Limit of Hardware Solutions for Self-Protecting Fault-Tolerant NoCs

AHMED LOURI, George Washington University
JACQUES COLLET, LAAS/CNRS
AVINASH KARANTH, Ohio University

We study the ultimate limits of hardware solutions for the self-protection strategies against permanent faults in networks on chips (NoCs). NoCs reliability is improved by replacing each base router by an augmented router which includes extra protection circuitry. We compare the protection achieved by the self-test and self-protect (STAP) architectures to that of triple modular redundancy with voting (TMR). Two STAP architectures are considered. In the first one, a defective router self-disconnects from the network, while it self-heals in the second one. In practice, none of the considered architectures (STAP or TMR) can tolerate all the permanent faults, especially faults in the extra-circuitry for protection or voting, and consequently, there will always be some unidentified defective augmented routers which are going to transmit errors in an unpredictable manner. This study consists of tackling this fundamental problem. Specifically, we study and determine the average percentage of residual unidentified defective routers (UDRs) and their impact on the overall reliability of the NoC in light of self-protection strategies. Our study shows that TMR is the most efficient solution to limit the average percentage of UDRs when there are typically less than a 0.1 percent of defective base routers. However, TMR is also the most cost prohibitive and the least power efficient. Above 1% of defective base routers, the STAP approaches are more efficient although the protection efficiency decreases inexorably in the very defective technologies (e.g. when there is 10% or more of defective base routers). For instance, if the chip includes 10% of defective base routers, our study shows that there will remain on the average 1% of UDRs, which causes a major challenge for NoC reliability.

CCS Concepts: • **Hardware** → **Network on chip**; **Fault tolerance**;

Additional Key Words and Phrases: Network-on-chips, reliability, built-in-self-test, self-healing

ACM Reference format:
Ahmed Louri, Jacques Collet, and Avinash Karanth. 2019. Limit of Hardware Solutions for Self-Protecting Fault-Tolerant NoCs. *J. Emerg. Technol. Comput. Syst.* 15, 1, Article 4 (January 2019), 16 pages.
https://doi.org/10.1145/3233986

1 INTRODUCTION

With technology scaling, hardware reliability has become a major obstacle to reaping the benefits of increased integration projected by Moore's law. Integrated circuit reliability is affected

This research was partially supported by NSF grants CCF-1420718, CCF-1513606, CCF-1703013, CCF-1547034, CCF-1547035, CCF-1540736, and CCF-1702980.
Authors' addresses: A. Louri, Department of Electrical and Computer Engineering, The George Washington University, 800 22nd Street NW, Room 5580, Washington DC 20052; J. Collet, Laboratoire d'Analyse et d'Architecture des Systèmes, Université Paul Sabatier, 7 avenue du colonel Roche, 31077 Toulouse Cedex 13; A. Karanth, School of Electrical Engineering and Computer Science, Ohio University, 322D Stocker Center, Athens, OH 45701.

1550-4832/2019/01-ART4 $15.00
https://doi.org/10.1145/3233986

by many parameters, including design-related parameters such as gate oxide width, wire cross-sectional area, integration density, and chip area; process-related parameters such as defect size, distribution, and density; and operation-related parameters such as voltage, power density, and temperature. There is a wide body of literature that deals with the reduction of dimensions and the inevitable increase of fluctuations occurring at atomic scale (e.g., see [1–4], which, in turn, raises the occurrence of permanent faults in future chips with billions of transistors. Making the chip capable of autonomously detecting and tolerating its faults is a problem of extreme difficulty, with multiple and overlapping aspects. One may consider tolerating faults at different abstraction levels: at the transistor level (which corresponds to the finest granularity in the physical layers) [1–3], then at the gate level [5–7], at the memory level with self-test and self-repair mechanisms [8–12], at the router level in networks on chips (NoCs) [13–16], at the processor (or core level). and with assertions at the software level [17].

Although processing cores and on-chip memory take up most of the silicon area and receive considerable attention in reliability research, the NoC occupies a growing amount of silicon area (and volume), spanning several metal layers and router logic. It is predicted that future chips will contain thousands of cores interconnected by a sizeable NoC. The NoC is neither immune to hard faults nor unaffected by the adverse increase in hard faults caused by technology scaling. Therefore, it is of utmost importance to focus on the reliability of future NoCs [18–24].

The reliability of NoCs depends on the occurrence of faults in links and/or in routers. Since the reliability of physical links has been extensively studied and adequate and cost-effective solutions exist [25–27], we do not consider faults in links in this work. We focus on the permanent faults appearing in routers, which we think are more difficult to detect. Permanent faults within the router constrain the flow of information within the router since the router is responsible for buffering the incoming packets, routing the packet to the appropriate output port, allocating input/output buffers and allocating a switch for reaching the output ports. Therefore, permanent faults within the router should be detected, and packets should be rerouted to avoid faulty components and ensure a reliable flow of information across the router.

There is a well-known approach to self-protecting the NoC that consists of replacing the original base router by an *augmented* router (AR), including extra self-protection circuitry. One can consider two variants: (1) *Self-Test and Protecting (STAP)* and (2) *Replicated Architecture with Voting (RAV).*

— In STAP architecture, the augmented router is made up of the base router, a fault detector, multiplexers (Mux) and demultiplexers (DMux; to enable physical reconfiguration) and a protection block, as will be shown in the next section. Each augmented router starts operation in a test session fully dedicated to the identifcation of permanent faults in the base router in conjunction with physical reconfiguration when faults are detected.
— In RAV design, the augmented router is made up of several replications of the base router that execute in parallel the same processing (so-called space redundancy) and a majority voter to define the correct result on the fly. The most common example is triple modular redundancy (TMR) [28, 29]. Note that there is neither an initial session dedicated to the identification of faults nor a physical reconfiguration in these architectures.

There are fundamental problems in each architecture, as follows.

— In the STAP approach, not all faults can be detected in the base router because there is no mechanism that can provide 100% fault coverage. Moreover, the extra protection circuitry itself can introduce faults that cannot be detected. Because of these issues, there will always be defective routers that are unidentifiable (we call them unidentified defective routers, UDRs). The percentage of UDRs increases in the strongly defective technologies, which we will define shortly.

—In the TMR approach, the cost (chip area and circuitry) is too prohibitive (triplication of the base router and additional voting circuitry). Moreover, the power consumption is too high because the redundant circuitry is always on as opposed to STAP architectures where there is no extra power consumption owing to protection (the base router or the protection is activated but not both at the same time). Finally, there is an intrinsic limitation because all RAVs degrade the reliability when the fault probability of the base router exceeds $p = 0.5$, regardless of the number of redundant routers.

In this work, we study the ultimate limits of hardware solutions for the self-protection strategies against permanent faults in NoCs. We tackle this often-neglected fundamental problem: that is, the increase of the percentage of UDRs starting from the weakly defective technologies (when the percentage of defective base routers is lower than 0.1%) to the strongly defective technologies when it is larger than a few percent. We compare the residual percentage of UDRs, when the protection is achieved by self-test and self-protect (STAP) architectures to that of triple modular redundancy with voting (TMR). As previously stressed, none of the considered architectures (STAP or TMR) can detect all permanent faults. Consequently, some unidentified defective augmented routers may remain that are going to transmit errors in an unpredictable manner.

The article is organized as follows. In Section 2, we introduce two NoC self-protecting architectures, one based on router self-disconnection and another on router self-healing. We also describe in detail the dependability issues generated by these architectures. In Section 3, we develop a probabilistic model to determine the appropriate state of the augmented router that may be fault free, protected. or unidentified and defective (UDR). We derive the average percentage of UDRs in the NoC. In Section 4, we derive the transistor counts for the self-healing routers. In Section 5, we study the persistence of UDRs for the STAP and TMR architectures. We provide our conclusions in Section 6.

2 PROPOSED STAP ARCHITECTURES

We consider two STAP architectures: the first is based on self-disconnection and the second on self-healing. In the self-disconnect architecture, the router disconnects from the NoC; in the self-healing architecture, the router repairs itself in whole or in part by activating a protection block.

2.1 Self-Disconnecting Architecture (SDA)

Figure 1 shows that the base router is surrounded by a fault detector (FD) to detect the faults and several multiplexers (Muxes) and demultiplexers (DMuxes) to enable physical reconfiguration of routes. For simplicity, we describe a 2×2 router; in reality, a 5×5 router is needed in a 2-dimensional (2D) mesh network. We call the full circuit shown in Figure 1 an AR. The thick lines show the paths activated at startup: All Muxes and DMuxes are configured so that the fault detection can test the base router while the inputs In1 and In2 are blocked during the test. In other words, all routers are by default disconnected at startup, when the test session is initiated. Self-detection and reconfiguration are conducted autonomously as follows.

(1) The test vector generator (see the TVG block) injects test vectors into the base router and compacts the test stream into a test signature S2. Compacting is based on the cyclic redundancy check (CRC) of the test stream [30].
(2) The test signature S2 is compared to the reference S1 (i.e., the signature of the fault-free base router). The comparator C generates a single decision bit, which controls the reconfiguration of all Muxes and DMuxes.

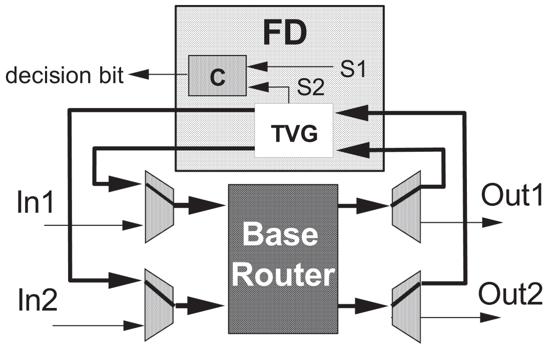

Fig. 1. Self-Disconnecting Architecture (SDA). Thick lines show the activated routes at startup. The fault detector (FD) is looped to the "Base Router" to execute the fault detection.

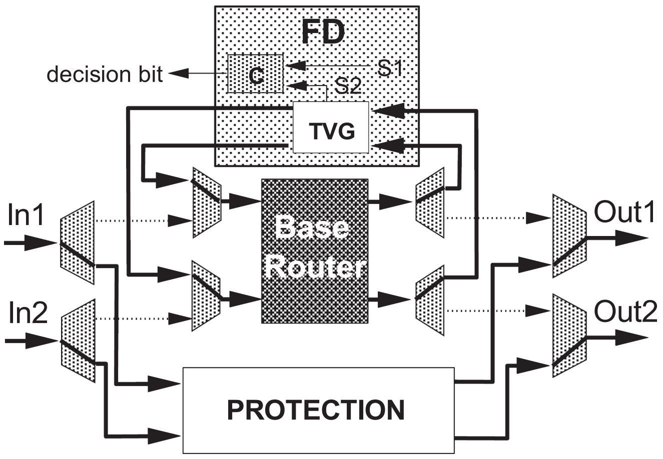

Fig. 2. Self-healing Architecture (SHA) Thick lines show the default routes activated across the architecture at startup, before the test.

(3) When S1 is equal to S2 (e.g., no fault detected), the Muxes and DMuxes are reconfigured so that the inputs In1 and In2 are forwarded to the base router and the base router is connected to the network (e.g., the DMuxes are connected to the output lines Out1 and Out2). If S1 is not equal to S2, the based router stays disconnected from the network.

2.2 Self-Healing Architecture (SHA)

Figure 2 shows the self-healing architecture (SHA).

The fundamental difference between the SHA and the SDA described in the prior section is the addition of the PROTECTION block, which consists of circuitry that can mimic in whole or in part the functionality of the base router with minimal hardware. The PROTECTION block consists of most of the router functionalities that are necessary when faults are observed within the main router. Several schemes have been proposed for achieving self-healing routers (e.g., see [13–15]). Probably, the simplest PROTECTION block is a bypass module that forwards West-incoming messages to East, similarly North-incoming messages to South, and conversely. The most complex is likely a duplication of the base router. As in the SDA, the base router is connected at startup to the fault detection block to enable self-test without reconfiguration.

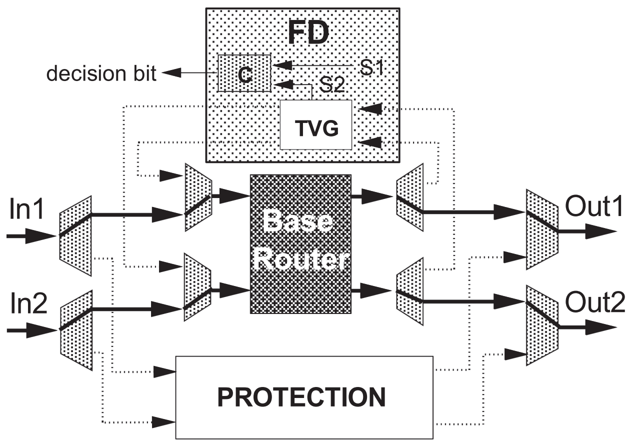

Fig. 3. Router configuration in the SHA architecture when the baseline router is not faulty at the end of the STAP procedure. Thick lines show the activated routes in the router. Inputs In1 and In2 are forwarded to the base router, and the fault detector disconnected.

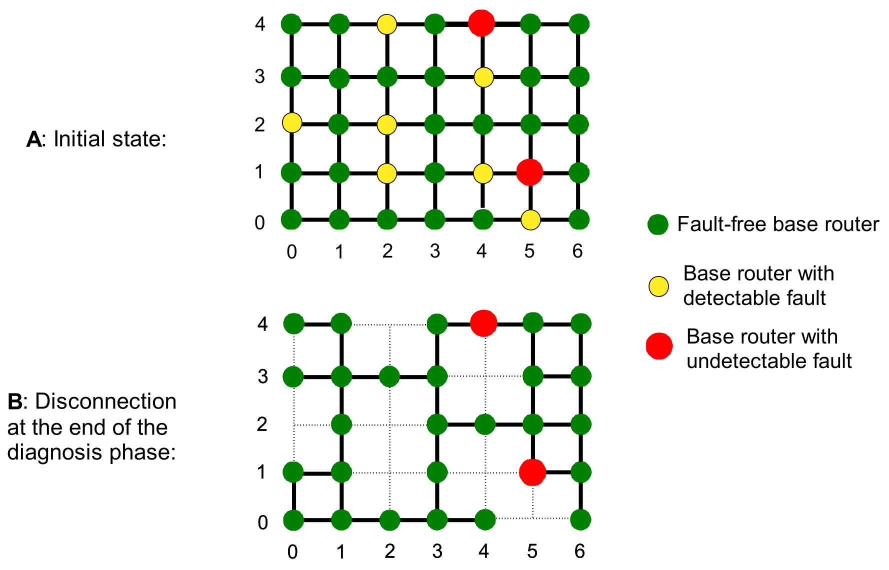
A: Initial state:
B: Disconnection at the end of the diagnosis phase:
0
1
2
3
4
5
6
Fault-free base router
Base router with detectable fault
Base router with undetectable fault

Fig. 4. Illustration of the fundamental problem of UDRs in the SDA. Figure 4(a) shows the initial state of the architecture, including detectable and undetectable routers. Figure 4(b) shows the state at the end of the startup detection and disconnection procedure.

If faults are detected (when S1 ≠ S2), no reconfiguration takes place. Thus, in the SHA, the PROTECTION block stays connected to the NoC. If no permanent fault is detected, reconfiguration takes place and the base router is connected to the NoC. Figure 3 shows the reconfiguration in the SHA.

2.3 Reliability Limits of the Proposed STAP Architectures

The limit of the proposed STAP network stems from the occurrence of UDRs. To illustrate this problem, we provide two examples covering both architectures. Figure 4 details this fundamental problem in the SDA.

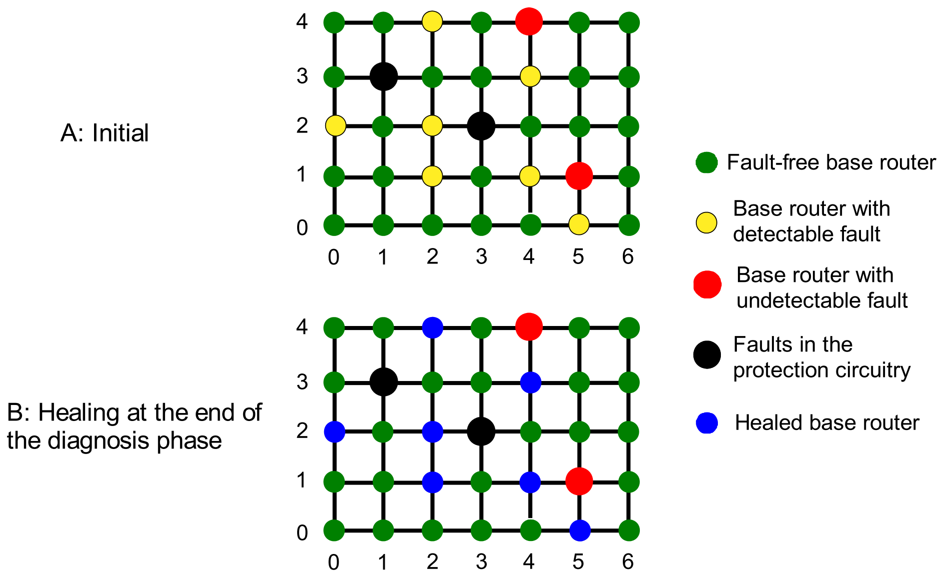
A: Initial
B: Healing at the end of the diagnosis phase
0
1
2
3
4
5
6
Fault-free base router
Base router with detectable fault
Base router with undetectable fault
Faults in the protection circuitry
Healed base router

Fig. 5. Illustration of the fundamental problem of UDRs in SHA. Figure 5(a) shows the initial state of architecture (it includes detectable and undetectable routers). Figure 5(b) shows the state after the startup STAP procedure.

We consider a 2D-mesh network with $5 \times 7 = 35$ nodes. Fault-free routers are represented by dark-green dots. We assume that there are 9 defective routers. Seven of them, owing to detectable faults in the base router, are indicated by yellow dots in positions (0,2), (2,1), (2,2), (2,4), (4,1), (4,3), and (5,0). In addition, there are two UDRs in positions (4,4) and (5,1) indicated by red dots as shown in Figures 4(a) and 4(b), the latter of which displays the NoC state at the end of the STAP procedure. Each augmented router diagnosed as faulty by its fault detector self-disconnects from the network and becomes a hole (see the holes in the network) except the two UDRs. The challenge consists of finding at least one connecting route between any pair of functional routers that circumvents the holes (i.e., the disconnected detectable defective routers) and includes no red routers (undetectable defective routers).

Figure 5 details the fundamental problem in the SHA. Figure 5(a) shows the initial state. It is similar to the initial state of the SDA except that there are two more UDRs (indicated by black circles) due to the possible occurrence of faults in the PROTECTION block. Note that, by default, there is no specific fault tolerance implemented in the PROTECTION BLOCK (e.g., the protection block is not further protected). Figure 5(b) shows the state of the architecture at the end of the STAP procedure. Routers diagnosed as faulty by the fault detector are healed (these are indicated by the blue circles in the network); the red UDRs and the newly introduced black UDRs show up in the final state of the NoC.

Figures 4 and 5 highlight two fundamental questions that become critical in massively defective technologies:

(1) What is the preferable solution to preserve the dependability of communications: A NoC with many holes but few UDRs (Figure 4) or a NoC without holes but more UDRs (Figure 4)?
(2) How to find at least one connecting route (between any pair of functional routers), which includes neither a red nor a black router (i.e., no UDRs). However, for any given chip, this

challenge cannot be fulfilled since it is totally impossible to pinpoint the exact location of the UDRs, hence, the necessity for this study.

In the next section, we calculate the average residual percentage of UDRs in the NoC for the proposed STAP architectures and recall the known results for TMR. We underline that, in this article, we quantify and characterize only the limits of dependability, and we do not provide solutions to discover the UDRs in a given NoC. This will be done in a separate study.

3 AVERAGE PERCENTAGE OF UNIDENTIFIED DEFECTIVE ROUTERS

It should be emphasized that the mean percentage of UDRs is also the probability P_{UDR} that the router is UDR. Thus, we first calculate the probabilities for the different operation states for STAP and TMR routers.

3.1 Reliability Limits of the Proposed STAP Architectures

Here, we consider the self-disconnecting and self-healing architectures described in the previous section. We distinguish three possible operation modes of the augmented router at the end of the STAP procedure:

Mode 0: Operation with Base Router Activated
This mode occurs in two cases.

Case 1: The base router, the Mux/DMux and the FD, are all fault-free. Thus, the FD calculates the correct test signature S2. As S2 = S1, all Muxes/DMuxes are configured to forward the inputs In1 and In2 to the base router, which, in turn, connects output lines Out1 and Out2 (see Figures 2 and 3). The probability in this case is

$$P_{01} = R_R R_D R_{Mux}, \tag{1}$$

where R_R, R_D, and R_{Mux} are, respectively, the reliabilities of the base router, the FD, and Muxes/DMuxes. The reliability of a block is the probability that it contains no permanent fault. For instance, the probability R_R that the base router is reliable is

$$R_R = (1 - f_{Tr})^{N_R}, \tag{2}$$

where f_{Tr} is the probability that a transistor is defective and N_R the base-router transistor count. The probability F_R that the base router is defective and includes one or several faults is $F_R \equiv 1 - R$. Similar equations hold for any block. For instance, the reliability of the fault detector (if no specific protection mechanisms are implemented) is

$$R_D = (1 - f_{Tr})^{N_D}, \tag{3}$$

where N_{D} is the transistor count of the fault detector.

Case 2: We mention this case although its contribution is small compared to that of case 1. The base router and the Muxes/DMuxes are still fault free, but the FD is now faulty. Nevertheless, from time to time, it computes the correct signature. Obviously, this case is very unlikely but it is still possible. Because S2 = S1, node inputs are forwarded to the base router. The corresponding probability is $P_{02} = R_R(F_D)R_{Mux}$, where $F_D = 1 - R_D$ is the probability that the fault detector is faulty. εF_D is the probability that FD is faulty but calculates the correct signature, where ε is a small number compared to 1.

The probability $\boldsymbol{R}(0)$ that the augmented router works deterministically in state 0, that is, with the base router activated, is

$$R\,(0) = R_R\,(R_D + F_D)\,R_{Mux}. \tag{4}$$

Mode 1: Operation with Protection Block Activated

This mode occurs in two cases:

Case 1: The base router includes faults, *and* at least one fault is detected by the fault detector, which switches the Mux and DMux to activate the protection circuit. The operations are reliable if the Mux/DMux and the protection block are also fault free. The corresponding probability is

$$P_{1,0} = F_{R,D} R_D R_{Mux} R_P, \tag{5}$$

where R_D, R_{Mux}, and R_P are the reliability probabilities calculated with equations similar to Equation (2). Thus,

$$R_P = (1 - f_{Tr})^{N_P} \tag{6}$$

$$R_{Mux} = (1 - f_{Tr})^{N_{Mux}} \tag{7}$$

N_P and N_{Mux} are the corresponding transistor counts.

— $F_{R,D}$ is the probability that the fault detector detects at least one fault in the base router. To calculate this term, we first introduce the fault coverage k of the detector. It is defined as the "average percentage of faults that the test can detect when applied to a module" [31]. It is an average value because faults are distributed differently in each module, making the fault detection efficiency module dependent. For a module with N transistors, the average number of defective transistors diagnosed as such is thus $kf_{Tr}N$. The complement $(1\text{-}kf_{Tr})N$ is the average number of transistors not diagnosed as faulty. As far as the average value of detected faulty transistors is concerned, $1 - kf_{Tr}$ may be viewed as the probability for one transistor not being diagnosed as defective. Note that: $1 - kf_{Tr} \equiv [1 - f_{Tr}] + [1 - k]f_{Tr}t$. This equation makes sense, as it means that a transistor is not diagnosed as faulty if it is not or if the fault is beyond the fault coverage k.

The probability that no fault is detected in the base router is $(1 - kf_{Tr})^{N_R}$. The complement is just $F_{R,D}$, that is, the probability that at least one fault is detected and, consequently, that the activation of the protection block reliably maintains the operation of the augmented router.

$$F_{R,D} = 1 - (1 - kf_{Tr})^{N_R} \tag{8}$$

Case 2: False alarm. It occurs when the protection block is uselessly activated while the base router is not defective. We mention it although it is of second order. It is possible to distinguish two subcases:

(a) The FD contains faults; thus, it may calculate an incorrect signature of the base router. If it does, the decision circuit activates the protection block erroneously. The corresponding probability is

$$P_{1,1} = [1 - \varepsilon] F_D R_{Mux} R_P.$$

where $[1\text{-}\varepsilon]F_D$ is the probability that the FD calculates some incorrect signature. ε is a small number compared to 1.

(b) Muxes/Dmuxes are faulty and activate the protection block without actual fault in the base router. The corresponding probability is $P_{1,2} = R_D \varepsilon F_{Mux} R_P$; ε is a small number compared to 1. This probability is surely extremely small, as several faults must occur simultaneously in Mux *and* DMux to reconfigure the routes across the augmented router.

The probability $\boldsymbol{R}(1)$ that the augmented router works deterministically with the protection block activated is the sum of $P_{1,0}$, $P_{1,1}$, and $P_{1,2}$. It reads as follows:

$$R(1) = (F_{R,D}R_D + [1-\varepsilon] F_D) R_{Mux}R_P + R_D \varepsilon F_{Mux} R_P. \tag{9}$$

Mode 2: Unreliable Operation

The augmented router operates in this mode when it contains faults, possibly undetected in the base router, and/or in the FD, the Mux/DMux, or the protection blocks that are not protected. The probability P_{UDR} that the augmented router is in the unreliable state 2 is

$$P_{UDR} = 1 - R(0) - R(1).$$

Equations become simpler if we consider only the dominant contributions, that is, if we use P_{01} (see Equation (1)) for $R(0)$ and P_{10} (see Equation (5)) for $R(1)$. Then,

$$P_{UDR} = 1 - [R_R + F_{R,D}R_P] R_D R_{Mux}. \tag{10}$$

The different terms of the right-hand side of Equation (10) are defined by Equations (2), (3), and (6) to (8), which depend only on f_{Tr}, k, and on the transistor count for the different blocks under consideration. When f_{Tr} is very small and the reliability of the detector and the Mux/DMux are very close to 1, P_{UDR} reduces to the next regular limit, which we shall reuse in Section 5.1:

$$\lim_{f_{Tr\to 0}} P_{UDR} \approx 1 - [R_R + F_{R,D}R_P] \approx (1-k) f_{Tr} N_R. \tag{11}$$

We must underline that the equations integrate no special protection for the fault detector (i.e., the CRC, which generates the test signature, and the comparison block C), because it is easy to make it very robust due to their relatively small size. We distinguish the following:

(1) Faults due to the intrinsic limitations of the CRC algorithm. This is because two different test streams may generate the same CRC and fail in detecting transmission error during the test. The probability of this error is $p = 2^{-N_{Poly}}$, where N_{Poly} is the degree of the CRC generation polynomial used to calculate the signature. The solution to avoid this problem consists of using a 32b CRC, which makes the error probability p negligible [30].

(2) Physical faults in the CRC circuitry. The occurrence of faults may be dramatically reduced using redundancy strategies and majority vote. However, the most efficient solution is likely to quadruple each transistor [5–7], replacing each transistor (say, T) by the logic function (T+T) (T+T) or TT+TT. This way, any single transistor defect (stuck-open, stuck-short, *and/or* bridge) is tolerated, as well as double stuck-open (or their corresponding bridge) defects (as long as they do not occur in any two parallel transistors) and double stuck-short (as long as they do not occur in any two series transistors).

(3) Physical faults in the comparator. They may also mask the difference between S1 and S2. TMR may make this circuit reliable. Transistor quadrupling is also a possible solution.

3.2 Reliability Limits of Replicated Architectures with Voting (RAV)

We restrict RAV to TMR architecture. Consistent with Figures 1 through 3, Figure 6 shows TMR applied to a 2-input/2-output base router. Figure 6 shows the case of 2b transmission parallelism. Each MV block is a 1b-majority voter. We distinguish two operation modes.

Mode 0: Reliable Operation

Operations are reliable (i.e., no error is ever transmitted) when at most one basic router contains faults and when all 1b majority voters are reliable. The corresponding probability is R_{TMR}:

$$R_{TMR} = \left(R_R^3 + 3R_R^2 F_R\right) R_{MV}^{5N_P}. \tag{12}$$

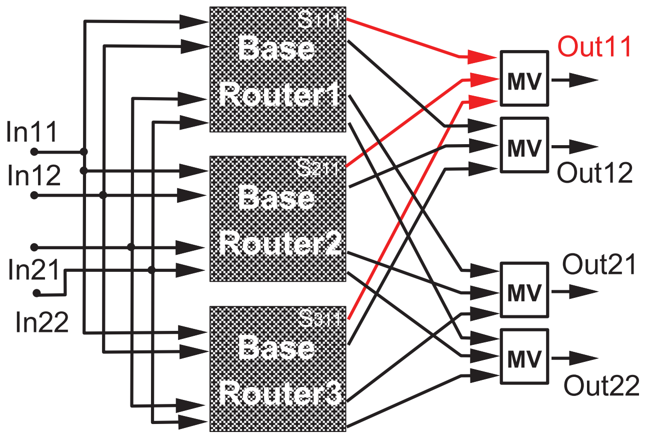

Fig. 6. Augmented router architecture in the TMR approach. MV is a 1b majority voter.

The exponent $5N_P$ comes from the application of TMR to routers in 2D mesh with 5-input/output routers and transmission parallelism N_P. The reliability R_{MV} of a 1b voter is simply

$$R_{MV} = (1 - f_{Tr})^{N_{MV}}. \tag{13}$$

Mode 1: Unreliable Operation
Router operations are unreliable when they are not in state 0. The corresponding probability is

$$P_{UDR} = 1 - \left(R_R^3 + 3R_R^2 F_R\right) R_{MV}^{5N_P}. \tag{14}$$

When $F_R \ll 1$, Equation (14) reduces to:

$$P_{UDR} = F_R^3 + 3F_R^2(1 - F_R) + \frac{5N_P N_{MV}}{N_R} F_R. \tag{15}$$

This equation is especially interesting as it shows that, in the weakly defective technologies (when $F_R \to 0$), the faults occurring in the unprotected voters become the dominant source of unreliability.

4 TRANSISTOR COUNTS IN AUGMENTED ROUTERS

The transistor counts are needed to calculate the reliability terms R_R, R_D, R_P, R_{Mux}, and $F_{R,D}$ (see Equations (2), (3), and (6)–(8)) and the probability P_{UDR} that a router is undetected and defective (Equations (10) and (14)). We derive below these counts for the different constituting blocks.

- Count for Mux/Demux: We count 7 transistors per 2-input multiplexer and 5 transistors per 2-output demultiplexer. Thus, for 32b wide paths in the augmented router and the 5-channel routers under consideration, the transistor count is $N_{Mux} = 32 \times 5 \times 12 \approx 2000$ in the SDA (Figure 1) and twice more in the SHA (Figure 2).
- Count for the baseline router: The base design considered in this work is a five-port NoC router architecture, as shown in [13–15]. The five ports correspond to the four directions in the 2D mesh plus one connection to the local processing element. The transistor count is $N_R \sim$ 200 kTr (kilo transistors) [32].
- Count for the CRC-based fault detector: The speed of the CRC circuit is not critical because it is active only during the initial test session. Nevertheless, to make the duration of the test as short as possible, it is advantageous to adopt a parallel implementation of the CRC

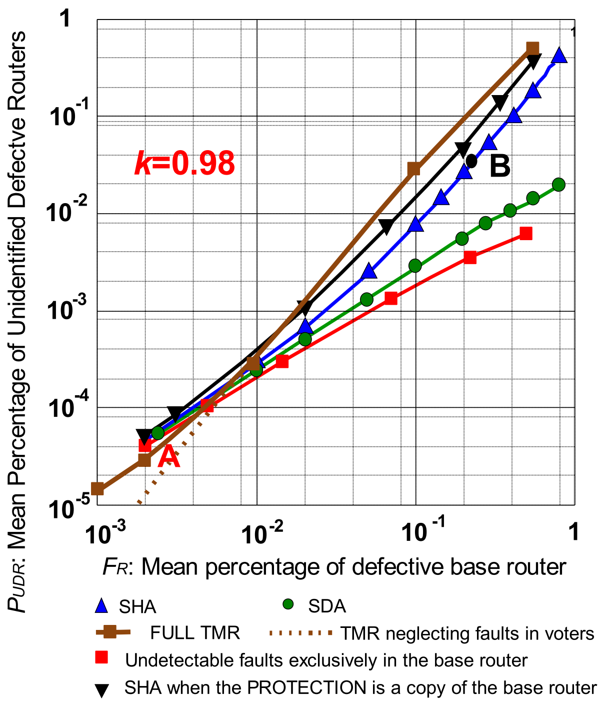

Fig. 7. Mean percentage of UDRs versus the percentage of defective base routers. The fault coverage of the detector is k = 0.98. Both axes are in log scale.

following [33–35] with, as negative consequence, an increase of the size of the CRC circuits. Nevertheless, regarding the transistor count N_D, it remains less than 30 kTr even for a 32b CRC.

— Count for the protection block: We count $N_P = 0$ in the SDA because there is no PROTECTION block. Several SHA for routers have already been published. The transistor count N_P of the PROTECTION block ranges from 85 to 657 kTr, depending on the implementation choices. We shall use N_P as a free adjustable parameter ranging from zero transistors (SDA case) to 200 kTr to show the critical impact of the block size on the percentage of UDRs.
— Count for 1b voter in TMR: A 1b synchronous-voter circuitry requires 12 or 14 transistors [36].

5 LIMITS OF SELF-PROTECTING ARCHITECTURES

In this section, we evaluate the limits of the self-protecting architectures with different detector coverage rates of k = 0.9 and k = 0.98. We chose these two different values of k to evaluate the impact when the NoC has typically less than 1% of defective base routers and 10% or more.

5.1 Limit of STAP Architectures for k = 0.98

Figure 7 displays the mean percentage of residual UDRs considering the SDA, SHA (see Equation (10)) or TMR (see Equation (14)) approaches to the percentage of defective base routers in the unprotected network (Equation (2)).

— The red curve shows the percentage of UDRs due to the sole occurrence of undetectable faults in the base router. When F_R is small—say, lower than 10^{-2}—the percentage of UDRs reduces to Equation (11), that is, $P_{UDR} = (1 - k)F_R$. Point A in Figure 7 follows exactly this law because its abscissa is F_R=2 × 10^{-3} and its ordinate just P_{UDR} = 4 × 10^{-5} = $(1\text{-}k)F_R$. Figure 7 clearly shows that all curves are very close in the weakly defective case, that is, when the NoC has typically less than 1% of defective base routers (that is, F_R < 10^{-2}). This means that UDRs are mostly due to the occurrence of uncovered faults in the base router. Faults occurring in the fault detector, in Mux/DMux, and in the PROTECTION block are negligible. Things change when the NoC includes typically more than 10% of defective base routers.

— The green curve shows the mean percentage of UDRs in the SDA. The green and red curves are obviously in close proximity. This means that, in the SDA, the faults occurring in the protection circuitry finally do not generate much more UDRs than those undetectable in the base router. This is not really surprising, because there is no protection block and the sole faults making the augmented router UDR (in addition to the undetected faults in the base router) are those occurring in the Muxes/DMuxes.
— The blue curve shows the mean percentage of UDRs in the SHA (see the red and black dots in Figure 5). The blue curve is clearly above the green and red curves in the strongly defective limit because there is a significant percentage of UDRs due to the occurrence of faults in the protection block. To highlight this effect, we added the black curve, which shows the percentage of UDRs when the PROTECTION block is large and duplicates the base router, with the transistor count of 200 kTr. It is obvious that any increase in size of the protection block incurs a significant increase of the percentage of UDRs in the NoC.
— Lastly, the solid brown line shows the mean percentage of UDRs in the TMR approach, calculated using Equation (14). Figure 7 shows that TMR is the most efficient approach to reduce the average percentage of UDRs when there are typically less than 0.4% of defective base routers. However, two important points need attention in the weakly defective case: First, neglecting faults in voters leads to an unrealistic overestimation of the efficiency of TMR to reduce the percentage of UDRs. This is demonstrated by comparing the dashed and solid brown lines in Figures 7 and 8. The dashed line (when faults in voters are neglected) is well below the full TMR solid line. It is too optimistic. UDRs are mostly due to faults in voters. Second, TMR is not dramatically more efficient than STAP because P_{UDR} linearly scales as $5N_PN_{MV}F_R/N_R$ (see Equation (15)) for TMR while it scales as $(1 - k)F_R$ in the STAP architectures (see Equation (11)). Above 2% of defective base routers, STAP architectures are clearly more efficient than TMR.

5.2 Limit of STAP Architectures for k = 0.9

Figure 8 shows P_{UDR} versus F_R when the fault coverage of the detector is only $k = 0.9$. Trends are similar to those described in the previous figure. The essential difference is the noticeable increase of the percentage of UDRs. This is logical because less faults can be detected in the base router when k is 0.9 instead of 0.98.

Figures 7 and 8 show that the SDA generates much less UDRs than the SHA, which seems at first sight a decisive advantage for this architecture. Unfortunately, the SDA creates holes in the NoC, with the negative consequence that the increase of the hole percentage reduces the number of existing routes between any two nodes. Consequently, the SDA faces a communication barrier in the defective technologies typically above 30% of disconnected routers.

Figure 9 highlights this problem. Blue squares represent the average percentage of lost routers versus the number of disconnected routers. A router is said to be lost either because it has been diagnosed as faulty and disconnected (the percentage is P_{10}; see Equation (5)) or because it is isolated from the rest of the NoC by the surrounding holes. Points A, B, and C in Figure 9 show

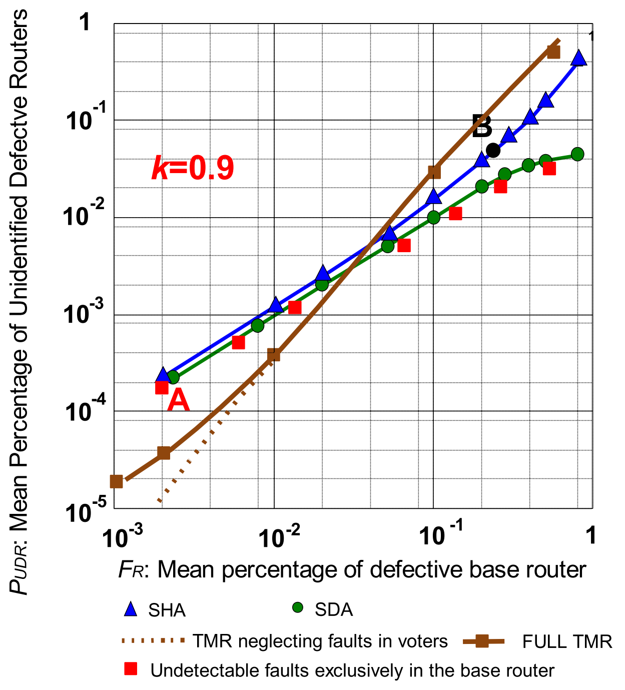

Fig. 8. Mean percentage of UDRs versus the percentage of defective base routers. Same calculation as reported in Figure 7, except that the fault coverage is $k = 0.9$. Both axes are in log scale.

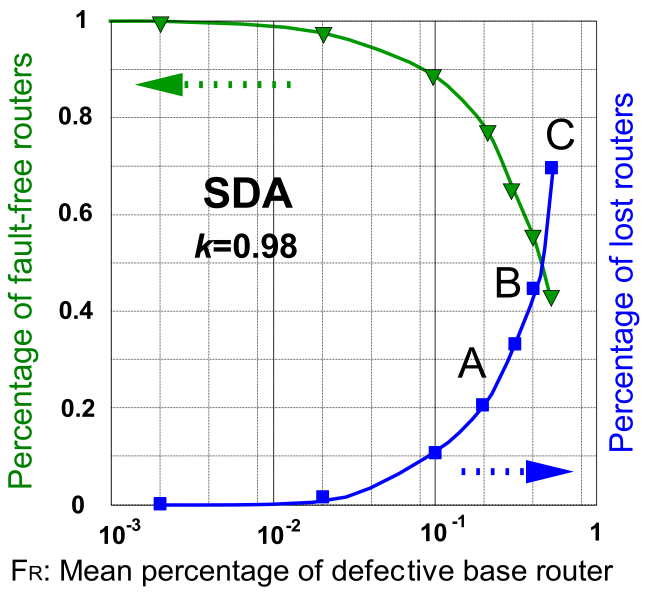

Fig. 9. Mean percentage of lost (blue squares) or fault-free routers (green triangles). Simulation of node reachability where conducted in a 2D mesh with 450 nodes.

the issue. Below 20% of defective (and disconnected) base routers (point A), the average number of lost routers is equal to the number of disconnected routers.

Now, point B shows that at 40% of disconnected routers, about 45% of the routers are lost. This occurs because about 5% of reliable routers are isolated by those that are disconnected. Things become dramatic in the more defective technologies. Point C shows that if about 50% of the routers are diagnosed as faulty and disconnected, then 70% of the routers are lost!

Now, Figure 9 leads to a communication barrier. It is indeed obvious, without any calculation, that the probability that at least one route exits between any two nodes must inevitably collapse with the increase of lost routers and for increasingly distant nodes. This problem is similar to the percolation limit in solids. Our studies [37, 38] showed that this limitation occurs above 30% of disconnected routers in a 2D mesh. These studies of the communication barrier were conducted using the flooding protocol, which surely consumes communication resources in high quantities but is probably the most efficient protocol to discover the existence of routes. Discovering routes with adaptive routing protocols (which are much less resource demanding; e.g., see [39, 40]) is of interest but cannot push away the communication barrier.

6 CONCLUSION

In this article, we studied the limits of hardware solutions for self-protecting NoCs starting from the weakly defective technologies (in which the percentage of defective base routers is lower than 0.1%) to the strongly defective technologies (in which it reaches upwards of 30%). We considered both STAP and RAV architectures. The goal was to study the persistence of UDRs, which are a major threat to the reliability of NoCs and the overall dependability of chips in future massively defective technologies. We distinguish two cases:

Weakly defective case (typically less than 1% of defective base routers):
TMR architecture is the most efficient solution (see Figures 7 and 8). Moreover, TMR has several important advantages: (1) no test session is needed at startup and (2) it tolerates transient faults at runtime. TMR remains an attractive choice when power dissipation and transistor overhead are not critical limiting constraints. If power dissipation is limited, STAP architectures must be considered.

Strongly defective case (typically between 5% and 30% of defective base routers):
STAP architectures are more efficient than TMR in limiting the number of residual UDRs. Moreover, they need fewer extra transistors and consume less power. The comparison between the two STAP architectures depends on the percentage of base defective routers. When the percentage of defective base routers is less than 20%, SDA is definitely the most attractive solution (see Figure 9) due to its simplicity and efficiency.

ACKNOWLEDGMENT

We thank the anonymous reviewers for their excellent feedback.

REFERENCES

[1] A. J. Bhavnagarwala, X. Tang, and J. D. Meindl. 2001. The impact of intrinsic device fluctuations on CMOS SRAM cell stability. *IEEE Journal of Solid-State Circuits* 36, 4 (2001), 658–665.

[2] A. Asenov, A. R. Brown, J. H. Davies, S. Kaya, G. Slavcheva. 2003. Simulation of intrinsic parameter fluctuations in deca-nanometer and nanometer-scale MOSFETs. *IEEE Transactions on Electron Devices* 50, 9 (2003), 1837–1852.

[3] Shekhar Borkar. 2005. Designing reliable systems from unreliable components: The challenges of transistor variability and degradation. *IEEE Micro* 25, 6 (2005), 10–16.

[4] Jayanth Srinivasan, Sarita V. Adve, Pradip Bose, and Jude A. Rivers. 2004. The impact of technology scaling on lifetime reliability. In *Proceedings of the International Conference on Dependable Systems and Networks (DSN'04)*. 177–186.

[5] J. G. Tryon. 1962. *Quadded Logic, Redundancy Techniques for Computing Systems*, R. H. Wilcox and W. C. Mann (Eds.). Spartan Books, Washington, DC, 205–228.
[6] P. A. Jensen. 1963. Quadded NOR logic. *IEEE Transactions* 3, 12 (1963), 22–31.
[7] A. H. El-Maleh, B. M. Al-Hashimi, A. Melouki, and F. Kahn. 2009. Defect tolerant N-transistor structure for reliable nanoelectronic design. *IET Computers & Digital Techniques* 3, 6 (2009), 570–580.
[8] Jin-Fu Li, Jen-Chieh Yeh, Rei-Fu Huang, and Cheng-Wen Wu. 2003. A built-in self-repair scheme for semiconductor memories with 2-D redundancy. In *Proceedings of the International Test Conference (ITC'03)*. 393–402.
[9] M. Nicolaidis, N. Achouri, L. Anghel. 2004. A diversified memory built-in self-repair approach for nanotechnologies. In *Proceedings of the 22nd IEEE VLSI Test Symposium*. 313–318.
[10] Swapnil Bahl. 2007. A sharable built-in self-repair for semiconductor memories with 2-D redundancy scheme. In *Proceedings of the 22nd IEEE International Symposium on Defect and Fault-Tolerance in VLSI Systems (DFT'07)*. 331–339.
[11] H. Cao, M. Liu, H. Chen, X. Zheng, C. Wang, and Z. Wang. 2012. Efficient built-in self-repair strategy for embedded SRAM with selectable redundancy. In *Proceedings of the 2nd International Conference on Consumer Electronics, Communications and Networks (CECNet'12)*. 2565–2568.
[12] M. Lee, Li-Ming Denq, and Cheng-Wen Wu. 2011. A memory built-in self-repair scheme based on configurable spares. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems* 30 (2011), 919–929.
[13] K. Constantinides, S. Plaza, J. Blome, B. Zhang, V. Bertacco, S. Mahlke, T. Austin, and M. Orshansky. 2006. Bullet-proof: A defect-tolerant CMP switch architecture. In *Proceedings of the 12th International Symposium on High Performance Computer Architecture*. 5–16.
[14] D. Fick, A. DeOrio, J. Hu, V. Bertacco, D. Blaauw, and D. Sylvester. 2009. Vicis: a reliable network for unreliable silicon. In *Proceedings of Design Automation Conference*. San Francisco, CA, 812–817.
[15] Shu-Yen Lin, Wen-Chung Sen, Chan-Cheng Hsu, and An-Yen (Ady) Wu. 2009. Fault-router with built-in self-test/self-diagnosis and Fault-isolation circuits for 2D-mesh based chip multiprocessor systems. *International Journal of Electrical Engineering* 16 (2009), 213–222.
[16] J. H. Collet, A. Louri, V. Tulsidas Bhat, and P. Poluri. 2011. ROBUST: A new self healing fault-tolerant NoC router. In *Proceedings of the 4th International Workshop on Network on Chip Architecture*. Porto Alegre, Brazil (NoCArc'11), 11–16.
[17] D. S. Rosenblum. 1995. A practical approach to programming with assertions. *IEEE Transactions on Software Engineering* 31, 1 (1995), 19–31.
[18] Pavan Poluri and Ahmed Louri. 2016. Shield: A reliable network-on-chip router architecture for chip multiprocessors. *IEEE Transactions on Parallel and Distributed Systems* 27, 10 (2016), 3058–3070.
[19] Dominic DiTomaso, Avinash Kodi, Ahmed Louri, and Razvan Bunescu. 2015. Resilient and power-efficient multi-function channel buffers in network-on-chip architectures. *IEEE Transactions on Computers* 64, 12 (2015), 3555–3568.
[20] Pavan Poluri and Ahmed Louri. 2015. A soft error tolerant network-on-chip router pipeline for multi-core systems. *IEEE Computer Architecture Letters* 14 (2015), 107–110.
[21] Dominic DiTomaso, Travis Boraten, Avinash Kodi, and Ahmed Louri. 2016. Dynamic error mitigation in NoCs using intelligent prediction techniques. In *Proceedings of the 49th ACM/IEEE International Symposium on Microarchitecture (MICRO'16)*. Taipei, Taiwan, 15–19.
[22] Dominic DiTomaso, Avinash Kodi, and Ahmed Louri. 2014. QORE: A fault-tolerant network-on-chip architecture with power-efficient quad function channel (QFC) buffers. In *Proceedings of the 20th IEEE International Symposium on High-Performance Computer Architecture (HPCA'14)*. Orlando, FL, 320–331.
[23] Pavan Poluri and Ahmed Louri. 2014. An improved router design for reliable on-chip networks. In *Proceedings of the 28th IEEE International Parallel and Distributed Processing Symposium (IPDPS'14)*. Phoenix, AZ, 283–292.
[24] Pavan Poluri and Ahmed Louri. 2013. Tackling permanent faults in the network-on-chip router pipeline. In *Proceedings of the 25th International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD'13)*. Porto de Galinhas, Brazil, 49–56.
[25] T. Lehtonen, P. Liljeberg, and J. Plosila. 2007. Online reconfigurable self-timed links for Fault-tolerant NoC. *VLSI Design* (2007), 1–13.
[26] T. Lehtonen, D. Wolpert, P. Liljeberg, J. Plosila, and P. Ampadu. 2010. Self-adaptive system for addressing permanent errors in on-chip interconnects. *IEEE Transaction on Very Large Scale Integration (VLSI) Systems* 18, 4 (2010), 527–540.
[27] Di Tomaso, T. Boraten, A. Kodi, and A. Louri. 2012. Evaluation of fault tolerant channel buffers for improving reliability in NoCs. In *55th International Midwest Symposium on Circuits & Systems (MWSCAS'12)*. Boise, Idaho.
[28] W. G. Brown, J. Tierney, and R. Wasserman. 1961. Improvement of electronic-computer reliability through the use of redundancy. *IEEE Transactions on Electronic Computers* EC-10, 3 (1961), 407–416.
[29] R. E. Lyons and W. Vanderkulk. 1962. The use of TMR to improve computer reliability. *IBM Journal* (1962), 200–209.
[30] P. Koopman, K. Driscoll, and B. Hall. 2015. Selection of cyclic redundancy code and checksum algorithms to ensure critical data integrity. Final Report, DOT/FAA/TC-14/49 contract (2015).

[31] 1994. National Technology Roadmap for Semiconductors. Semiconductor Industry Association. Available at http://www.rennes.supelec.fr/ren/perso/gtourneu/enseignement/roadmap94.pdf.
[32] Sriram Vangal, Jason Howard, Gregory Ruhl, et al. 2008. An 80-tile sub-100-W teraflops processor in 65-nm CMOS. In *IEEE Journal of Solid State Circuits* 43, 1 (2008), 29–41.
[33] Chao Cheng and K. K. Parhi. 2006. High-speed parallel CRC implementation based on unfolding, pipelining and retiming. *IEEE Transactions on Circuits and Systems* 53, 10 (2006), 1017–1022.
[34] C. Kennedy and A. Reyhani-Masoley. 2008. High speed parallel CRC circuits. In *Proceedings of the Conference on Signals, Systems and Computers (Asilomar)*. 1823–1829.
[35] S. Singh, S. Sujana, I. Babu, and K. Latha. 2013. VLSI implementation of parallel CRC using pipelining, unfolding and retiming. *IOSR Journal of VLSI and Signal Processing* 2 (2013), 66–72.
[36] J. M. Cazeaux, D. Rossi, and C. Metra. 2004. New high self-checking voters. *Proceedings of IOLTS* (2004), 58–63.
[37] P. Zając and J. H. Collet. 2007. Production yield and self-configuration in the future massively defective nanochips. In *Proceedings of the IEEE Symposium on Defects and Fault Tolerance in VLSI Roma*. 197–205.
[38] P. Zając, J. H. Collet, and A. Napieralski. 2008. Self-configuration and reachability metrics in massively defective multiport chips. In *Proceedings of 14th International On-Line Testing Symposium*. 219–224.
[39] Y. Zou and S. Pasricha. 2010. NARCO: Neighbor aware turn model based fault tolerant routing for NoCs. *IEEE Embedded System Letters* 2, 3 (2010), 85–89.
[40] V. Y. Raparti and S. Pasricha. 2016. CHARM: A checkpoint-based resource management framework for reliable multicore computing in the dark silicon era. In *Proceedings of the IEEE International Conference on Computer Design (ICCD'16)*.

Received December 2017; revised April 2018; accepted June 2018