challenge in the proposed routing fabric is to design a voter circuitry to distinguish between the faulty and nonfaulty signals. In general, a voting scheme requires at least three inputs to distinguish the nonfaulty signal, however, we propose a fault masking circuitry to distinguish the nonfaulty signal using only the original net and its redundant net. In this section, we present a low-overhead fault masking scheme compatible with the proposed routing architecture. To this end, we first define some notations which will be commonly used in this section. As shown in Table III, 1^s and 0^s are nonfaulty signals, whereas 1^{sf}, 0^{sf}, 1^{wf}, and 0^{wf} represent faulty signals

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE III
NOTATIONS USED FOR ERROR PROPAGATION

Notation	Definition	Notation	Definition
1^s	Strong non-faulty signal with logical value of one	0^s	Strong non-faulty signal with logical value of zero
1^{sf}	Strong faulty signal with logical value of one	0^{sf}	Strong faulty signal with logical value of zero
1^w	Weak non-faulty signal with logical value of one	0^w	Weak non-faulty signal with logical value of zero
1^{wf}	Weak faulty signal with logical value of one	0^{wf}	Weak faulty signal with logical value of zero
Z	High impedance signal		

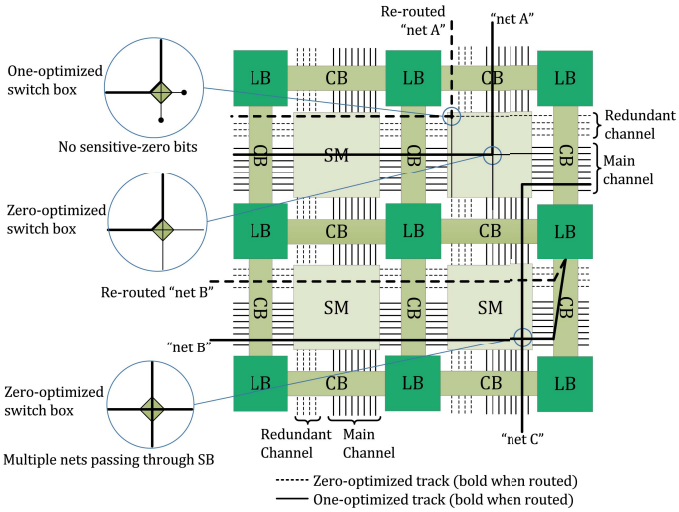


Fig. 6. Sample sensitive nets in the proposed architecture.

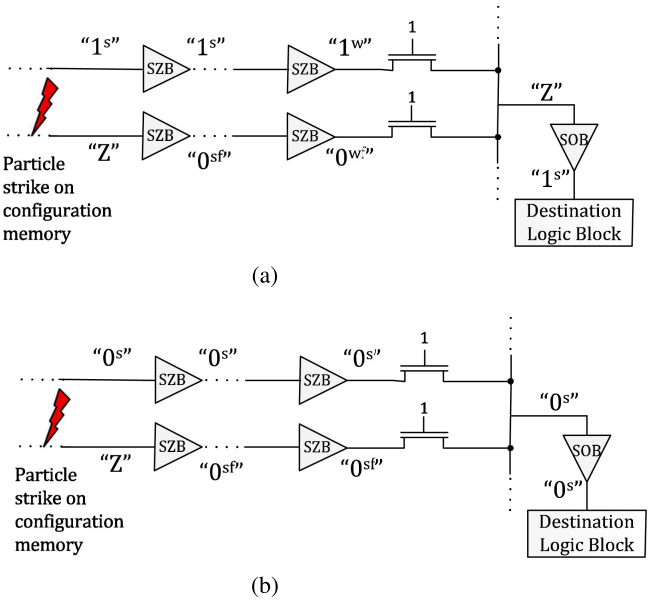


Fig. 7. Proposed fault masking circuitry. (a) Open faulty net and its non-faulty replica net with logical value of one. (b) Open faulty net and its non-faulty replica net with logical value of zero.

as defined in Table III. These notations will be also used to explain the proposed fault masking scheme.

In the proposed fault masking scheme, we use skewed buffers in which the corresponding pull-down networks are stronger than their pull-up networks or vice versa. Skewed buffers with stronger pull-up or pull-down networks are called strong one buffers (SOB) and strong zero buffers (SZB), respectively. The proposed fault masking circuitry using skewed buffers is shown in Fig. 7. Using SZBs in the routing signals ensures that open faults will always result in the logical value of zero after passing through a SZB. Hence, any open fault passing through such skewed buffer will lead to 0^{sf}. This potentially removes possibility of having either 1^{wf} or 1^{sf} values in the routing signals.

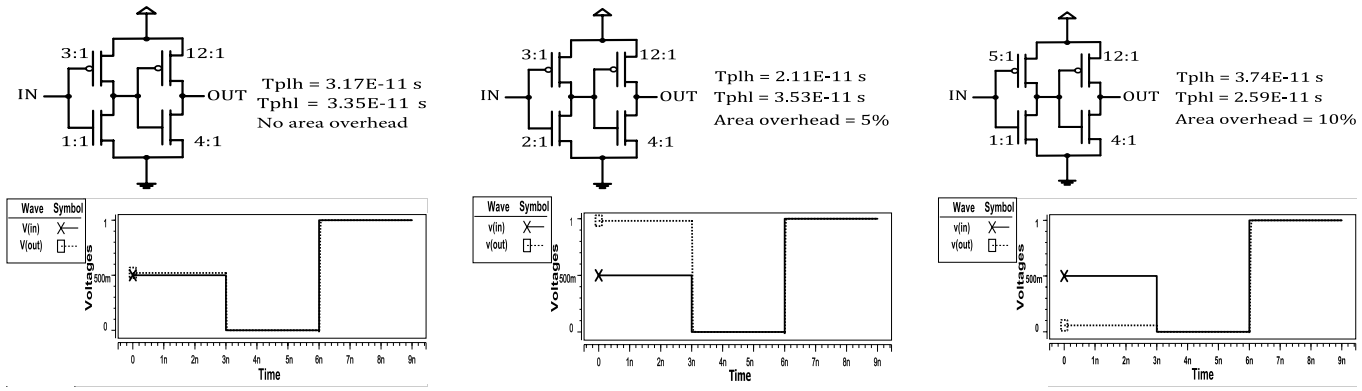
In the proposed fault masking scheme, any original net and its replica net are voted using SOBs. SOBs are connected to the destination LB, as shown in Fig. 7. Since all 1^{wf} and 1^{sf} values are eliminated using SZBs, all logical one values in the routing fabric will be nonfaulty. Hence, faulty signals propagated to the convergence points would be either 0^{wf} or 0^{sf}.

In Fig. 7(a), a signal with logical value of one is faulty. The signal on the bottom, affected with an open fault, passes through the first buffer making it a strong zero but faulty signal (0^{sf}). Our HSPICE simulations demonstrate that when the faulty and nonfaulty signals (i.e., 0^{sf} and 1^s) converge, both will be weakened at the convergence point before the pass transistors, resulting in either 0^{wf} or 1^w. The weakened signals will be further distorted after passing through the pass transistors, resulting in high-impedance signal (Z). The last buffer stage at the convergence point, which is implemented as a SOB, restores the distorted Z signal value to a 1^s and the circuit remains error-free. Another scenario, shown in Fig. 7(b), is where a signal with logical value of zero is faulty. The signal on the bottom, affected with an open fault, passes through the first buffer making it a strong zero but faulty signal (0^{sf}). When the two signals converge (0^{sf} and 0^s), since both have the value of zero, they result in a strong zero signal in the convergence point, which is an error-free signal. This signal is then fed into the LB by the input buffer and the circuit remains error free.

To avoid timing issues, both SOBs and SZBs have to be modified in size to maintain original timing since skewed buffers with the same size as the normal buffers have higher worst case delays. This imposes an area overhead to routing buffers. This overhead, however, is negligible for the following reasons. First, the routing buffers have a negligible contribution to the total area of SRDs. Second, the increase in the size of routing buffers is not significant by itself even as compared with the area of a normal routing buffer.

Fig. 8 shows a normal buffer and the transistor sizing required to change it to a SZB or SOB in 45-nm technology. As shown in Fig. 8(b), the SOB increases the buffer delay by 5% and also imposes 5% area overhead in terms of transistor width. Similarly, as shown in Fig. 8(c), the SZB imposes 10% area overhead in terms of transistor width and also

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.



(a) (b) (c)

Fig. 8. Buffers used in the routing fabric of the proposed scheme. (a) Normal buffer. (b) Strong one buffer. (c) Strong zero buffer.

increases the delay by 11%. Please note that as compared with LBs (LUTs and FFs) and the routing fabric (SBs, connection blocks, and wires), SZBs, and SOBs contribute to a very small part of SRDs. As such, we expect such overhead would be very negligible in the overall SRD device.

C. Proposed Routing Algorithm

In the proposed routing algorithm, as outlined in Algorithm 1, we use the expected number of sensitive cells for different patterns, as presented in Table I, to rank sensitive nets to route replica nets in the redundant channel (line 1). While rerouting, any single sensitive zero cell should be avoided in the redundant channel. This is because the redundant channel is not optimized for the logical value of zero. Hence, the main goal of the redundant router is to find free tracks capable of rerouting the sensitive nets in the redundant channel while avoiding SBs with sensitive zero configuration bits (line 7 through line 9). The proposed routing algorithm is much less time-consuming than the original routing since design nets are already ordered by the original routing algorithm. After obtaining the order in which nets are supposed to be rerouted, the proposed routing algorithm tries to find paths for each net with the same timing as in the main channel using the same global routing pattern (line 5 through line 11). Once a route is found, which does not violate timing and error sensitivity requirements, it will be chosen as a replica net and the algorithm will be continued for the other nets (line 12 through line 16). This is done for all nets to achieve the maximum fault tolerance with the given area budget.

Algorithm 1 Redundant Routing Channel Algorithm

Input: Detailed routing in the main channel
Input: Expected number of sensitive cells for different SB patterns as presented in $Table1$
Output: Detailed routing in the redundant channel

1 $SortedList \leftarrow$ Rank all original nets in the main channel based on open fault sensitivity (according to $Table1$);
2 **while** *$SortedList$ is not empty* **do**
3 $ptr \leftarrow$ Head of $SortedList$;
4 **for** *All entries in $SortedList$* **do**
5 **while** *there is an untested route for $SortedList[ptr]$ in the redundant channel* **do**
6 $Route_i \leftarrow$ Next available route;
7 **if** $Timing(Route_i) > Timing(OriginalRoute)$ *or $Route_i$ has single sensitive cell* **then**
8 $Route_i \leftarrow Null$;
9 continue;
10 **else**
11 break;
12 **if** $Route_i \neq$ *Null* **then**
13 Route replica net on $Route_i$;
14 Increment ptr;
15 **else**
16 Remove $SortedList[ptr]$ from $SortedList$;

VI. Experimental Results

To evaluate the efficiency of the proposed scheme, we have implemented MCNC benchmark circuits using both the proposed scheme and traditional protection techniques such as TMR, 5MR, and partial TMR. In our experiments, the original placement and routing is performed using VPR 6.0 [31], while the redundant routing algorithm is implemented using a program developed in Java language.

In general, there are two approaches for analyzing a device reliability. The first approach is obtaining failure rate by analytical calculation [32] and the second approach is injecting faults into a real or emulated system [33]. The approach that we use in this paper is based on analytical calculations. Baseline device routing fabric is assumed to have an error rate of 10^{-3} per hour for one million configuration cells. Using the mentioned Java program, we count the number of vulnerable, 1-FT, 2-FT,. . ., *n*-FT sensitive cells. Here, by *n-FT*, we refer to the protected part of nets in the system that can tolerate *n* faults before the routing fabric fails. Then, considering the fact that each net has a different error rate based on their level of protection, we obtain the net error rate. For example, consider one million sensitive configuration cells, where 10% of the cells are vulnerable, 80% are 1-FT,

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE IV
PROPOSED SCHEME VERSUS CONVENTIONAL PROTECTION SCHEMES

Method	Failure Rate	Overhead	Protection State	Failure Rate of the Proposed Scheme with the Same Area Overhead	System Protection State With the Same Area Overhead Using the Proposed Scheme
Baseline	1.00E-03	0	Vulnerable	-	-
Partial TMR-40%	6.01E-04	100%	40% 1-FT	1.14E-04	65% 1-FT, 8% 2-FT or higher
Partial TMR-80%	2.02E-04	200%	80% 1-FT	9.74E-06	97% 1-FT, 26% 2-FT, 13% 3-FT or higher
TMR	3.00E-06	250%	100% 1-FT	6.32-013	100% 1-FT, 52% 2-FT, 11% 3-FT, 5% 4-FT or higher
5MR	5.00E-09	480%	100% 2-FT	2.03E-21	100% 2-FT, 72% 3-FT, 22% 4-FT or higher

and 10% are 2-FT, the net error rate is calculated as follows: $0.1 \times (10^{-3}) + 0.8 \times (2 \times 10^{-6}) + 0.1 \times (3 \times 10^{-9}) \approx 10^{-4}$. To maintain a fair comparison, the same approach has been used for obtaining error rate in TMR and the proposed scheme. We have also used the asymmetric SRAM proposed in [11] to obtain the error rates for asymmetric SRAM cells due to its perfect error rate reduction and low overhead, which results in almost error free cell for the optimized logical value and 15%error rate reduction for the other logical value.

Fig. 9 shows the fraction of the configuration bits protected by the proposed scheme for different area budgets. As an example, with 250% area overhead, the proposed routing fabric will be 100% immune to SEUs, 52% immune to double upsets, 11% immune to three upsets, and 5% immune to more than three upsets. The multiple upsets can affect either a single net or different nets. Note if one uses zero-optimized SRAM cells in the redundant channel, the overhead to move a system from a *(n-1)-FT* system to a *n-FT* system can be less than 100% due to compact routing. However, such scheme will be vulnerable to open faults in the redundant channel, resulting in the increased system failure rate.

In Table IV, we compare the proposed scheme with conventional protection schemes (partial TMR, full TMR, and 5MR). In Table IV, partial TMR-40% and partial TMR-80% denote a system whose 40% and 80% portion is protected using TMR scheme, respectively. Since VPR does not support functional simulation to obtain the severity of sensitivity of different cells, we consider all sensitive cells to be equally sensitive. To provide a fair comparison, only the number of sensitive cells and their level of protection have been considered in both partial TMR and the proposed scheme. It is worth mentioning that in the proposed scheme and our implementation of TMR, LBs are not replicated. Rather, we replicate the nets that connect these LBs together. Hence, the considerations used in [9] are not applicable to our implementation of TMR. It can be seen that the proposed scheme provides higher dependability when compared with the methods such as TMR. The failure rates for the proposed scheme is reduced much more significantly when the degree of fault tolerance of the system is increased compared with the conventional schemes.

To demonstrate that our assumption of the baseline device error rate does not result in loss of generality, the failure rate for both the proposed and conventional protection schemes for different baseline device error rates has been shown in Fig. 10. It can be observed in Fig. 10 that as the dependability requirement increases, the proposed scheme provides higher

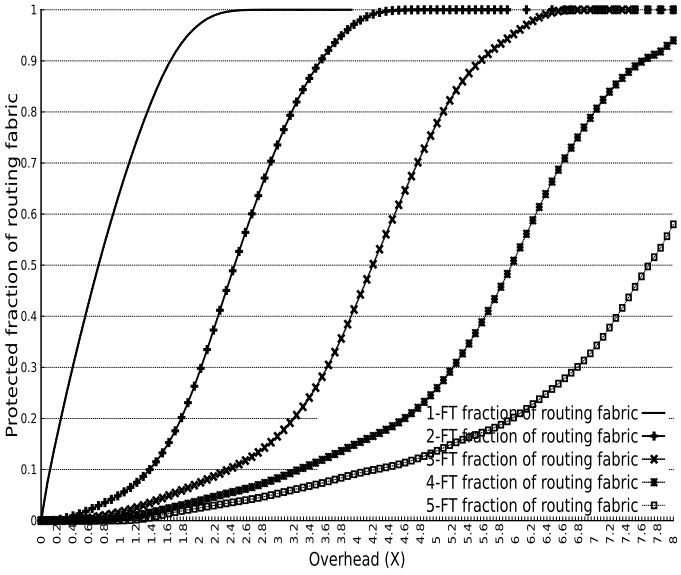


Fig. 9. Protection level provided by the proposed routing scheme for different area overheads (averaged over MCNC benchmark circuits).

dependability levels than its counterparts, making the proposed scheme even more suitable for safety-critical and enterprise systems. As shown in Fig. 10, the proposed scheme reduces the failure rate by roughly one order of magnitude as compared to the partial TMR with the same area overhead. Compared with TMR and 5MR, the proposed scheme reduces failure rate by six and eight orders of magnitude, respectively, while performance remains intact in the proposed scheme. It is worth mentioning that the failure rate of the routing fabric is of the main concern in SRDs as opposed to LUTs. However, the system-level protection schemes such as TMR and 5MR are able to protect against soft errors in the routing fabric as well as LBs.

To demonstrate the efficiency of the proposed routing algorithm and its applicability to larger circuits, we have conducted our experiments on four sample large circuits from the IWLS-05 benchmark suite [34]. Protection level provided by the proposed routing scheme for different area overheads for these four sample circuits along the minimum required grid size can be observed in Fig. 11. Grid size is the number of horizontal and vertical tracks of LBs in the SRD. Although rerouting efficiency highly depends on the behavior of nets in a particular design, it can be seen that the proposed routing algorithm is capable of rerouting nets in more complex circuits.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

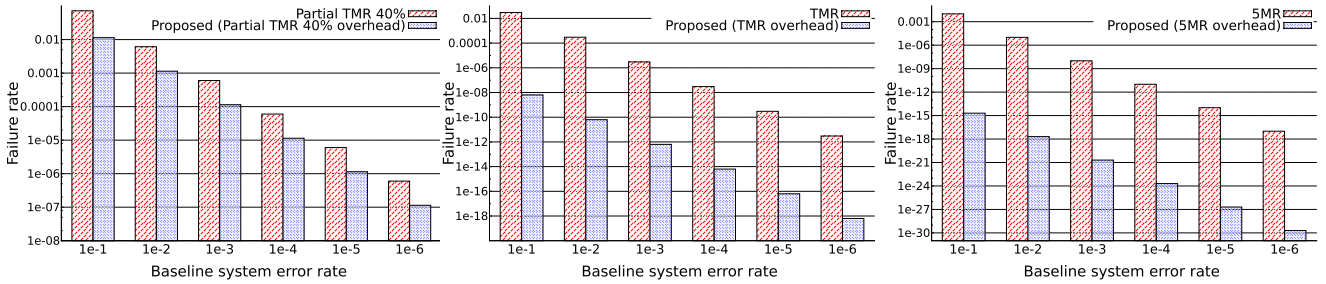


Fig. 10. Failure rate of the proposed routing fabric as compared to the conventional protection schemes with equivalent area overhead. (a) Proposed scheme versus partial TMR (40%). (b) Proposed scheme versus TMR. (c) Proposed scheme versus 5MR.

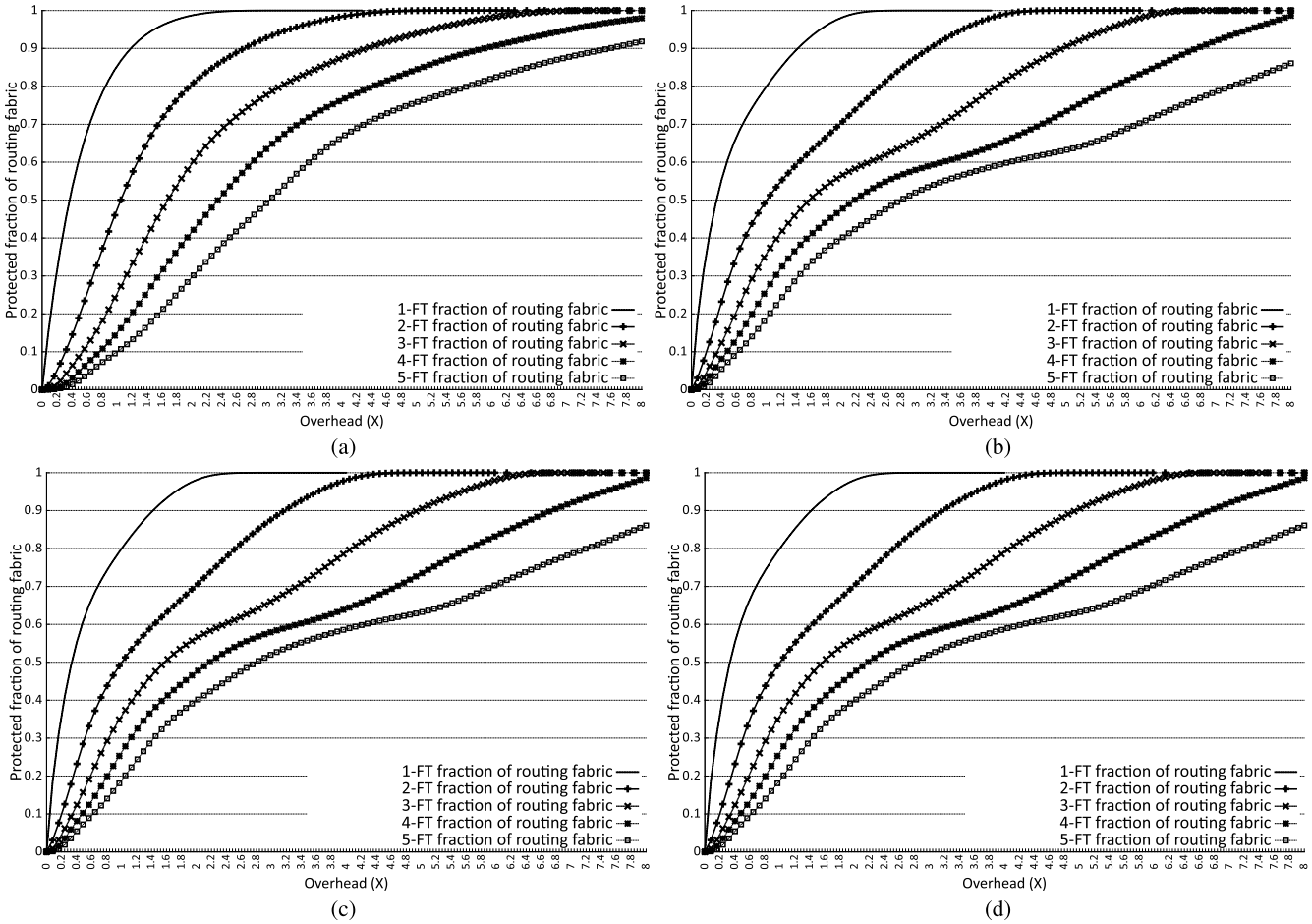


Fig. 11. Protection level provided by the proposed routing scheme for different area overheads for four sample large circuits from IWLS-05 benchmark suite. (a) Aescore (19 × 19 grid). (b) Memctrl (20 × 20 grid). (c) Systemcaes (15 × 15 grid). (d) TV80 (15 × 15 grid).

VII. Conclusion

In this paper, we first presented a study investigating susceptibility of different SB patterns to soft errors. Based on the proposed study, we presented a hybrid routing fabric using asymmetric SRAM cells to reduce routing failure rate. In the proposed routing fabric, the original signals are routed in the main channel, while the replica signals are routed in the redundant channel. Experimental results showed that our proposed scheme can reduce system failure rate several orders of magnitude as compared with TMR and 5-TMR with the same area overhead. The significant failure rate reduction is achieved while performance has remained intact. As a future work, we will update the original place and route algorithms to take advantage of the observations demonstrated for susceptibility of different SB patterns to soft errors. This will help to further reduce system failure rate when area budget is limited.

References

[1] H. Asadi, M. B. Tahoori, B. Mullins, D. Kaeli, and K. Granlund, "Soft error susceptibility analysis of SRAM-based FPGAs in high-performance information systems," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 6, pp. 2714–2726, Dec. 2007.

[2] H. Asadi and M. B. Tahoori, "Analytical techniques for soft error rate modeling and mitigation of FPGA-based designs," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 12, pp. 1320–1331, Dec. 2007.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[3] S. D'Angelo, C. Metra, S. Pastore, A. Pogutz, and G. Sechi, "Fault-tolerant voting mechanism and recovery scheme for TMR FPGA-based systems," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. (DFT)*, Nov. 1998, pp. 233–240.

[4] F. L. Kastensmidt, L. Sterpone, L. Carro, and M. S. Reorda, "On the optimal design of triple modular redundancy logic for SRAM-based FPGAs," in *Proc. Design, Autom. Test Eur. (DATE)*, Mar. 2005, pp. 1290–1295.

[5] C. Carmichael, *Triple Module Redundancy Design Techniques for Virtex FPGAs*. San Jose, CA, USA: Xilinx, 2000.

[6] M. Niknahad, O. Sander, and J. Becker, "Fine grain fault tolerance—A key to high reliability for FPGAs in space," in *Proc. IEEE Aerosp. Conf.*, Mar. 2012, pp. 1–10.

[7] L. Sterpone and M. Violante, "A design flow for protecting FPGA-based systems against single event upsets," in *Proc. 20th IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. (DFT)*, Oct. 2005, pp. 436–444.

[8] M. Violante *et al.*, "Simulation-based analysis of SEU effects in SRAM-based FPGAs," *IEEE Trans. Nucl. Sci.*, vol. 51, no. 6, pp. 3354–3359, Dec. 2004.

[9] L. Sterpone and M. Violante, "A new reliability-oriented place and route algorithm for SRAM-based FPGAs," *IEEE Trans. Comput.*, vol. 55, no. 6, pp. 732–744, Jun. 2006.

[10] S. Srinivasan, A. Gayasen, N. Vijaykrishnan, M. Kandemir, Y. Xie, and M. Irwin, "Improving soft-error tolerance of FPGA configuration bits," in *Proc. IEEE/ACM Int. Conf. Comput. Aided Design (ICCAD)*, Nov. 2004, pp. 107–110.

[11] B. S. Gill, C. Papachristou, and F. G. Wolff, "A new asymmetric SRAM cell to reduce soft errors and leakage power in FPGA," in *Proc. Design, Autom. Test Eur. (DATE)*, Apr. 2007, pp. 1–6.

[12] S. Miao, P. Ou, X. Zhou, and L. Wang, "Zero-hardened SRAM cells to improve soft error tolerance in FPGA," in *Proc. 2nd Int. Symp. Intell. Inf. Technol. Appl.*, vol. 2. Dec. 2008, pp. 278–282.

[13] *Virtex-4 Platform FPGA User Guide*, Xilinx, San Jose, CA, USA, Dec. 2008.

[14] *Stratix-2 Platform FPGA Hand Book*, Altera, San Jose, CA, USA, Apr. 2011.

[15] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *Proc. 7th Int. Field-Program. Logic Appl. (FPL)*, Sep. 1997, pp. 213–222.

[16] H. Ebrahimi, M. S. Zamani, and H. R. Zarandi, "A decoder-based switch box to mitigate soft errors in SRAM-based FPGAs," in *Proc. 15th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2010, pp. 832–837.

[17] S. P. Park, D. Lee, and K. Roy, "Soft-error-resilient FPGAs using built-in 2-D Hamming product code," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 2, pp. 248–256, Feb. 2012.

[18] A. Rohani and H. R. Zarandi, "Mitigating and tolerating SEU effects in switch modules of SRAM-based FPGAs," in *Proc. 5th Southern Conf. Program. Logic (SPL)*, Apr. 2009, pp. 171–176.

[19] Z. Ghaderi, S. G. Miremadi, H. Asadi, and M. Fazeli, "HAFTA: Highly available fault-tolerant architecture to protect SRAM-based reconfigurable devices against multiple bit upsets," *IEEE Trans. Device Mater. Rel.*, vol. 13, no. 1, pp. 203–212, Mar. 2013.

[20] S. Golshan and E. Bozorgzadeh, "Single-event-upset (SEU) awareness in FPGA routing," in *Proc. 44th ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2007, pp. 330–333.

[21] B. Pratt, M. Caffrey, P. Graham, K. Morgan, and M. Wirthlin, "Improving FPGA design robustness with partial TMR," in *Proc. 44th IEEE Int. Rel. Phys. Symp.*, Mar. 2006, pp. 226–232.

[22] B. Pratt, M. Caffrey, J. F. Carroll, P. Graham, K. Morgan, and M. Wirthlin, "Fine-grain SEU mitigation for FPGAs using partial TMR," *IEEE Trans. Nucl. Sci.*, vol. 55, no. 4, pp. 2274–2280, Aug. 2008.

[23] Y. Hu, Z. Feng, L. He, and R. Majumdar, "Robust FPGA resynthesis based on fault-tolerant Boolean matching," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2008, pp. 706–713.

[24] J.-Y. Lee, Z. Feng, and L. He, "In-place decomposition for robustness in FPGA," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2010, pp. 143–148.

[25] Z. Feng, Y. Hu, L. He, and R. Majumdar, "IPR: In-place reconfiguration for FPGA fault tolerance," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2009, pp. 105–108.

[26] C.-C. Peng, C. Dong, and D. Chen, "SETmap: A soft error tolerant mapping algorithm for FPGA designs with low power," in *Proc. 16th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2011, pp. 388–393.

[27] M. Jose, Y. Hu, R. Majumdar, and L. He, "Rewiring for robustness," in *Proc. 47th ACM/IEEE Design Autom. Conf. (DAC)*, Jun. 2010, pp. 469–474.

[28] J.-Y. Lee, Y. Hu, R. Majumdar, L. He, and M. Li, "Fault-tolerant resynthesis with dual-output LUTs," in *Proc. 15th Asia South Pacific Design Autom. Conf. (ASP-DAC)*, Jan. 2010, pp. 325–330.

[29] N. Jing, J.-Y. Lee, W. He, Z. Mao, and L. He, "Mitigating FPGA interconnect soft errors by in-place LUT inversion," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2011, pp. 582–586.

[30] H. Ebrahimi, M. S. Zamani, and S. A. Razavi, "A switch box architecture to mitigate bridging and short faults in SRAM-based FPGAs," in *Proc. IEEE Int. Symp. Defect Fault Tolerance VLSI Syst. (DFT)*, Oct. 2010, pp. 218–224.

[31] J. Rose *et al.*, "The VTR project: Architecture and CAD for FPGAs from verilog to routing," in *Proc. ACM/SIGDA Int. Symp. Field Program. Gate Arrays (FPGA)*, 2012, pp. 77–86.

[32] G. Asadi and M. B. Tahoori, "Soft error rate estimation and mitigation for SRAM-based FPGAs," in *Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays (FPGA)*, 2005, pp. 149–160.

[33] N. Jing, J.-Y. Lee, Z. Feng, W. He, Z. Mao, and L. He, "SEU fault evaluation and characteristics for SRAM-based FPGA architectures and synthesis algorithms," *ACM Trans. Design Autom. Electron. Syst.*, vol. 18, no. 1, 13:1–13:18, Jan. 2013.

[34] K. McElvain, "IWLS'93 benchmark set: Version 4.0," in *Proc. Distrib. MCNC Int. Workshop Logic Synth.*, vol. 93. 1993.

Sadegh Yazdanshenas received the B.Sc. degree from the Iran University of Science and Technology, Tehran, Iran, in 2012, and the M.Sc. degree from the Sharif University of Technology (SUT), Tehran, in 2014.

He is currently a Researcher with the Data Storage Systems and Networks Laboratory, Department of Computer Engineering at SUT. His current research interests include reconfigurable computing, fault-tolerant design, and emerging nonvolatile memory technologies.

Hossein Asadi (M'08) received the B.Sc. and M.Sc. degrees in computer engineering from the Sharif University of Technology (SUT), Tehran, Iran, in 2000 and 2002, respectively, and the Ph.D. degree in electrical and computer engineering from Northeastern University, Boston, MA, USA, in 2007.

He was with EMC Corporation, Hopkinton, MA, USA, as a Research Scientist and Senior Hardware Engineer, from 2006 to 2009. From 2002 to 2003, he was a member of the Dependable Systems Laboratory, SUT, where he researched hardware verification techniques. From 2001 to 2002, he was a member of the Sharif Rescue Robots Group. He has been with the Department of Computer Engineering, SUT, since 2009, where he is currently a tenured Assistant Professor. He is the Founder and Director of the Data Storage Systems Laboratory at SUT. He has authored and co-authored more than 50 technical papers in reputed journals and conference proceedings. His current research interests include data storage systems and networks, solid-state drives, and reconfigurable and dependable computing.

Dr. Asadi was a recipient of the Technical Award for the Best Robot Design from the International RoboCup Rescue Competition, organized by AAAI and RoboCup, and the Distinguished Lecturer Award from SUT in 2010, one of the most prestigious awards in the university.

Behnam Khaleghi received the B.Sc. degree in computer engineering from the Sharif University of Technology (SUT), Tehran, Iran, in 2013.

He is currently a Research Assistant with the Data Storage Systems Laboratory at SUT. His current research interests include reconfigurable architectures and computer-aided design.