Fault-Tolerant Network Interface for Spatial Division Multiplexing Based Network-on-Chip

Anup Das, Akash Kumar, Bharadwaj Veeravalli
Department of Electrical & Computer Engineering
National University of Singapore, Singapore
{akdas, akash, elebv}@nus.edu.sg

***Abstract*—The progressive maturity of VLSI manufacturing technology is helping in integrating more and more processing elements and memory units on a single die to form a Multi-processor System-On-Chip (MPSoC). Network-on-Chip (NoC) is adopted as communication backbone for most of these modern day multiprocessor systems. As complexity of these system scales, there has been a growing concern on the dependability of these processing and communication elements. In this paper, we propose a centralized hardware fault-tolerant network interface (NI) for NoCs based on spatial division multiplexing. Experiments show that the proposed design has better throughput than a non fault-tolerant design with only 18% area overhead. We also introduce an area optimized distributed fault-tolerant NI architecture which provides 50% more throughput than the centralized design for high fault rates.**

***Index Terms*—Fault-Tolerance; Network-on-Chip; Spatial Division Multiplexing; Network Interface**

I. Introduction

As VLSI manufacturing technology is getting more and more mature, the tendency to pack millions of transistors on a single die is also on rise [1]. To extract maximum flexibility and improve performance, increasingly more and more processing elements (PEs) are getting integrated on a single die. Intel Corporation announced teraflop research chip in 2008 comprising of 80 cores [2]. As System-on-Chip (SoC) technology scales, demand for scalable communication architecture is also on the rise. To interconnect different elements in an SoC, network-on-chip (NoC) was proposed by Benini et al. in [3] as an efficient and scalable alternative to shared bus.

The major focus of the NoC research community has been based on time division multiplexing (TDM) wherein, data packets for multiple connections are transmitted over the same wire at different instances of time [4]. However, most of the TDM-NoCs have considerable area and power overhead. One of the effort to reduce area and power is to use NoCs based on spatial division multiplexing (SDM) [5]. As opposed to conventional TDM-based NoCs, which utilize the concept of packet switching, SDM-based NoCs provide guaranteed throughput by providing dedicated wires between source and destination PEs.

As more and more PEs are interconnected on a single chip using NoC (both TDM-based and SDM-based), there are growing concerns on dependability of these PEs and the underlying network backbone i.e. the NoC. Shrinking transistor geometries and aggressive voltage scaling are contributing to permanent faults [6]. This is affecting chip yield. Criticality of this problem is unimaginable for real time systems where presence of a single fault can cost human lives. One of the major challenges in designing for deep sub-micron era is to build a system with built-in fault-tolerance so as to provide graceful degradation of performance rather than complete halt in event of faults. Quite a few works have been done towards achieving fault-tolerance in NoCs [7]–[15]. Most of these works either target faults in network switches or look at system level. Very few designs exist for protecting network interface (NI), which plays a critical role in inter-PE communication.

Key Contributions – In this paper, we improve the design of NI for SDM-based NoC so that it can tolerate hardware faults with minimal area and power overhead. The key contributions are listed below.

- *A centralized fault-tolerant NI which can tolerate multiple hardware faults*. When a fault occurs in any component, a controller performs connection remapping so that faulty hardware is isolated and operation is continued using available functional hardware.
- *A new NI architecture based on control flow*. Data from a PE is pushed to NI only when an external outgoing wire is free to receive data. The design is area efficient as compared to the existing design. Additionally, this design is scalable with the number of connections.
- *A distributed fault-tolerant NI design based on the new architecture*. Different components of the design can adapt to a faulty condition.

For the purpose of evaluation, we have considered an SDM-based NoC with 8 wires on each link. We have synthesized the designs using UMC 65nm cells and using Synopsys Design Compiler. Results show that the centralized design only has 18% more area and 27% more power as compared to the existing NI of an SDM-based NoC. We also introduce an area optimized NI design where multiple components can adapt to faults in a distributed manner. The centralized and the distributed designs are compared with a standard triple modular redundancy (TMR) based NI. We see that the proposed designs are area and power efficient, and deliver similar throughput to that of a TMR-based NI for lower fault-rates (less than 10%). However, as the fault rate increases, the throughput degradation of the centralized design is worse. The distributed design can sustain and deliver 50% througput when the centralized design completely breaks at higher fault rates.

This paper is organized as follows. In Section II we give an introduction to prior work of fault-tolerance in NoC. We then introduce the readers to SDM-based NoCs and the architecture of NI used as reference design in Section III. The details

of the fault classification, detection model and metric for performance are provided in Section IV. We introduce the details of the centralized and the distributed fault-tolerant NI design in Section V and VI respectively. Section VII presents the experimental setup and results. Finally, Section VIII concludes the paper with some recommendations for designers.

II. Related Work

The existing fault-tolerance work on NoC can be broadly classified into two categories – connection-level and hardware-level fault-tolerance. A connection-level fault-tolerance technique reconfigures existing connections or reroutes packets under faulty conditions. As opposed to this, a hardware-level fault-tolerance technique relies on reconfigurable hardware components to mitigate the impact of faults.

There are multiple schemes for connection-level fault-tolerance. One of the widely used schemes is flooding algorithm [7]. Every node in a NoC forwards the incoming data packet to a subset of its adjacent nodes till the data packet reaches its destination node. There are different variants of flooding algorithm like probabilistic flooding, directed flooding, etc. In probabilistic flooding, the flood (multiple copies of the same packet) is randomly distributed in all directions, whereas the directed flooding takes a packet destination into account when forwarding the packet.

Another popular scheme for connection-level fault-tolerance is rerouting of packets when a fault is detected. Non-adaptive routing schemes like XY routing exist where packets are first transmitted along X direction and then in Y direction before they reach the receiver [8]. In times of a fault, one of the other shortest paths is selected between the sender and receiver. There are schemes like dynamic routing [9] based on shortest path from source PE to destination PE. Both flooding and re-routing are not suitable for SDM-based NoCs, where connections are established at connection setup time and remain unchanged throughout the connection lifetime.

For hardware fault-tolerance, main focus has been on protecting network links and switches. There are architectures capable of recovering from permanent switch faults by replacing them with neighboring switches [10]. There are also techniques of using spare routers in the NoCs [14]. Another method employs the concept of reconfigurable routers by using spare buffers of adjacent routers in times of faults [11]. Built-In-Self-Test (BIST) hardware has also been proposed for each router to determine the fault site and select between ECC, port swapping and crossbar bypass [15]. However, all these methods protect only the switches and not the NI.

There are some designs to make the network tolerant to faults in NIs, e.g. connecting each core to more than one NIs. Such multi-NI technique is discussed in [12] and [13]. The design has high area overhead and can become a bottleneck if more than one cores share the same NI in the event of a fault. This multi-NI technique is also not suitable for SDM-based NoCs since multiple connections need to be established between each pair of source and destination PE. This limits the total number of simultaneous connections possible.

The most recent work towards fault-tolerant NI is [16]. Here, the authors propose a two-level fault-tolerance to mitigate faults in NI. They used a mix of Hamming Code for soft errors and limited redundancy for the hard faults. Their design is targeted for NI of a TDM-based NoC. The key architectural difference between NI of a TDM and SDM-based NoC is that, for a TDM-based NoC, a NI transmits data packet for one connection at a time over parallel wires. On the other hand, a NI of an SDM-based NoC simultaneously transmits data packets of multiple connections serially over the wires. Moreover, since switching is not required after a connection setup, the LUTs are not necessary in the NIs. Thus, the design technique of [16] is not suitable for the an SDM-based NoC.

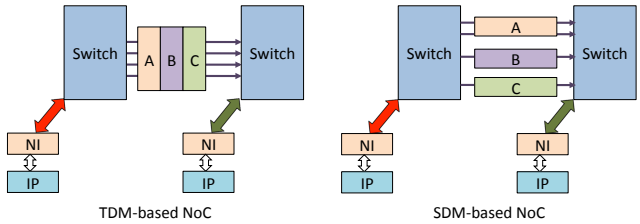

Fig. 1. Difference between TDM and SDM-based NoCs

III. Spatial Division Multiplexing based NoCs

SDM-based NoCs have been proposed as an alternative to TDM-based NoCs [5]. In SDM-based NoCs, a subset of available wires is dedicated between a pair of PEs to form a connection. Each connection thus has exclusive usage of the wires assigned to it. Data from PEs is serialized at transmitter NI and sent over the wire. At the receiving end, data is de-serialized at the receiver NI. Figure 1 shows the difference between an SDM-based and a TDM-based NoC. For a TDM-based NoC, three different connections A, B and C are time multiplexed over the wires. For an SDM-based NoC, data for all connections is transmitted simultaneously over multiple wires. The number of wires dedicated for a connection is bandwidth dependent. In Figure 1, two of the wires are assigned to connection A and one each to connection B and C. SDM networks do not require any switching at the routers once the connections are configured; it is essentially circuit switched once the data leaves the network interface. Thus, packets no longer need to be buffered in the switches saving area and power. However, serialization and de-serialization of data add to complexity in the NI.

SDM-NoCs are power and area efficient as compared to the TDM-based NoCs. The authors in [5] observed 8% reduction in power and 31% reduction in area for an 8x8 NoC. These savings have the potential to drive future MPSoCs employing hundreds of cores. An area-efficient NI architecture for SDM-based NoC, proposed in [17] is used as the base design for all comparison in this paper. This is shown in Figure 2. The design consists of two sections – transmitter and receiver. The transmitter consists of multiple n-to-1 bit serializers. A PE delivers data packets for an outgoing connection to a fifo which in turn forwards the packet to an attached data distributor. The number of distributors in the NI limits the number of connection that can be established from the corresponding PE. Each distributor can deliver data to any of the serializers. This is shown by the "or" gates in front of each serializer. During connection setup, distributors are programmed to deliver data to a subset of the

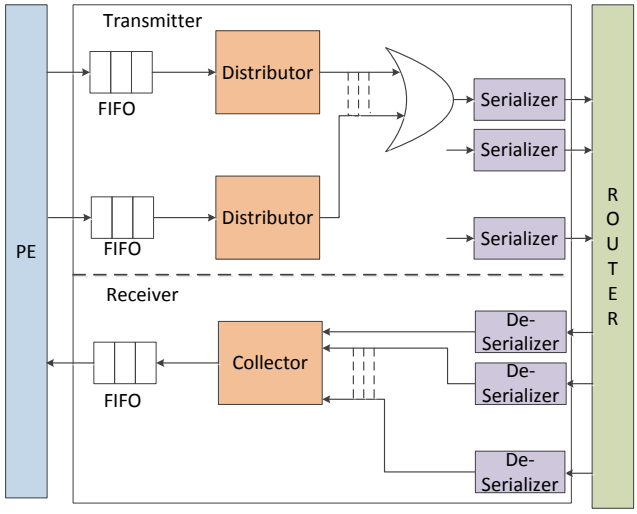

Fig. 2. Base NI Design

serializers. This subset depends on the destination node and the bandwidth requirement for the connection.

The receiver end is just the compliment of the transmitter. The serialized data arriving at the NI is first deserialized and then sent to the collector, which delivers them to the PE through the input fifos. Without loss of generality, the transmitter section is emphasized in the remainder of the paper.

The data distributors are the most critical component of an SDM-based NI. While faults in the serializers can be easily managed using connection remapping, any fault in the data distributors will render a complete outgoing connection infeasible. The network interface needs to be made tolerant to such distributor faults to ensure no data is lost and the number of connections of a PE is not affected. The focus of the current research is therefore on making the NI tolerant to hardware faults in the distributor.

IV. Permanent faults in Network-on-Chips

With shrinking feature size, dependability is becoming major concern even for NoCs. It is predicted that 20%-30% fault rate is expected in future MPSoCs [18]. As in most components in SoC, Networks-on-Chip also suffers from two categories of faults – permanent and transient. Transient faults are intermittent and can occur due to alpha or neutron emissions. Transient faults are measured in flit error rate. Prior work in transient fault-tolerance reveals a flit-rate between 10^{-9} to 10^{-12} for NoC [19]. Permanent faults, as the name itself indicates, are damages to the circuit caused by such phenomena like electro migration, dielectric breakdowns, broken wires etc. These faults are caused during manufacturing or during the product lifetime due to component wear-out. Behavior of a system under permanent faults is time invariant.

Permanent faults are usually modeled as fail-stop where a complete module shuts-down and informs other modules about the same. These faults are described in terms of Mean-Time-Between-Failures (MTBF) and the failure rates are mostly expressed in Failures in Time (FIT). Although permanent faults are less frequent than transient faults, however, recovery from permanent failures are extremely crucial for a NoC (or any component) to continue its operation albeit some acceptable performance degradation.

The design technique proposed in this paper deals with permanent faults in network interface design, caused during manufacturing as well as due to component wear-out. In this section, we provide details of the fault model used together with the diagnosis technique and performance metric.

A. Diagnosis

There are different techniques for diagnosis of faults for networks-on-chip [20] [21]. The current research is orthogonal to any fault diagnosis mechanism. However, scan-chains are built in the design to support scan based failure detection.

B. Fault Model

We classify the faults occurring in the Network Interface into two categories.

- **Data Fault:** Data formatting or processing inside the NI can be corrupted by the presence of one or more hardware faults inside NI. This type of data faults can occur in the fifo, the distributor or the serializers.
- **Channel Fault:** Any fault in the channel can permanently disable a link from future use.

In this paper we deal with the data path faults occurring in the network interface of NoC. Link faults are mitigated by dedicating fewer wires for a connection or using higher frequency for data transfer.

C. MTBF Measure

As has been established in Section III, the major components of an NI are the distributors, collectors and serializers. In this section we describe techniques to improve the MTBF of the components of an NI to make it fault-tolerant.

Serializers :- The base design of NI (Figure 2) provides support for tolerating faults in the serializers. Specifically, to satisfy a connection requirement, multiple wires (and the corresponding serializers) are allocated. Assuming a fail-stop model, once a serializer becomes faulty, the defect information is fed back to the associated core which changes the number of allocated wires/serializers for that connection. This can potentially degrade the throughput if no extra serializers are left to be allocated. Thus a graceful degradation of performance is guaranteed under faulty condition.

Distributors/Collectors :- The distributors (or collectors) are the heart of a NI design. They are responsible for distributing (or collecting) data to (or from) the correct serializers. The same technique for fault-tolerance of serializers cannot be adopted for distributors due to the fact that there is only one distributor for a connection as opposed to multiple serializers per connection. Two possible techniques of fault-tolerance involve introducing redundancy or hardware reconfiguration. It has been shown in [22], the reliability of M-of-N redundancy systems [1]and that of hardware reconfiguration systems are given by Equation 1 where $R(t)$ is the reliability of each distributor and R_{ru} is the reliability of the reconfiguration

[1]N identical modules with output dependent on M modules. TMR is 2-of-3 system

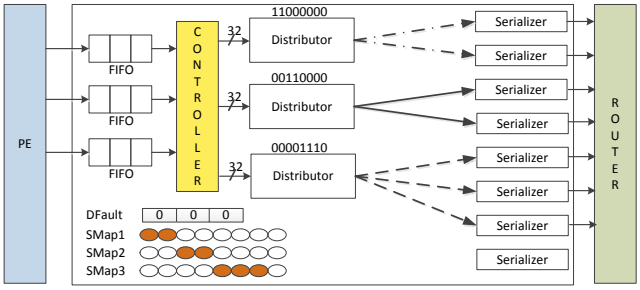

Fig. 3. Centralized fault-tolerant NI design

unit. This assumes independent failure rates. The reliability of a M-of-N system and that of reconfigurable system are higher than that of the reliability of a single distributor. The MTBF is related to reliability according to equation 2.

$$R_{M\_of\_N}(t) = \sum_{i=M}^{N} \binom{N}{i} R^i(t)[1 - R(t)]^{N-i} \quad (1)$$

$$R_{Reconfig}(t) = R_{ru}(t)(1 - [1 - R(t)]^{N+1})$$

$$MTBF = MTTF + \ Mean\ Time\ To\ Repair \quad (2)$$

$$MTTF = \int_0^\infty R_{overall}(t)\,\mathrm{d}t$$

As seen from these equations, MTBF of reconfiguration-based and M-of-N systems are more than the base design. In this paper we therefore propose and evaluate two architectures for fault-tolerant NI – one based on dynamic reconfiguration (Centralized) and the other based based on redundancy (Distributed). Although, the centralized design models the reconfiguration based system, but there are no spare distributors, instead they serve a connection and assume additional responsibilities under a faulty condition. Similarly, the distributed design models an M-of-N system, but the architecture is improved to avoid multiple distributors per connection. Thus significant area and power savings are obtained as compared to a typical M-of-N systems but providing the same reliability.

D. Fault Metric

There has been some research on fault-tolerant metric for NoC infrastructure [23]. For the current research, we use the following three metrics for comparison.

- Power
- Area
- Throughput

We use the above three metrics for comparison of the architectures proposed here. Based on the fact that failure probability increases with area of a component, we also use throughput per unit fault-density (fault per unit area) as a metric of comparison.

V. Centralized Fault-Tolerant Approach

In the centralized architecture, a controller is introduced between data distributors and output fifos of a PE. Whenever data is available, the controller pushes the data from an output fifo to a distributor. When a fault occurs in a distributor, the controller maps the corresponding output fifo to another distributor.

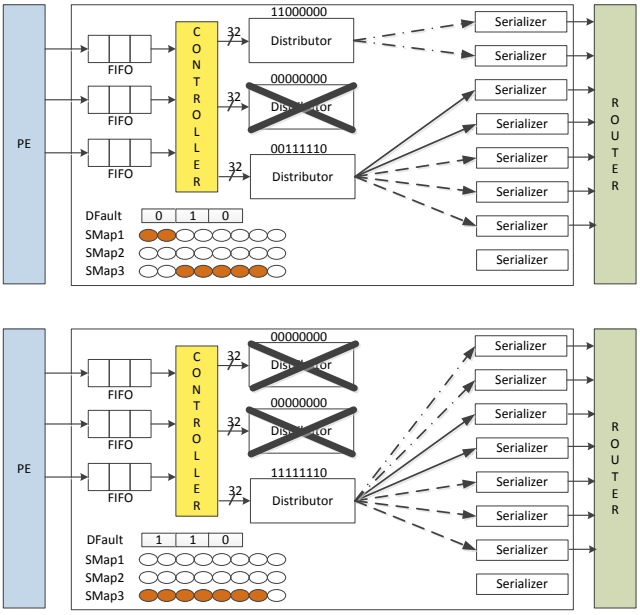

Fig. 4. Operation of centralized design

To perform distributor and serializer allocation, the controller maintains d $n-$bit serializer maps (SMap), where d is the number of distributors and n is the number of serializers. SMap represents the serializers that a distributor needs to activate for data transmission. Further, a $d-$bit DFault register is maintained to indicate the fault status of all distributors.

Considering a fault-free scenario, all positions of the DFault register are set to 0. The SMaps are filled based on the destination PE and the bandwidth requirement of a connection. Figure 3 represents a scenario when three connections are established with 2, 2 and 3 links respectively with all distributors functioning correctly. For ease of representation, we have omitted the "or" gates from the inputs of each serializer.

When a permanent fault is detected in any distributor, either during production testing or during the device lifetime, the corresponding location of the DFault is permanently set as 1 and the controller reconfigures the mapping accordingly. The runtime reconfiguration of centralized fault-tolerant NI design is shown in Figure 4 for single and double-fault scenarios. The controller checks the DFault register and directs the data for the faulty distributor to one of the free distributors. In case no distributor is free, one of the busy distributors is chosen. The controller ensures that the data for the fault-free distributor is given highest priority over the data for the faulty distributor(s). As shown in the figure, three distributors are active for three connections from a PE. If a fault is detected in one of the distributors, say distributor 2, the corresponding bit of the DFault is set to 1. Since there is no free distributor, data originally meant for distributor 2 is mapped to distributor 3 in our example. This is shown with the change in the SMap for distributor 3. In case of a further fault in distributor 1, the entire load is transferred to distributor 3. This results in graceful performance degradation.

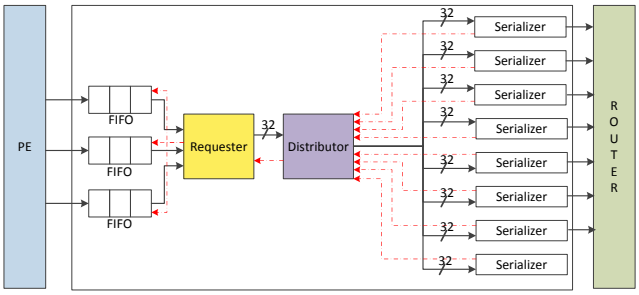

Fig. 5. New NI architecture

VI. Distributed Fault-Tolerant Approach

The data distributors of an SDM-based NI are the major contributor for area and power. We observe that distributors of the base design are not fully utilized all the time. Specifically, the distributors are idle when serialized data is sent over the link for the corresponding connection. In order to minimize the area and power of the NI, we propose an architectural change in the data distributors, so that a single distributor can serve all the connections from a PE. This helps to decouple the number of distributors from the number of connections to be established from a PE. In this section, we introduce the new design for the distributors and highlight the key architectural changes with respect to the base design in [17].

A. Operation

The operation of the distributor is based on request-acknowledge protocol with serializers requesting for data only when they finish sending the serialized packet to the network. This is shown in Figure 5 with red dashed lines indicating request flow and black solid lines indicating data flow. Once a connection is established, participating serializers request for data. These requests are forwarded to corresponding output fifos via the distributor and the requester. Upon getting the request, the output fifo sends data back to the requesting serializer. Moreover, we also split the distributor design to separate the distributor-fifo and the distributor-serializer interface. The new component is called the "requester". In order to maintain data correctness and ensuring correct data is sent to the correct output line, every component in the design is identified by uniquely assigned IDs, namely serializer ID (sID), distributor ID (dID), requester ID (rID) and the fifo ID (qID). At the start of operation, each serializer is assigned a fifo ID to indicate the fifo from which data needs to be fetched. The distributor polls for ready signal of each serializer once every clock. If it finds that a particular serializer requires data, it forwards the request to the requester along with control data – its own dID, the sID of the requesting serializer and corresponding qID. The distributor then moves on to poll the next serializer at the next clock cycle.

Requester logic is rather simple. Upon receiving a request from a distributor, the requester stores the incoming dID and sID bits and forwards the request to the correct fifo determined by incoming qID bits. When data comes back from the fifo, the requester pushes this data to the requesting distributor (known from the dID bits) along with the stored sID bits. Finally, the distributor sends this data to the requesting serializer determined by incoming sID bits. Once a request is polled

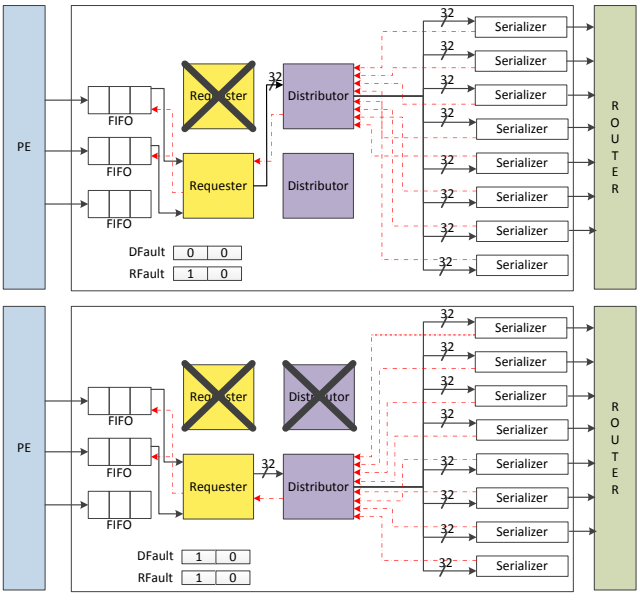

Fig. 6. Operation of distributed design

for data, it takes 4 clock cycles to propagate the request to a fifo and getting data to the serializer. Further, each serializer is polled every 8 cycles. Hence, every serializer can obtain data in at most 12 cycles. The serializer design is modified to include a buffer of one word giving us a tolerance of 33 cycles to process the data, or up to 29 outgoing wires without any impact in performance.

B. Throughput Saturation and Fault-Tolerance

As the number of links increases beyond 29, the throughput saturates. This is based on data packet size of 32 bits. In order to overcome this limitation and to add fault-tolerance, we can have multiple distributor-requester pairs; each pair can be dedicated to multiple connections. In an event of a fault in a distributor, one of the other distributors can take over the task of the faulty distributor albeit at a reduced throughput.

By separating the distributor-fifo and distributor-serializer interface, the distributed approach provides more flexibility. Specifically, in the base design and the centralized fault-tolerant approach, a fault in either the distributor-fifo interface or the distributor-serializer interface would render a distributor faulty. In the new design, a fault in the requester (distributor-fifo interface in the base design) can be bypassed using another requester. Thus, a finer control of fault-tolerance is provided.

Distributors and requesters in the new architecture can adapt to faults by runtime connection re-mapping. There are two registers in the design – RFault and DFault. The RFault is maintained by the distributors while the DFault is maintained by the requesters. In a fault-free scenario, all bits of both RFault and DFault are set to 0. Connection is carried out using a distributor-requester pair. When a fault occurs in the pair, say in the requester, the corresponding bit of the RFault is set to 1. The distributor then forwards the serializer data request to another functioning requester. On the other hand, if a distributor becomes faulty, the corresponding DFault bit is set to 1 and the requester sends the fifo data to a functioning distributor. Since the qID is embedded with the serializer

TABLE I
COMPARISON OF VARIOUS NI ARCHITECTURES

Design	Area without scan circuit (μm^2)	Area with scan circuit (μm^2)	Max Freq. (MHz)	Power (μW)
Base	17,678	20,614	750	275
TMR-based	31,034	31,034	750	449
Centralized	20,904	25,256	750	349
Distributed (1 pair)	16,146	18,932	917	294
Distributed (2 pairs)	19,936	23,678	917	351

request and sID in the return data, any distributor-requester combination can be involved in serving a serializer for a connection. Figure 6 shows how the data mapping changes at runtime in presence of faults. As can be seen, the distributed fault-tolerant NI can work as long as one distributor-requester combination is free from faults.

VII. EXPERIMENTS AND RESULTS

For the purpose of evaluation, we have considered SDM-based NoCs with 8 outgoing wires. The data packets from each PE are assumed to be of 32 bits. The base network interface design is generated using the on-line tool developed by the authors in [17]. For the purpose of comparison, we also developed a triple modular redundancy (TMR) based implementation of the NI[2]. Corresponding to each distributor in the base design there are two additional distributor with a majority voting circuit. We compare the area, power and performance of the base NI design with TMR-based NI and the two design techniques proposed in this paper.

Synopsys Design Compiler is used to synthesize the designs using 65nm technology libraries from UMC. The clock frequency used for synthesis is 100MHz, but the designs support higher frequency as discussed in a later section. We used Synopsys Power Compiler (PwC) to report the power based on Switching Activity Interface Format (SAIF) flow. Area numbers are reported post-synthesis.

A. Area Comparison

Each NI is designed to have two outgoing and incoming connections. Therefore, the base and the centralized designs have two distributors in each NI. The TMR-based implementation has two sets of 3-distributors. The new architecture only requires one distributor-requester pair for correct operation. However, for fault-tolerant operation distributed design with two distributor-requester pairs is considered throughout this section, unless mentioned otherwise.

Table I shows the area of the transmission side of the network interface for the five designs. The area overhead of the centralized and the TMR-based approach are approximately 18% and 75% respectively in comparison with base design. The new distributed architecture with one distributor-requester pair consumes 9%less area than the base-design, while the fault-tolerant distributed design with two pairs has 12% area overhead. Although it is expected that TMR based design should be three times the area of base design, we see only 75% increase in area since TMR is implemented only for the data

[2]Although, TMR-based systems are frequently used for tolerating transient faults, but can also be used for hardware faults [24]. Moreover, due to the non-existance of any fault-tolerant techniques for Network Interface for SDM-NoC, we use TMR-based design as a reference.

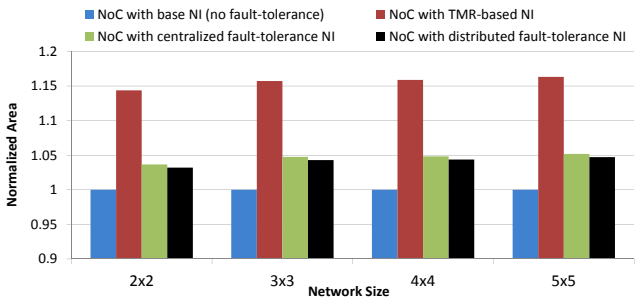

Fig. 7. Total area of NoC

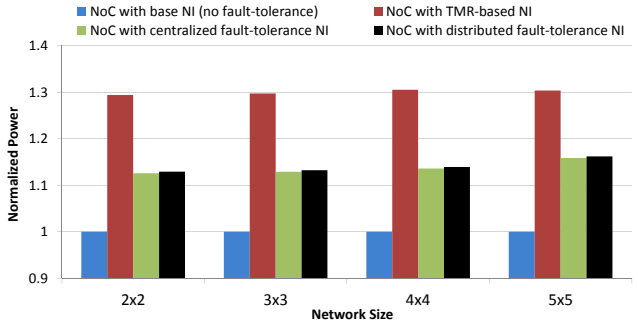

Fig. 8. Total Power of NoC

distributors of NI. The area overhead for the centralized fault-tolerant design comes from two major components – the new controller introduced and the modifications in the distributors to incorporate fault-tolerance. Prior research in the area of fault-tolerant NoC reports an area overhead of 42% for routers [15] and 75% for NI [16]. The designs introduced in this paper have less than 20% overhead due to fault-tolerance.

To analyze the impact of fault-tolerant NI for a complete SDM-based NoC, we plot the total area of NoC with increasing size of the network. This is shown in Figure 7 for the four different NI designs. The area overhead of NoCs with fault-tolerance is less than 5% of the total area of NoC without fault-tolerance for both centralized and distributed designs.

Table I also reports the area of network interface after implementing the fault detection circuit using scan chains. This results in an overhead of around 20% for both centralized and distributed designs. This is consistent with the numbers predicted in prior-art for fault-detection circuits [25], [26]. The maximum frequency of operation for the distributed design is more than the other three designs. For the distributed design, splitting the requester and distributor reduces the critical path delay. This helps in increasing the frequency of operation.

B. Power Comparison

Table I shows the power consumption of various NI designs in comparison with the base design. The numbers are reported for 100MHz operation assuming switching activity of 0.3. The TMR-based NI consumes 63% more power than the base design while the centralized approach consumes 27% more power. This power overhead in the centralized design is due to the added controller. The new architecture without fault-tolerance consumes 7% more power when compared to the base design while the fault-tolerant two-pair distributed approach consumes 28% more power. Prior research in the

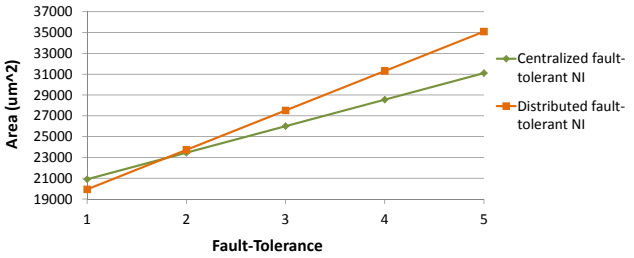

Fig. 9. Impact on gate-count with increasing fault-tolerance for NI

TABLE II
FAULT COVERAGE IN NETWORK INTERFACE USING VARIOUS TECHNIQUES

Fault Class	Base Design	TMR-based	Centralized	Distributed
Detected	190,543	219,724	200,773	199,583
Undetected	129	165	261	345
ATPG Untestable	198	216	223	160
Not Detected	24	33	355	16
Total Faults	190,894	220,138	201,612	200,104
Fault Coverage	99.82%	99.80%	99.58%	99.73%

field of fault-tolerance reports power overhead of around 50% for fault tolerance [16]. The power consumption of the entire NoC is shown in Figure 8 normalized to the power consumed in the base design. The average power overhead of the TMR-based design is 30% while about 13% overhead for the other two designs. The overhead increases marginally with increasing network size.

C. Impact of Increasing Fault-tolerance

We now provide the impact of increasing levels of fault-tolerance on area. For the centralized design, increasing fault-tolerance implies increasing the number of distributors and the size of DFault. For the distributed design, higher fault-tolerance is achieved by adding more distributor-requester pairs. Figure 9 shows the impact of increasing extra sets of distributors for the two designs, e.g. a 2-fault tolerant NI point in distributed design can tolerate faults in at most 2 distributors and 2 requesters i.e. 4 components in total. For the centralized design, a 2-fault tolerant point can tolerate at most 2 faults in only distributors. Any fault in the controller will render the NI useless. The increasing gap in area between the two designs with increasing fault-tolerant level can be explained by higher area of distributor-requester pair in the distributed design as compared to the distributor in the centralized design.

D. Fault Coverage

Synopsys DFT-Max is used to stitch scan chain for all four NI designs. The fault coverage numbers are reported in Table II. Faults in each category are comparable for all the designs except for the centralized one, where approximately 10 times more faults are not-detected. The majority of these not-detected faults are in the controller. Although the numbers could be improved using high analysis effort of the tool, the improvement is minor (approx 2%) as compared to the tool runtime (2x). Thus, there is a high probability of a fault in the controller not being detected using scan. The distributed design on the other hand, performs better and achieves higher fault coverage.

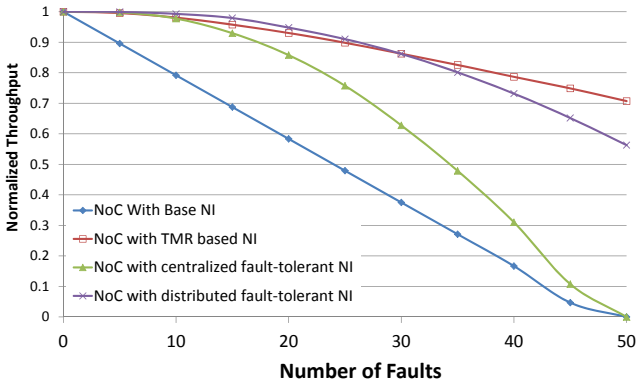

Fig. 10. Network performance with increasing faults

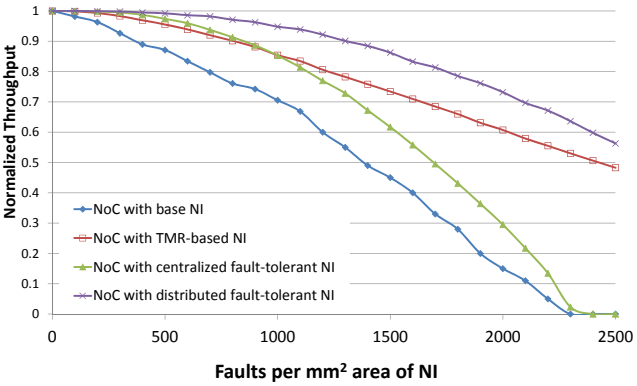

Fig. 11. Network performance with increasing fault density

E. Design Latency in Fault

The design latency is computed for both the designs under faulty conditions. The latency of centralized fault-tolerant architecture is calculated as follows. During a fault, the design takes 4 cycles to reconfigure. Priority is given for assigned distributor to first serve its own serializer before starting to service serializers of the faulty distributor. Hence, the latency is given by $L = 4 + i$ where i ranges from 0 to 7, and denotes the number of the serializers originally linked with the correct distributor. For the distributed design, the latency is $L = 4 + 2 \times n$, where n denotes the number of outgoing links. Thus, with 32 bits per word, the design can have at most 14 wires with no impact on performance.

F. Network Performance

In this section, we analyze network performance as the number of faults in the network increases. For evaluation of the different fault-tolerance scheme, we developed a model of a 4x4 NoC using Matlab. The base NI and the centralized fault-tolerant NI has three distributors each. The NI based on distributed approach has 3 distributor-requester pairs. Faults are randomly injected and results are normalized over 1000 runs. Figure 10 plots the normalized throughput of the NoC using base NI, TMR-based NI, the centralized and the distributed fault-tolerant NI.

Few observations can be made from the above plot. First of all, the throughput of the base design decreases linearly with the number of faults. Every fault in a distributor renders the outgoing connection infeasible thereby reducing the available

throughput. Secondly, all fault-tolerant designs have less than 10% reduction in performance for up to 15 faults. As more faults are injected, the performance degradation of centralized design is worse than for TMR-based design and the NoC with distributed NI. Both TMR-based and distributed NI based designs have redundant hardware which can tolerate multiple faults. For the distributed design, an NI breaks only when all distributors or requesters are faulty. In a way, each NI of the distributed design can tolerate upto 4 hardware faults. For the NI based on centralized design, a maximum of two faults can be tolerated. Finally, at around 50 faults, the NoC breaks completely for both the base design and the centralized fault-tolerant NI. The throughput-reduction for the TMR-based NI and the distributed NI is only 30% and 45% respectively.

We see that TMR and the distributed NI-based NoC perform better for a higher number of faults. However, both designs are based on redundant hardware. It is, therefore, not fair to compare the designs only on the basis of number of faults. To normalize the comparison of the fault-tolerant techniques, we plot the normalized throughput against fault-density in Figure 11. The fault density is defined as faults per unit area of the NI (number of faults per mm^2). We see that the performance of all fault-tolerant designs degrades gracefully as the fault density increases. This figure shows that the distributed design provides highest throughput for higher fault rates.

VIII. Conclusions And Discussions

In this paper, we presented two fault-tolerant architectures for NI which plays a crucial role in inter-PE communication. We evaluated both architectures with respect to area, power, latency and throughput. Our analysis shows that designers need to make careful considerations to select ideal design technique for their architecture. First of all, the distributed design is more area efficient and provides better throughput than the centralized version. As more distributors are added to the system for higher fault-tolerance, the centralized design becomes more efficient. Secondly, the distributed design offers the advantage that any fault at a single place can be easily tolerated with no performance penalty, while for the centralized design, a fault in the controller renders the NI useless. Finally, from the scalability perspective, by decoupling the outgoing connections with the distributors, the distributed design becomes more scalable with increase in the number of connections. A tool has been developed to automatically generate fault-tolerant NIs for various configuration parameters and is made available for use on-line for the benefit of research community.

There are still a few open areas where some more improvements are possible. Currently, in both approaches, link faults are handled at the application level. As a future work, fault-tolerant design of links can be looked into. Another potential area could be the design of an efficient load balancing algorithm and the consideration of constant failure rate models. Both designs discussed here picks up one of the other functional distributors if a fault is detected in any distributor. This is independent of how many outgoing wires the new distributor is already serving. A more granular load-balancing technique can be looked at in the future.

References

[1] M. Arden *et al.*, "The International Technology Roadmap for Semiconductors–Perspectives and challenges for the next 15 years* 1," *Current Opinion in Solid State and Materials Science*, 2002.

[2] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain *et al.*, "An 80-tile sub-100-w teraflops processor in 65-nm cmos," *IEEE Journal of Solid-State Circuits*, 2008.

[3] L. Benini and G. D. Micheli, "Networks on Chips: A New SoC Paradigm," *IEEE Computer*, 2002.

[4] K. Goossens, J. Dielissen, and A. Radulescu, "Æthereal network on chip: concepts, architectures, and implementations," *IEEE Design & Test of Computers*, 2005.

[5] A. Leroy, D. Milojevic, D. Verkest, F. Robert, and F. Catthoor, "Concepts and Implementation of Spatial Division Multiplexing for Guaranteed Throughput in Networks-on-Chip," *IEEE Transactions on Computers*, 2008.

[6] C. Constantinescu, "Trends and challenges in VLSI circuit reliability," *IEEE Micro*, 2003.

[7] M. Pirretti, G. Link, R. Brooks, N. Vijaykrishnan, M. Kandemir, and M. Irwin, "Fault tolerant algorithms for network-on-chip interconnect," in *IEEE Annual Symposium on VLSI*, 2004.

[8] L. Ni and P. McKinley, "A survey of wormhole routing techniques in direct networks," *IEEE Computer*, 1993.

[9] M. Ali, M. Welzl, and S. Hellebrand, "A dynamic routing mechanism for network on chip," in *IEEE NORCHIP*, 2005.

[10] F. Refan, H. Alemzadeh, S. Safari, P. Prinetto, and Z. Navabi, "Reliability in application specific mesh-based NoC architectures," in *IEEE International On-Line Testing Symposium (IOLTS)*, 2008.

[11] C. Concatto, D. Matos, L. Carro, F. Kastensmidt, A. Susin, E. Cota, and M. Kreutz, "Fault tolerant mechanism to improve yield in NoCs using a reconfigurable router," in *ACM Symposium on Integrated Circuits and System Design (SBCCI)*, 2009.

[12] V. Rantala, T. Lehtonen, P. Liljeberg, and J. Plosila, "Multi network interface architectures for fault tolerant Network-on-Chip," in *IEEE International Symposium on Signals, Circuits and Systems (ISSCS)*.

[13] F. Koupaei, A. Khademzadeh, and M. Janidarmian, "Fault-Tolerant Application-Specific Network-on-Chip," in *Proceedings of the World Congress on Engineering and Computer Science*, 2011.

[14] Y. Chang, C. Chiu, S. Lin, and C. Liu, "On the design and analysis of fault tolerant NoC architecture using spare routers," in *IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)*, 2011.

[15] D. Fick, A. DeOrio, J. Hu, V. Bertacco, D. Blaauw, and D. Sylvester, "Vicis: a reliable network for unreliable silicon," in *ACM/IEEE Design Automation Conference (DAC)*, 2009.

[16] L. Fiorin, L. Micconi, and M. Sami, "Design of Fault Tolerant Network Interfaces for NoCs," in *IEEE Euromicro Conference on Digital System Design (DSD)*, 2011.

[17] Z. Yang, A. Kumar, and Y. Ha, "An Area-efficient Dynamically Reconfigurable Spatial Division Multiplexing Network-on-Chip with Static Throughput Guarantee," in *IEEE Field Programmable Technology (FPT)*, 2010.

[18] S. Furber, "Living with failure: lessons from nature?" in *IEEE European Test Symposium (ETS)*, 2006.

[19] Murali, S. and Theocharides, T. and Vijaykrishnan, N. and Irwin, M.J. and Benini, L. and De Micheli, G., "Analysis of error recovery schemes for networks on chips," *IEEE Design & Test of Computers*, 2005.

[20] A. Amory, E. Brião, É. Cota, M. Lubaszewski, and F. Moraes, "A scalable test strategy for network-on-chip routers," in *IEEE International Test Conference (ITC)*, 2005.

[21] C. Aktouf, "A complete strategy for testing an on-chip multiprocessor architecture," *IEEE Design & Test of Computers*, 2002.

[22] I. Koren and C. Krishna, *Fault-tolerant systems*. Morgan Kaufmann, 2007.

[23] C. Grecu, L. Anghel, P. Pande, A. Ivanov, and R. Saleh, "Essential fault-tolerance metrics for NoC infrastructures," in *IEEE International On-Line Testing Symposium (IOLTS)*, 2007.

[24] J. Vial, A. Bosio, P. Girard, C. Landrault, S. Pravossoudovitch, and A. Virazel, "Using tmr architectures for yield improvement," in *IEEE Defect and Fault Tolerance of VLSI Systems (DFTVS)*, 2008.

[25] I. Lee, M. Basoglu, M. Sullivan, D. Yoon, L. Kaplan, and M. Erez, "Survey of Error and Fault Detection Mechanisms," University of Texas at Austin, Tech. Rep., April 2011.

[26] A. Narsale and M. Huang, "Variation-tolerant hierarchical voltage monitoring circuit for soft error detection," in *IEEE Quality of Electronic Design (ISQED)*, 2009.