Received August 28, 2019, accepted September 24, 2019, date of publication September 30, 2019,
date of current version October 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2944490

Defender: A Low Overhead and Efficient Fault-Tolerant Mechanism for Reliable On-Chip Router

NAVEED KHAN BALOCH[1], MUHAMMAD IRAM BAIG[2], AND MASOUD DANESHTALAB[3]
[1]Computer Engineering Department, University of Engineering and Technology at Taxila, Taxila 47040, Pakistan
[2]Electrical Engineering Department, University of Engineering and Technology at Taxila, Taxila 47040, Pakistan
[3]School of Innovation, Design and Engineering, Mälardalen University, 722 20 Västerås, Sweden

Corresponding author: Naveed Khan Baloch (naveed.khan@uettaxila.edu.pk)

This work was supported by the Swedish Knowledge Foundation (KKS) through the DeepMaker and DPAC Projects.

ABSTRACT The ever-shrinking size of a transistor has made Network on Chip (NoC) susceptible to faults. A single error in the NoC can disrupt the entire communication. In this paper, we introduce Defender, a fault-tolerant router architecture, that is capable of tolerating permanent faults in all the parts of the router. We intend to employ structural modifications in baseline router design to achieve fault tolerance. In Defender we provide the fault tolerance to the input ports and routing computation unit by grouping the neighboring ports together. Default winner strategy is used to provide fault resilience to the virtual channel arbiters and switch allocators. Multiple routes are provided to the crossbar to tolerate the faults. Defender provides improved fault tolerance to all stages of routers as compared to the currently prevailing fault tolerant router architectures. Reliability analysis using silicon protection factor (SPF) and Mean Time to Failure (MTTF) metrics confirms that our proposed design Defender is 10.78 times more reliable than baseline unprotected router and then the current state of the art architectures.

INDEX TERMS Network-on-Chip, router architecture, permanent fault tolerance, silicon protection factor, mean time to failure.

I. INTRODUCTION

Advancements in semiconductor technology [1], [2] allow us to fabricate silicon dies with billions of transistors. This facilitates the development of chip multiprocessors (CMPs). To provide communication among multiple cores on the chip, a distinctive architecture, called Network on Chip (NoC) [3], [4], is used. NoC has two types of components (also called blocks); (i) computational blocks and (ii) communication blocks. Computation blocks are processing cores, and communication blocks consist of routers and links, that transfers data between cores in the form of packets from source to destination routers.

As the technology scales down to nanometers and operating frequencies grow higher, NoC components incur various faults [5], [6]. The two classes of faults are permanent and transient faults. Faults that arise either at the production time or due to physical damage during operation of the circuit and continually affect the function of the circuit are called permanent faults. Reasons of permanent faults are time-dependent dielectric breakdown (TDDB) [7], negatively biased temperature instability [8], hot carrier injection [9], and electro-migration [10]. The fault that lasts only for a few clock cycles and temporarily affects the operation of the circuit is called transient fault. These faults are usually arising because of alpha particles striking the logic from packaging material [11], thermal radiations from cosmic rays [12], and manufacturing process variations [13]. A fault produces errors in the operation of the chip, for example causing increased latency, packet loss, packet error, and can create a deadlock. A single fault may paralyze the whole chip if proper precautions are not considered in the design. Hence it is an utmost desire at design stage to create such a circuit that can avoid the faults at later stages. Our focus in this paper is to provide fault-tolerance in each component of the router.

The router in NoC architecture consists of many components like buffers, Routing Computation unit (RC), Virtual channel Allocators (VA), Switch Allocators (SA) and Crossbar (XB). Traversing of the packet through the pipeline of the router is shown in Figure 1.

The associate editor coordinating the review of this manuscript and approving it for publication was Zonghua Gu.

FIGURE 1. Router pipeline stages.

When a packet entered in a router, it arrives in the assigned buffer and then traverses through these pipeline stages, starting from RC to the downstream components. The detailed working of the input port and pipeline components of the baseline router is explained in section 2. The proper operation of these stages is essential for a router to work correctly and to forward the packet to the correct destination. Many researchers have already worked to tolerate the faults occurring in NoC router and links [14]–[18]. However, these approaches do not provide fault tolerance to each component of router. We have proposed the techniques to tolerate the faults occurring on all the components of the router by grouping the input ports for buffers and RC protection, temporal parallelism for VA, rectification circuitry at SA, and the bypass path for the crossbar.

The remaining sections of the paper are presented as follows. In section 2, we provide an overview of baseline router architecture with all the internal components. In section 3, we describe the other permanent fault tolerant router architectures related to our work. In section 4 the problem statement is described in detail. In section 5 the proposed fault-resilient router: Defender architecture is described with fault tolerance capability for all internal components. In section 6, experimental setup is given and a detailed discussion on results for area overhead, reliability and performance analysis is provided. Section, 7 concludes the paper.

II. OVERVIEW OF BASELINE NETWORK ON CHIP ROUTER

There are many router architectures available in the literature such as [19], [20]. We have selected a generic 2-stage router [21] as shown in Figure 2. The baseline router consists of the multiplexers, de-multiplexers, buffers, virtual channels, routing computation (RC) unit, virtual channel allocator (VA), switch allocator (SA) and crossbar (XB).

A. INPUT UNIT

This is the first part of the router which receives the incoming packet. It consists of MUX, DeMUX, and fixed length buffers (as shown in Figure 2). These buffers are named a virtual channel. A single physical channel is shared among all the virtual channels. Each arriving flit kept in a specific VC buffer selected by its VC identifier.

B. ROUTING COMPUTATION UNIT

A packet consists of multiple flits, i.e., head flit, one or more body flits and a tail flit. In routing computation stage, the output port for downstream router is calculated for a head flit of packet as soon as it arrives in the VC. The working procedure of the routing computation stage depends on the routing

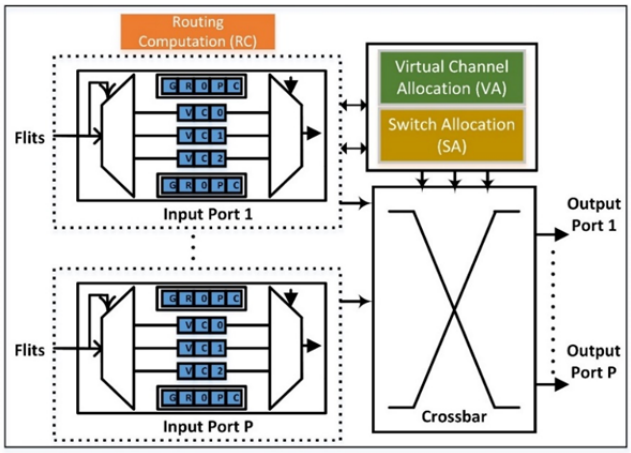

FIGURE 2. Baseline router for network on chip.

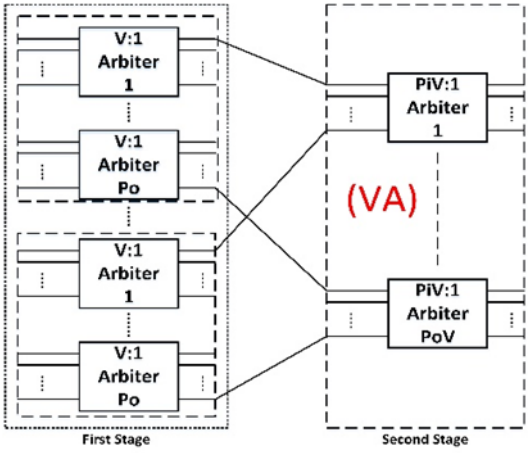

FIGURE 3. Baseline virtual channel allocation.

algorithm. There are many routing algorithms available for the NoC, some are simple dimension order and more complex, i.e., adaptive routing algorithms.

C. VIRTUAL CHANNEL ALLOCATION UNIT

Virtual channel allocation is also performed only for head flit. The functionality of virtual channel allocator is to assign a free virtual channel to each packet at the next router. The process of virtual channel allocation consists of two stages (as shown in Figure 3).

In the first stage a single request is selected from each requesting virtual channel and the second stage removes the conflicts between different input virtual channels that have assigned the identical virtual channel at the downstream router.

D. SWITCH ALLOCATION UNIT

After the virtual channel is allocated to a flit, the next stage is switch allocation. In this stage, a virtual channel competes with other input virtual channels to gain access to the crossbar for a specific output port (as shown in Figure 4). Unlike virtual channel allocation, which is performed only

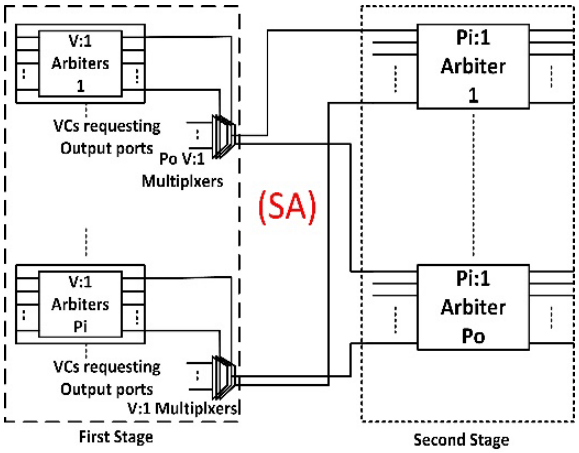

FIGURE 4. Baseline switch allocation.

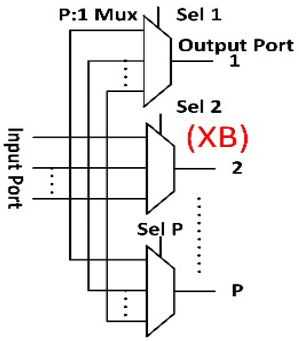

FIGURE 5. Baseline crossbar unit.

for head flit, this stage is performed for each flit. In a generic 2-stage NoC router architecture the two switch allocators are implemented. One is used to handle the speculative requests and other for non-speculative requests. The priority of non-speculative requests is higher.

E. CROSSBAR UNIT

The crossbar is used to connect the input ports to the output ports. The winning virtual channel in the switch allocation stage can now use the crossbar to transmit its flit to the downstream router (as shown in Figure 5). The crossbar connections configure each cycle, which is determined by the winning flit in the switch allocation stage.

Several approaches are proposed as discussed in the next section to improve the performance and reliability of NoC router.

III. RELATED WORK

The preceding methodologies for fault tolerant NoC router design are reviewed in this section. Permanent fault in any component of the router may outcome in complete failure of that router and affects the communication in the network. Researchers have proposed various approaches to tackle these faults. Pavan and Louri [22], [23] proposed the methodology to tolerate permanent faults in the pipeline stages of the router. The proposed fault resilience technique is achieved by using spatial redundancy, resource sharing, and adding some correction circuitry. Each port in the router contains an additional RC unit so that if one RC fails the other can be used to calculate the output port for a packet. VA stage of the router is protected by arbiter sharing within the port. The faults in the SA stage are tolerated by using a default path. Crossbar is protected by providing the additional paths to reach the output port.

Kim *et al.* [24] proposed a router architecture named as RoCo, which disintegrate the components of a router and structure them into rows and columns resulted into smaller crossbars and parallel arbiters. This architecture can have performance degradation in case of faults. When a hard fault occurs in the crossbar connecting north and south output port, the flits trying to reach that output port will be blocked results in performance loss in the network. Constantinides *et al.* [25] suggested a defect tolerant chip multiple processor (CMP) router architecture named as Bulletproof. It is based on spatial redundancy techniques which required multiple copies of the same component to tolerate the hard faults resulting in more considerable area overhead. Bulletproof uses a generic model of a bathtub curve to represent permanent faults. Moreover, bulletproof suggests automatic cluster decomposition model for achieving modularity in router architecture design. It takes a netlist and creates equal size partitions. It divides the fault tolerance process into sub-processes of detection, diagnosis, repair, and recovery.

Fick *et al.* [26] proposed a router architecture named Vicis, which can tolerate permanent faults on links and routers. Vicis employed inherent redundancy to maintain correct operations of the router. Port swapping and bypassing a path is used to tolerate permanent faults in router microarchitecture. Moreover, Vicis used distributed routing algorithm to avoid faults in the network. The faulty link information is collected with the help of the BIST controller and is used to reroute the traffic around faulty links and routers. Vicis used input port swapping algorithm and network rerouting techniques to improve the reliability of the router. To support the input port swapping, the links of Vicis router are functionally bidirectional. Each link consists of two input ports and two output ports. If any of the port is failed, the remaining three ports are available to make new connections. The Vicis router also has a bypass bus to protect the router against crossbar failure.

Mohit et al. in [27] presented a routing algorithm that is designed for the mesh of tree based connected network. This routing algorithm can avoid faulty routers in case of failure. They modified the simple deterministic algorithm for NoC to provide fault tolerance and proved that the proposed algorithm is live-lock free and ensures the delivery of packets to the destination nodes. This can only be used for the transient faults and cannot tolerate the faults occurring inside the router components. In [28] another fault resilient routing algorithm is specified which can provide better performance and scalability to the mesh network in case of faults. They have considered the router and link failure in the network

and show that their algorithm can provide better performance and reduces the hop count compared to the other recent algorithms. These fault tolerant techniques disconnect the healthy cores in the network which is undesired. Khalil et al. in [29] proposed a self-healing router architecture which can tolerate the transient and permanent faults. The fault detection unit available in the router can inform the neighboring routers about the faulty state. In case of faults the router reconfigures itself to be used just as a path for packets. Routing control unit in faulty state is used to calculate the output ports and all the packets coming from different input ports are routed in a round robin fashion. The proposed technique can avoid the internal faults but the router can stop working with single fault in routing control unit or the MUX used for selection in reconfigured state.

Junchi et al. in [30] describe the need for continuous testing in NoC. Built in self-test (BIST) is necessary for detection of faults. They Proposed a framework EsyTest which can detect the faults in NoC very efficiently and utilizes the idle cycles for testing to reduce the performance loss. It can test the data path and control unit separately and tolerate the detected faults by reconfiguration of router or by fault resilient routing algorithm. Hala et al. in [31] presented another fault resilient router architecture and focus on the faults occurring in the input ports. They use the Built-in self-test technique for the detection of faults and by hardware redundancy approach to avoid the faults occurring particularly in the state fields of the virtual channels.

The recovery process can be further classified in term of disabling, ignoring, and replacing faulty components. For tolerating the permanent faults, a simplified approach is the Triple Modular Redundancy (TMR) [32]. In this approach, each component of the router architecture is duplicated or triplicated depending upon the N-modular redundancy approach, where N is the number of additional components available. For example, in a triple modular redundancy approach, each unit has two extra copies. All three units perform the computation, and the final output is determined by a voting system that compares all three results, and the majority output is selected as a final output. In this way, it can tolerate a single fault at the cost of more substantial area overhead, which is undesired. Wang *et al.* [33] have proposed a high-performance reliable (HPR) fault resilient router architecture. They have utilized virtual channel closing for an input port, lookahead routing for routing computation stage, default winning strategy for arbiters and bypass bus to tolerate the crossbar faults.

IV. PROBLEM STATEMENT

There is a need to provide such an architecture which can avoid a greater number of faults in the input ports and pipeline stages. Input ports in the routers consist of VC buffers, DeMUXes and MUXes and gets the most significant share of area and power inside the router. The optimal utilization of these components is essential for reliable packet communication. Shield [23] and HPR [33] are state of the art and higher

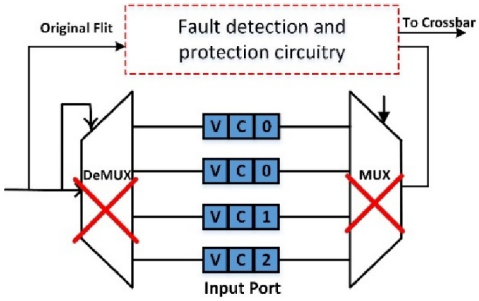

FIGURE 6. MUX and DeMUX failure in input port.

fault tolerating architectures available. Shield [23] can only tolerate the faults in the pipeline components of the router and assumed that input port buffers are well protected by ECC techniques. They ignored the failure of other components present in the input port, i.e., MUX and DeMUX. HPR [33] uses the virtual channel closing technique for fault tolerance in the input ports. They utilize the double bit detection and single bit correction ECC code. When the flit entered in the input port, the ECC code is stored in the particular buffer and compared with the generated code. In the case of a single bit error, the fault recovery unit can correct it otherwise that virtual channel is closed for future use. They are tolerating a total 16 faults in all the input ports of the router using this technique, but as shown in figure 6 with a single MUX and DeMUX failure in HPR [33] input port, all the resources become inoperative, and all the detection and correction circuitry is useless.

So, there is a need to provide more reliability in this unit. We have grouped the adjacent ports to solve this problem. Our architecture provides improved reliability in the input port units and provides enhanced reliability to the RC and VA stages as compared to the HPR [33] with low area overhead. We also propose better fault tolerant architecture to the SA unit by utilizing the default winning strategy. Our proposed router architecture Defender is superior from these techniques in that it provides tolerance to a higher number of faults to the input port and for each pipeline stage.

It utilizes the inherent redundancy in the router microarchitecture to tolerate the permanent faults by utilizing temporal parallelism, rectification circuitry, and multiple routes and offers noticeable improvements over baseline and state of the art router designs.

V. DEFENDER THE PROPOSED FAULT TOLERANT ROUTER ARCHITECTURE

We named our proposed fault-tolerant router architecture as Defender. It provides better reliability compared to the state-of-the-art reliable router architectures. In this section first, we present the effects of faults at each stage, and then we present the techniques used for the detection and tolerance of these faults.

A. DEFENDER: INPUT PORT FAULT TOLERANCE

Input port consists of MUX, DeMUX, and virtual channels and consumes the most significant area of the router.

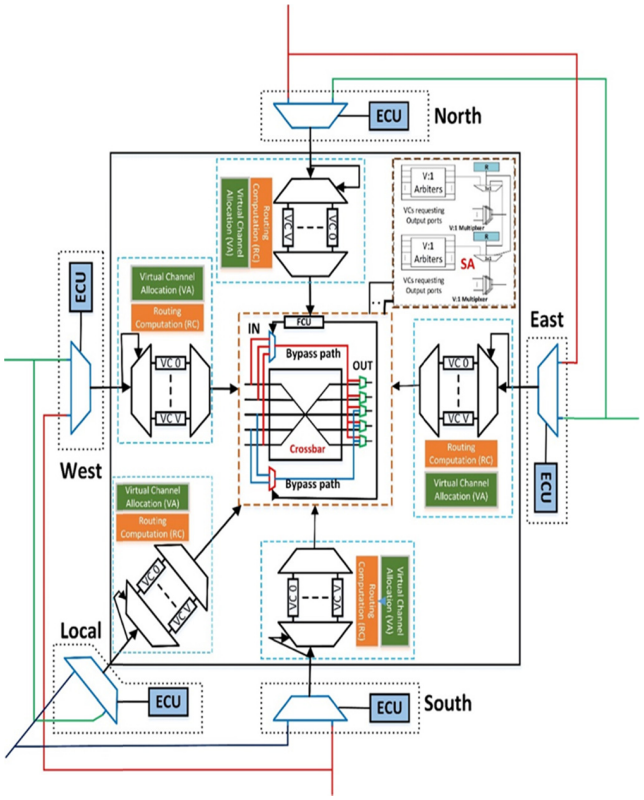

FIGURE 7. Proposed fault tolerant router architecture (Defender).

Therefore, it is evident that protection to all the parts of the input port is necessary otherwise the router may fail its operation, i.e., in case of faulty DeMUX at the start of the input port, flits cannot enter into the router, and all the resources become inoperable. Similarly, permanent faults in the MUX and VC buffers also affects the operation of the input port and block the flits. Network on Chip is a highly redundant architecture. There are five ports in a typical NoC router. To protect the input ports from faults, we grouped the adjacent ports. As shown in Figure 7, we grouped the north with the east port, and another group contains the remaining three south, west, and local ports. The internal resources of these groups are shared among the members in fault situations. Each module that is used for grouping of ports consists of MUXes and an Error Control Unit (ECU) which utilize the NoCAlert checkers [34] for the detection of faults in the input port. Using this module for grouping of ports does not affect the working of other port. Therefore, the failed component in each port is tolerated by the other member of the group. In fault situations, if VC buffers, DeMUX, MUX, VC allocator, RC or the module used for grouping is not working, the error control module of other port can efficiently handle the situation and provide the healthy resources from that port to both the channels. If the incoming channel is faulty, then this architecture can utilize the available resources in that input port, and resource wastage is prevented.

In the proposed shared input port architecture, each input channels have two paths to reach the downstream router,

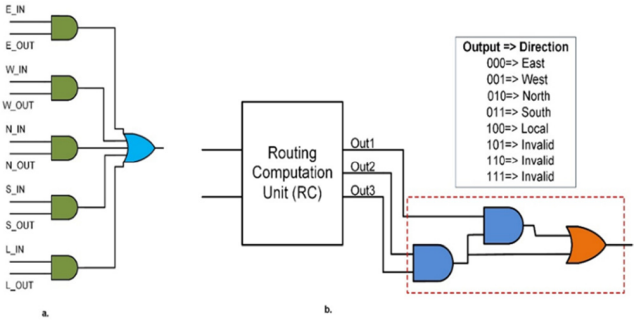

FIGURE 8. RC fault detection checkers (a) non-minimal wrong output port (b) Invalid output port.

therefore any fault resilient deflection routing is not required. The dimension order XY routing algorithm is enough. So, there is no chance for a deadlock or live lock to occur in this architecture. Shield [23] and HPR [33] provides the separate fault tolerant mechanism for RC and VA. The Defender also provides the separate fault tolerant techniques to these units as well as due to the sharing of ports the reliability of these stages is improved. In Figure 7, we have also shown the fault-tolerant architectures for the SA and XB for reference, but it is discussed in the separate sections.

B. DEFENDER: RC STAGE FAULT TOLERANCE

In the RC stage, the next hop is calculated for the packet, which takes it one step closer to the destination router. A faulty RC may calculate the wrong output port which takes the packet away from its destination. In case of adaptive routing, this misrouted packet may reach its destination with increased latency, and in case of deterministic routing, a deadlock may occur in the network. A faulty RC may also calculate the invalid output port. For example, if the router has five ports (numbered 0 to 4), value 5 and onwards are invalid [35].

Figure 8 shows the checkers based of NoCAlert [34] for detection of RC faults in the router. Checker in (Figure 8a) detects the calculation of wrong output port, which can transmit the packet in the wrong direction away from the destination. Checker in (Figure 8b) can detect the invalid output port direction. As shown in the figures, each output port direction is assigned a 3-bit code from 0 to 4. Rest of the numbers in 3-bit representation from 5 onwards are invalid.

Lookahead routing is utilized in 2-stage NoC router, in which a current router 'Q' can calculate the address of downstream router 'Q+1'and calculates the address of next downstream 'Q+2' for a specific head flit. So, the faulty RC computation in 'Q+1' does not affect this stage because the next RC in 'Q+2' router uses the lookahead routing in one additional clock cycle to calculate the output port direction. We exploit this redundant RC computation strategy available in baseline router to avoid faults at this stage, without any additional hardware. Our proposed design is more fault resilient to this stage; if two RC modules in a row become faulty, our architecture allows packets to go to the other member port of the group.

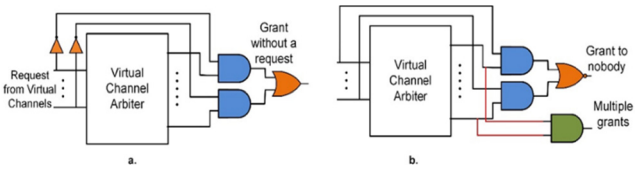

FIGURE 9. VA fault detection checkers (a) Grant without request (b) Grant to nobody, multiple grants.

C. DEFENDER: VA STAGE FAULT TOLERANCE

Virtual Channel Allocation (VA) stage is responsible to allocate a free VC buffer to the incoming packet. Its calculation is based on the destination information available in the head flit. Faults on this stage can allocate a used VC buffer to a new packet, allocation of VC buffer without request and allocation to nobody when there is at least one request. When an inaccurate VC allocation occurs to the occupied VC buffer in the downstream router, then the new flit overwrites the existing flit, results in data corruption. If VA allocates the downstream VC buffer without request, the allocated buffer resource is wasted and cannot be allocated to the needy flit at later times.

A fault when the VA allocates to nobody results in the flit to stay in the current router and results in the deadlock. Fault resilience at this stage can prevent data corruption, prevention against resource wastage and to avoid the deadlock in the network.

Detection and localization of faults is necessary to provide tolerance. We have utilized the detection mechanism given by NoCAlert [34] for this purpose. Checker given in Figure 9a can detect the fault if the downstream VC buffer is allocated without a request for prevention of resource wastage. It consists of simple logic gates for fault detection and gives very less area overhead. Checkers in Figure 9b detect the faults of multiple grants to a simultaneous VC buffer to avoid data corruption and grant to nobody to avoid deadlock in the network.

We utilize the VA fault tolerance mechanism provided by the HPR [33] in which a register is used as a default path if the arbiter is faulty. The output VC's are compared and the one with most free buffers is allocated to the register as default winner, shown in Figure 10.

A 2×1 MUX is used to select the output of the arbiter in the absence of fault and default winner result from the register in the presence of a fault. This technique can provide protection even when all the VA's in the router are faulty, but this fault prevention mechanism is only for the first stage of VA. The second stage of the VA is not tolerated. Our proposed architecture uses this mechanism for tolerating the first stage faults and provide even more resilience against faults because of input port grouping. In the case of faulty VA, the flits can be transmitted to the other port in the group having healthy VA. In this way, our proposed design outperforms the state-of-the-art mechanisms in term of reliability.

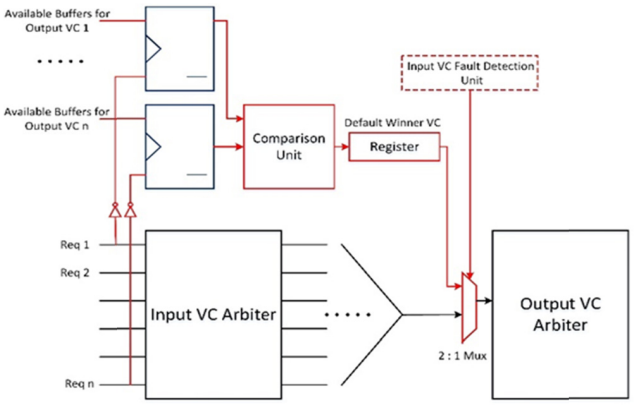

FIGURE 10. VA fault tolerance: Default winner strategy.

D. DEFENDER: SA STAGE FAULT TOLERANCE

Switch allocation stage is responsible for arbitrating between different input VC of the input ports to assign a crossbar time in the next cycle. A faulty SA stops the flits from reaching downstream router by traversing the crossbar. In this situation, the flits remain in the VC buffer and unable to release it, which is the cause of performance degradation and can also create the deadlock in the network. For detection of these faults, we utilized the light weight checkers provided by [34]. The detection circuitry in the form of checkers is given in the previous section.

Dual switch allocators are available in 2-stage NoC router. One is used to handle the speculative requests, and the other one is used for non-speculative requests. This hardware redundancy can be utilized to provide fault tolerance at this stage. In case of a fault, one of the two arbiters can be used for allocation, which is originally proposed by [33]. As shown in the figure in Figure 11, arbiters are selected at runtime by a faulty control unit and 2×1 MUXes. The basic idea is to ensure that the non-speculative requests are handled properly and the output port is reachable. Speculative requests are handled by SA_1_IN_i and SA_1_OUT_j, while the non-speculative requests are handled by SA_2_IN_i and SA_2_OUT_j. If the SA_2_IN_i is faulty, then the fault control unit selects the SA_1_IN_i for non-speculative requests and put the speculative requests to the other faulty arbiter. In this way, the speculative requests are also handled in the next cycle when the healthy arbiter is allocated. This idea works fine, but what if both the arbiters become faulty? We have proposed a solution to this problem by little modification in the internal architecture of the arbiter circuit, as shown in Figure 12 below.

To tolerate the internal faults, we have chosen a bypass path. As the fault control unit detects the fault, it activates bypass path which is a register having default id of the virtual channel. This is achieved by a MUX of 3×1, where one input is from one v:1 arbiter, one input from a register within the same port having default virtual channel id and other from a nearby port register having default virtual channel id. This technique pairs up two ports together, and one port remains

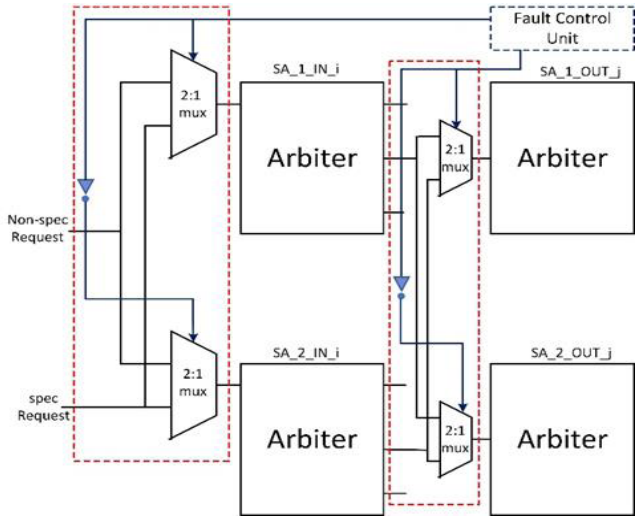

FIGURE 11. SA fault tolerance: Runtime arbiter selection.

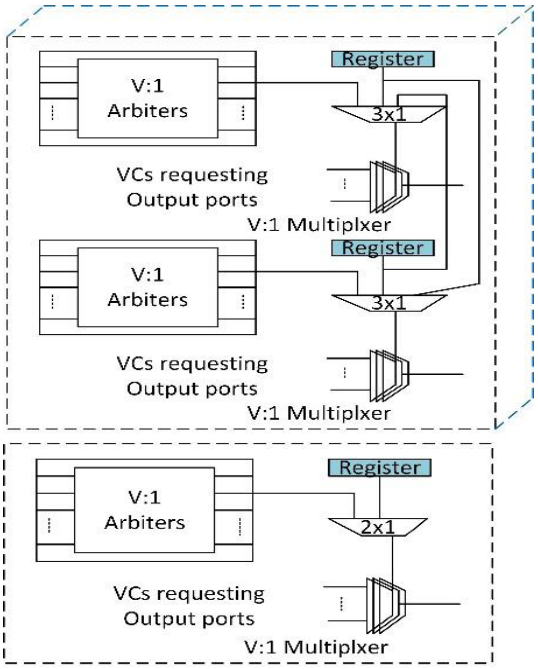

FIGURE 12. SA fault tolerance: Bypass path to tolerate the internal faults.

alone, as shown in Figure 12. Once the arbiter related to an input port is defective first, it selects a bypass path which has default id of any virtual channel. If a fault occurs in the bypass path, then it selects another bypass path which is coming from another port. Now this virtual channel is used as a default virtual channel for participating in the second stage of switch allocation. In this way, fault tolerance for each pair is 4.

E. DEFENDER: XB STAGE FAULT TOLERANCE

The crossbar in the router is used to connect ports. A faulty crossbar restricts the flits to reach the downstream router. Crossbar consists of multiple multiplexers for establishing the connection. A faulty crossbar is detected by confirming

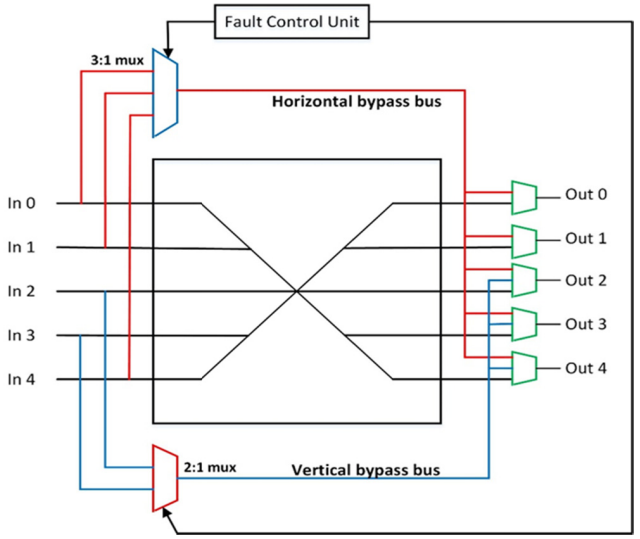

FIGURE 13. Multiple routes to tolerate permanent faults at crossbar.

the defects in the multiplexers. The light weight checkers provided by [34] are used for the detection of permanent faults. The multiplexer is considered as faulty if the selected signal for the multiplexer provides no output on the port.

Fault tolerant crossbar design is shown in figure 13, as initially proposed by [33]. There are two bypass busses in this design one for the X dimension and other for Y dimension. The first bus is used to connect all the input ports of X dimension and the local port to all the output ports. The second bypass bus connects the Y dimension input ports to the output ports and the local port. There is no conflict between these two bypass buses.

In case of a fault, the two busses can transfer two flits at a time which decreases the performance loss. This design tolerates the faults even if all the multiplexers are faulty.

VI. RESULTS AND ANALYSIS

We evaluated the defender our proposed router architecture in terms of area overhead, reliability analysis as Silicon Protection Factor (SPF), and Mean Time to Failure (MTTF). Performance analysis by calculating the latencies using synthetic and real traffic patterns. These results are compared with the baseline and another state-of-the-art architecture.

A. AREA OVERHEAD

We implement the baseline router and Defender: our proposed fault resilient router in Verilog. Cadence Encounter RTL compiler is used for synthesis of the designs in NangateOpenCell 45 nm library. Area overhead is an important factor to confirm the fault tolerance efficiency. The following equation 1 obtains area Overhead

$$\textit{Area Overhead} = \frac{\textit{Fault tolerant design area}}{\textit{Baseline Area}} \quad (1)$$

Figure 14 shows the area overhead of the baseline design with state-of-the-art architectures. It can be seen that defender is

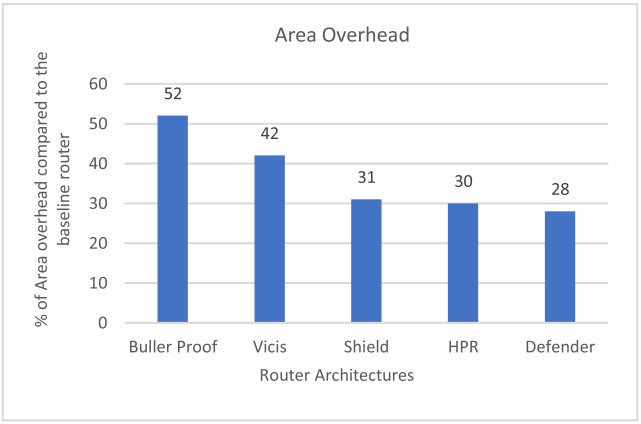

FIGURE 14. Area overhead of fault-resilient router architectures.

more area efficient in providing the fault tolerance compared to previous designs.

B. RELIABILITY IMPROVEMENT COMPARISON USING SPF

There are many types of metrics available to access the reliability of the circuit. Considering the area overhead for this purpose is very beneficial to confirm the fault tolerance efficiency. Increase in area overhead is obvious if we use the spatial redundancy to provide the fault resilience to a circuit. Therefore, it is necessary to consider a metric that utilize area overhead with fault tolerance ability. We are using Silicon Protection Factor (SPF) given by Constantinides et. al in [25] for the comparison of reliability of proposed router with the state-of-the-art architectures.

Silicon Protection Factor of circuit can be calculated by Equation (2):

$$SPF = \frac{Mean\ No\ of\ faults\ to\ cause\ failure}{Area\ Overhead} \tag{2}$$

Here, the mean no of faults to cause failure can be calculated by Equation (3):

$$Mean\ no\ of\ faults = \frac{Min\ faults + Max\ faults}{2} \tag{3}$$

Here *Min faults* and *Max faults* are the minimum and maximum no of faults to failure.

A fault at the input port is defined as a faulty MUX, DeMUX and faulty virtual channels. A faulty MUX or DeMUX can block the entire input port, and all the resources become inoperable. A fault in the virtual channel causes an incorrect flit. In defender, we have grouped the adjacent ports to provide fault tolerance. The internal resources of these groups are shared among the members in fault situations. One paired group provides fault tolerance to 4 VC buffers, MUX, DeMUX, One Error Control Unit (ECU) module used for grouping the ports and 3 VC buffers faults in an adjacent port. So, a total 10 number of faults are tolerated by one paired group. The other group which connects three input ports can also tolerate 10 faults. The router can tolerate a maximum of 20 (10x2) faults at the input ports. The minimum number of faults to cause failure at the input port is 2.

RC is the next stage in the router in which the occurrence of fault results in the calculation of the wrong output port. Our protection strategy to this unit can avoid 7 faults. A generic 2-stage router provides inherent redundancy which can tolerate 5 RC faults in the router. Two more faults can be tolerated by grouping the ports. A failure occurs at minimum two faults if the RC at the downstream router is not working.

After the routing computation, the next stages are VA and SA. In our protection strategy, the SA can work properly in the presence of a maximum 15 faults. Due to the inherent redundancy in 2-stage NoC router our proposed technique can provide guard to non-speculative requests if all the arbiters are failed. To tolerate the internal faults, we have utilized the bypass path inside the SA, as explained in section 5, which can tolerate the 5 more faults. A minimum of 3 faults can cause failure in SA. A maximum of 20 faults are tolerated in the VA unit in our proposed strategy. It works fine even if all the arbiters are faulty. On the other side only a presence of 2 faults can cause failure if the default winner and input VC are faulty simultaneously. The crossbar is used to connect the input ports with the output. A fault in this unit is defined as the faulty MUX. In the proposed technique with the inclusion of two bypass busses in this unit can result in tolerating a total 5 faults in it. Only 2 faults in the crossbar result in failure if both the MUXes and bypass busses are faulty.

The minimum number of faults to cause failure can be calculated as min {2(Input port), 2(RC), 2(VA), 3(SA), 2(XB)}, which is 2 faults. The maximum number of faults tolerated by the router can be calculated to sum all the faults tolerated by various components which are 20(Input port) + 7(RC) + 20(VA) + 15(SA) + 5(XB) = 64. One more fault in the router cause failure in operation. Mean a number of faults to cause failure is 33. Area overhead of the proposed design is 28%. SPF of our proposed router defender is 33 / 1.28 = 25.8

The SPF value of our proposed router is compared with the other designs, as shown in Table 1. Greater the SPF value shows that the architecture is highly reliable, it can be seen that Defender is much more reliable than the state-of-the-art architectures.

C. LIFETIME IMPROVEMENT ESTIMATION USING MTTF

Lifetime improvement comparison of defender with the baseline can be calculated by the Mean Time to Failure (MTTF) [36], [37]. Where, MTTF of a circuit can be calculated by the following equation 4.

$$\frac{10^9}{Failures - in - Time\ (FIT)} \tag{4}$$

Here FIT is the failure of operations per billion hours in a given component. To find out the FIT, we use the Failure in Time Estimation Model proposed in Paluri and Louri [37] Lifetime modeling framework proposed by Shin *et al.* [38] is utilized in this work. The value of FIT for a single field

TABLE 1. SPF comparison for reliability improvement.

Architecture	Faults to cause failure	SPF
BulletProof	3.15	2.07
Vicis	9.3	6.55
RoCo	5.5	<5.5
Shield	15	11.4
HPR	28.5	21.9
[31]	28.5	24.35
Defender	33	25.8

TABLE 2. FIT estimation of baseline 2-stage router.

Unit	FC	FIT of the FC	# of FC's	FIT of the Unit
Input Buffer	128-bit DFF	0.5	40960	20480
RC	6-bit comparator	11.7	10	117
VA	20:1 Arbiter	36.7	20	1468
	20:1 Arbiter	36.7	20	
SA	4:1 Arbiter	7.4	10	215
	5:1 Arbiter	9.3	10	
	4:1 mux	4.8	10	
XB	128-bit 5:1 mux	819.2	5	4096
				26376

effect transistor (FET) which is produced by time dependent dielectric breakdown (TDDB) can be calculated by Equation 5 [7]

$$FIT_{per\ FET} = dutycycle \times \frac{10^9}{A_{TDDB}} \times Vdd^{a-bT} \times e^{\frac{-X+\frac{Y}{T}+ZT}{kT}} \tag{5}$$

Here A_{TDDB}, a, b, X, Y, Z are fitting parameters, k is the Boltzmann's constant, T is operating temperature which is 300 Kelvin and Vdd is operating voltage which is 1V. In equation 5, the dutycycle is set to 100%. The FIT value of basic logic gate then easily can be calculated by multiplying the transistor count with the $FIT_{per\ FET}$ The Sum of Failure (SOFR) [39] model is used to calculate the FIT value of the component and then the entire router. FIT estimation of baseline 2-stage NoC router is calculated by HPR [33] and is given in Table 2.

D. FIT ESTIMATION OF RELIABLE ROUTER: DEFENDER

- **Input Port:** To protect the input port from faults, we have grouped the adjacent ports. If one port is not working, the flits of that port can be transferred to the neighboring port. This technique doesn't need any extra circuitry.

TABLE 3. FIT estimation of reliable router defender.

Unit	Component	FIT of the Unit
VA	20 2-bit DFF	660
	20 20-bit 2:1 Muxes	
SA	20 2:1 Muxes 5 2-bit DFF (registers), 5 3:1 muxes	57
XB	3 128 bit 2:1 muxes	2252.8
	4 128bit 3:1 muxes	

- **RC unit:** No extra circuitry is needed for the protection of the RC unit in our double routing strategy. Moreover, if RC of a port is not working, it can utilize the neighboring port RC for the calculation of the output port.
- **VA unit:** the default winner strategy provides fault tolerance at this stage. 20 2bit registers are used to save the result of default winner and 20 2:1 multiplexer is needed to select the output result from the registers are arbiters.
- **SA unit:** 20 2:1 multiplexer is needed in SA to select the nonfaulty arbiters to ensure the non-speculative switch allocation requests. To provide the inside protection to the SA, we have utilized the 5 2-bit DFF registers for default winner and 5 3:1 multiplexer is utilized to select between the registers and arbiters.
- **XB unit:** Crossbar unit in a reliable router is protected from faults by additional circuitry of four 3:1 multiplexer and three 2:1 multiplexer.

Table 3 shows the FIT estimation of the reliable router for individual components. It also shows the extra component used to avoid the faults.

E. MTTF OF PROPOSED ROUTER: DEFENDER

Using the SOFR model the MTTF value of the baseline 2-stage router is calculated as given in equation 6

$$MTTF_{Baseline} = \frac{10^9}{20480 + 117 + 1468 + 215 + 4096} \approx 37913 hours \tag{6}$$

The FIT of the reliable router using the SOFR model is calculated as $660 + 57 + 2252.8 = 2969.8$. Equation 7 can is utilized to calculate the MTTF value of the reliable router defender and given below.

$$MTTF_{Reliablerouter} = \frac{10^9}{FIT_1} + \frac{10^9}{FIT_2} + \frac{10^9}{FIT_1 + FIT_2} \tag{7}$$

Here FIT_1 is the FIT value of baseline router (26376), and FIT_2 is the FIT value of the reliable router (2969.8). Hence the MTTF value of Defender is 408713 hours, which is 10.78 times to that of baseline router. Defender is 10.78 times more reliable then baseline. Figure 15 shows the comparison of MTTF of other states of the art reliable NoC routers with the Defender, which shows that Defender is more reliable than other fault tolerant routers.

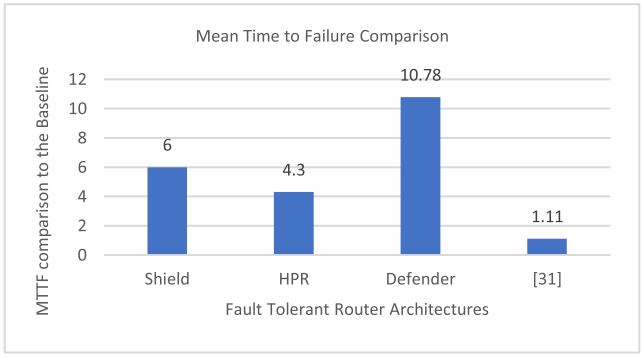

FIGURE 15. Mean time to failure (MTTF) comparison.

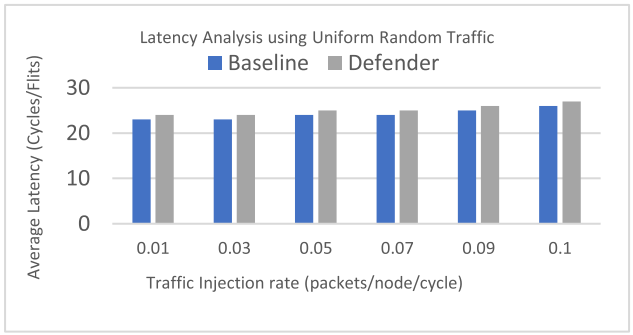

FIGURE 16. Latency analysis using uniform random traffic pattern.

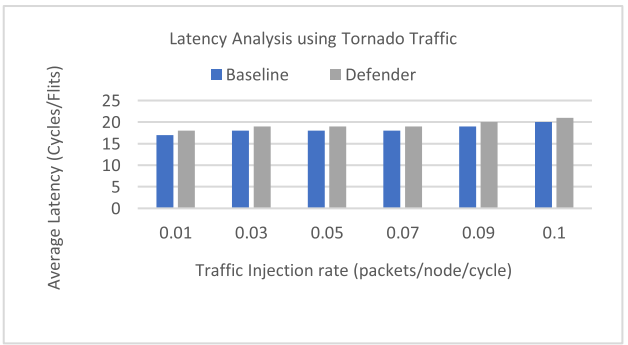

FIGURE 17. Latency analysis using tornado traffic pattern.

F. PERFORMANCE ANALYSIS

For performance analysis, we have calculated the latencies for synthetic and real traffic in the simulator. We use GEM5 [40] integrated with Garnet 2.0 [41] for simulating 8×8 mesh on different synthetic traffic patterns. We make use of uniform random synthetic traffic pattern at injection rates ranges from 0.01 to 0.1. We adopted XY routing protocol for transmitting the packets within the router. The Link latency is assumed to be 1. For each injection rate, simulation is repeated 10 times and taking average value to calculate the average flit latency. The simulation cycles of each injection rate are placed 500000 to calculate the flit latency. We have injected faults randomly in randomly chosen 8×8 router. It is

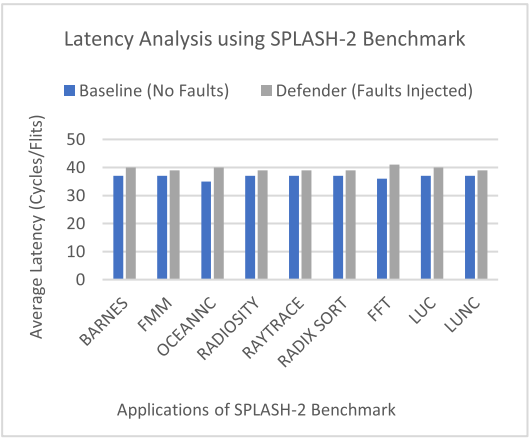

FIGURE 18. Latency analysis using SPLASH-2 benchmark traffic.

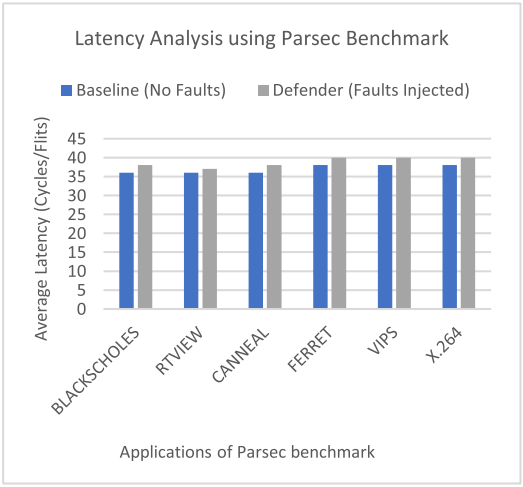

FIGURE 19. Latency analysis using parsec benchmark traffic.

observed that as the number of faults increases the latency of the proposed router architecture also increase. The latency of the router increases up to 6% for Uniform Random traffic pattern as compared to the baseline router architecture, shown in Figure 16.

The same experiment performed for tornado traffic pattern. The latency overhead for tornado traffic pattern is approximately 4% as compared to baseline router architecture, shown in Figure 17.

For analysis with benchmark traffic pattern, we simulated 8×8 mesh-based NoC with each core associated with its cache and directory. The MOESI_CMP_directory coherence protocol is used in the NoC. We simulated NoC with SPLASH-2 [42] and PARSEC [43] benchmark traffic patterns. Figure 18 and Figure 19 shows the latency of proposed reliable router architecture defender in the presence of fault with fault-free baseline design. The average increase in latency for SPLASH2- and PARSEC is 16% and 13% respectively.

VII. CONCLUSION

In this work, we presented Defender, a fault tolerant router architecture capable of tolerating permanent faults in all the parts of the router. We considered each component of the router separately and proposed a fault-tolerant mechanism for that part. The proposed architecture gives minimum latency overhead as shown in the simulation results. The Mean Time to Failure metric use for analysis show that Defender is 10.78 more reliable than the baseline router. The Silicon Protection Factor confirms that the defender can tolerate a greater number of faults with less area overhead compared to the state-of-the-art reliable router architectures. The techniques used in this work for fault tolerance can also be used to the other routers. Overall evaluation of the defender shows that the it can tolerate a greater number of faults with less area overhead compared to the other state of the art reliable router architectures.

ACKNOWLEDGMENT

This work was carried out at the Digital Systems Laboratory in Computer Engineering Department, University of Engineering and Technology Taxila, Pakistan. This work was supported by the Swedish Knowledge Foundation (KKS) through the DeepMaker and DPAC Projects.

REFERENCES

[1] S. Borkar, "Design challenges of technology scaling," *IEEE Micro*, vol. 19, no. 4, pp. 23–29, Jul. 1999.
[2] S. Borkar, "Thousand core chips: A technology perspective," in *Proc. 44th Annu. Design Automat. Conf.*, San Diego, CA, USA, Jun. 2007, pp. 746–749.
[3] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," *IEEE Comput. Soc.*, vol. 35, no. 1, pp. 70–78, Jan. 2000.
[4] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani, "A network on chip architecture and design methodology," in *Proc. ISVLSI*, Pittsburgh, PA, USA, Apr. 2002, pp. 117–124.
[5] H. Zimmer and A. Jantsch, "A fault model notation and error-control scheme for switch-to-switch buses in a network-on-chip," in *Proc. 1st IEEE/ACM/IFIP Int. Conf. Hardw./Softw. Codesign Syst. Synth.*, Newport Beach, CA, USA, Oct. 2003, pp. 188–193.
[6] S. Borkar, "Designing reliable systems from unreliable components: The challenges of transistor variability and degradation," *IEEE Micro*, vol. 25, no. 6, pp. 10–16, Nov. 2005.
[7] S. Oussalah and F. Nebel, "On the oxide thickness dependence of the time-dependent-dielectric-breakdown," in *Proc. IEEE Hong Kong Electron Devices Meeting*, Hong Kong, Jun. 1999, pp. 42–45.
[8] C. E. Blat, E. H. Nicollian, and E. H. Poindexter, "Mechanism of negative-bias-temperature instability," *J. Appl. Phys.*, vol. 69, no. 3, pp. 1712–1720 1991.
[9] G. V. Groeseneken, "Hot carrier degradation and ESD in submicrometer CMOS technologies: How do they interact?" *IEEE Trans. Device Mater. Rel.*, vol. 1, no. 1, pp. 23–32, Mar. 2001.
[10] R. Barsky and I. A. Wagner, "Electromigration-dependent parametric yield estimation," in *Proc. ICECS*, Tel Aviv, Israel, Dec. 2004, pp. 121–124.
[11] T. C. May and M. H. Woods, "Alpha-particle-induced soft errors in dynamic memories," *IEEE Trans. Electron Devices*, vol. ED-26, no. 1, pp. 2–9, Jan. 1979.
[12] J. F. Ziegler, "Terrestrial cosmic ray intensities," *IBM J. Res. Dev.*, vol. 42, no. 1, pp. 117–140, Jan. 1998.
[13] K. J. Kuhn, "Reducing variation in advanced logic technologies: Approaches to process and design for manufacturability of nanoscale CMOS," in *IEDM Tech. Dig.*, Dec. 2007, pp. 471–474.
[14] A. K. Kodi, A. Sarathy, and A. Louri, "Adaptive channel buffers in on-chip interconnection networks—A power and performance analysis," *IEEE Trans. Comput.*, vol. 57, no. 9, pp. 1169–1181, Sep. 2008.
[15] D. Bertozzi, L. Benini, and G. De Micheli, "Error control schemes for on-chip communication links: The energy-reliability tradeoff," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 24, no. 6, pp. 818–831, Jun. 2005.
[16] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw, "A highly resilient routing algorithm for fault-tolerant NoCs," in *Proc. DATE*, Nice, France, Apr. 2009, pp. 21–26.
[17] S. Lin, J. Shi, and H. Chen, "Designing cost-effective network-on-chip by dual-channel access mechanism," *J. Syst. Eng. Electron.*, vol. 22, no. 4, pp. 557–564, Aug. 2011.
[18] D. DiTomaso, A. Kodi, and A. Louri, "QORE: A fault tolerant network-on-chip architecture with power-efficient quad-function channel (QFC) buffers," in *Proc. 20th HPCA*, Orlando, FL, USA, Feb. 2014, pp. 320–331.
[19] L. S. Peh and W. J. Dally, "A delay model and speculative architecture for pipelined routers," in *Proc. HPCA*, Monterrey, Mexico, Jan. 2001, pp. 255–266.
[20] W. J. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*. Amsterdam, The Netherlands: Elsevier, 2004.
[21] A. Jantsch and H. Tenhunen, *Networks on Chip*, vol. 396. Norwell, MA, USA: Kluwer, 2003.
[22] P. Poluri and A. Louri, "Tackling permanent faults in the network-on-chip router pipeline," in *Proc. SBAC-PAD*, Porto de Galinhas, Brazil, Jan. 2014, pp. 49–56.
[23] P. Poluri and A. Louri, "Shield: A reliable network-on-chip router architecture for chip multiprocessors," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 10, pp. 3058–3070, Oct. 2016.
[24] J. Kim, C. Nicopoulos, D. Park, V. Narayanan, M. S. Yousif, and C. R. Das, "A gracefully degrading and energy-efficient modular router architecture for on-chip networks," *ACM SIGARCH Comput. Archit. News*, vol. 34, no. 2, pp. 4–15, May 2006.
[25] K. Constantinides, S. Plaza, J. Blome, B. Zhang, V. Bertacco, S. Mahlke, T. Austin, and M. Orshansky, "BulletProof: A defect-tolerant CMP switch architecture," in *Proc. IEEE HPCA*, Austin, TX, USA, Feb. 2006, pp. 5–16.
[26] D. Fick, A. DeOrio, J. Hu, V. Bertacco, D. Blaauw, and D. Sylvester, "Vicis: A reliable network for unreliable silicon," in *Proc. 46th ACM DAC*, San Francisco, CA, USA, Jul. 2009, pp. 812–817.
[27] M. Upadhyay, M. Shah, P. V. Bhanu, S. J, and L. R. Cenkeramaddi, "Fault tolerant routing methodology for mesh-of-tree based network-on-chips using local reconfiguration," in *Proc. HPCS*, Orleans, France, Jul. 2018, pp. 570–576.
[28] D. Sinha, A. Roy, K. V. Kumar, P. Kulkarni, and J. Soumya, "D_n-FTR: Fault-tolerant routing algorithm for Mesh based network-on-chip," in *Proc. RAIT*, Dhanbad, India, Mar. 2018, pp. 1–5.
[29] K. Khalil, O. Eldash, and M. Bayoumi, "Self-healing router architecture for reliable network-on-chips," in *Proc. 24th ICECS*, Batumi, Georgia, Dec. 2017, pp. 330–333.
[30] J. Wang, M. Ebrahimi, L. Huang, X. Xie, Q. Li, G. Li, and A. Jantsch, "Efficient design-for-test approach for networks-on-chip," *IEEE Trans. Comput.*, vol. 68, no. 2, pp. 198–213, Feb. 2018.
[31] H. J. Mohammed, W. N. Flayyih, and F. Z. Rokhani, "Tolerating permanent faults in the input port of the network on chip router," *J. Low Power Electron. Appl.*, vol. 9, no. 1, p. 11, Feb. 2019.
[32] 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. DATE*, Munich, Germany, vol. 2, Mar. 2005, pp. 1290–1295.
[33] L. Wang, S. Ma, C. Li, W. Chen, and Z. Wang, "A high performance reliable NoC router," *Integration*, vol. 58, pp. 583–592, Jun. 2017.
[34] A. Prodromou, A. Panteli, C. Nicopoulos, and Y. Sazeides, "NoCAlert: An on-line and real-time fault detection mechanism for network-on-chip architectures," in *Proc. MICRO*, Vancouver, BC, Canada, Dec. 2012, pp. 60–71.
[35] D. P. Gaver, "Time to failure and availability of paralleled systems with repair," *IEEE Trans. Rel.*, vol. 12, no. 2, pp. 30–38, Jun. 1963.
[36] P. Ramachandran, S. V. Adve, P. Bose, and J. A. Rivers, "Metrics for architecture-level lifetime reliability analysis," in *Proc. ISPASS*, Austin, TX, USA, Apr. 2008, pp. 202–212.
[37] P. Poluri and A. Louri, "An improved router design for reliable on-chip networks," in *Proc. IPDPS*, Phoenix, AZ, USA, May 2014, pp. 283–292.
[38] J. Shin, V. Zyuban, Z. Hu, J. A. Rivers, and P. Bose, "A framework for architecture-level lifetime reliability modeling," in *Proc. IEEE/IFIP DSN*, Edinburgh, U.K., Jul. 2007, pp. 534–543.
[39] T. Williams, "Probability and statistics with reliability, queueing and computer science applications," *J. Oper. Res. Soc.*, vol. 34, no. 9, pp. 916–917, Jan. 1983.

[40] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The gem5 simulator," *ACM SIGARCH Comput. Archit. News*, vol. 39, no. 2, pp. 1–7, May 2011.
[41] N. Agarwal, T. Krishna, L.-S. Peh, and N. K. Jha, "GARNET: A detailed on-chip network model inside a full-system simulator," in *Proc. ISPASS*, Boston, MA, USA, May 2009, pp. 33–42.
[42] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: Characterization and methodological considerations," *ACM SIGARCH Comput. Archit. News*, vol. 23, no. 2, pp. 24–36, Jun. 1995.
[43] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The PARSEC benchmark suite: Characterization and architectural implications," in *Proc. ACM/PACT*, Toronto, ON, Canada, Oct. 2008, pp. 72–81.

NAVEED KHAN BALOCH received the B.Sc. degree in computer engineering from the University of Engineering and Technology at Taxila (UET Taxila), Pakistan, in 2007, and the M.S. degree from the UET Taxila, where he is currently pursuing the Ph.D. degree. He was an Embedded System Designer with multinational companies, from 2007 to 2010. He joined UET Taxila as a Lecturer. He has published many research articles in his field and has experience in embedded system designing, fault-tolerant systems, and reconfigurable computing. He is currently working on self-healing digital systems. During his tenure in the academia, he did many collaborations with industry and foreign universities in the fields of on-chip networks, embedded vision, and reconfigurable computing. He is also an Assistant Professor with the Computer Engineering Department, UET Taxila.

MUHAMMAD IRAM BAIG is currently a Professor with the Electrical Engineering Department, University of Engineering and Technology at Taxila, Pakistan, where he is also the Director of the Centre of ASIC Design and DSP and creates facilities for experimentation and research. He has published many research articles in fault-tolerant systems, reconfigurable computing, and embedded systems. He did many collaborations in the industry and academia and delivered many successful projects on ASIC design and FPGA. His current research interests include FPGA-based accelerators for deep learning algorithms, high-performance computing, and fault-tolerant designs.

MASOUD DANESHTALAB was a University Lecturer and the Group Leader with the University of Turku, Finland, from 2012 to 2014. He joined the Royal Institute of Technology (KTH), Sweden, as an European Marie Curie Fellow, in 2014. He is currently a tenured Associate Professor with Mälardalen University (MDH) and a Researcher Visitor with KTH. He has published one book, four book chapters, and over 180 refereed international journals and conference papers within an H-index of 24. His research interests include on-/off-chip interconnection networks, hardware/software co-design, reconfigurable computing, neuromorphic architectures, and evolutionary multiobjective optimization. He is also a Technical Program Committee Member of several IEEE and ACM conferences, including DAC, DATE, ASPDAC, ICCAD, NOCS, ESTIMedia, VLSI Design, ICA3PP, SOCC, VDAT, DSD, PDP, ICESS, Norchip, MCSoC, CADS, EUC, DTIS, NESEA, CASEMANS, NoCArc, MES, HPIN, PACBB, MobileHealth, and JEC-ECC. He has served as a Guest Editor for several prestigious journals, such as *Computing* (Springer), IET CDT, ACM TECS, ACM JETC, JSA (Elseviers), MICPRO, *Integration*, and CEE. He has been appointed as an Associate Editor of the *Journals of Computers & Electrical Engineering* (CAEE, Elsevier) and *Microprocessors and Microsystems* (MICPRO) along with the *World Research Journal of Computer Architecture* (JCA). He is on the Editorial Board of *The Scientific World Journal*, the *International Journal of Distributed Systems and Technologies* (IJDST), the *International Journal of Adaptive, Resilient and Autonomic Systems* (IJARAS), the *International Journal of Embedded and Real-Time Communication Systems* (IJERTCS), and the *International Journal of Design, Analysis and Tools for Integrated Circuits and Systems* (IJDATICS). He is also the Key Organizer of the SIGMARC ACM/IEEE International Workshop on Network-on-Chip Architectures (NoCArc/eight editions) and ACM International Workshop on Many-Core Embedded Systems (MES/four editions). He has represented Sweden in the Management Committee of the EU COST Actions IC1202: Timing Analysis on Code-Level (TACLe). Since 2016, he has been on the Eurimicro board of Director and a member of the HiPEAC Network.