Transient and Permanent Error Control for High-End Multiprocessor Systems-on-Chip

Qiaoyan Yu[1], José Cano[2], José Flich[2], Paul Ampadu[3]

[1]Department of Electrical and Computer Engineering
University of New Hampshire
Durham, NH, USA 03824
qiaoyan.yu@unh.edu

[2]Parallel Architectures Group
Department of Computer Engineering
Universitat Politècnica de València
46022 Valencia, Spain
jocare@gap.upv.es, jflich@disca.upv.es

[3]Department of Electrical and Computer Engineering
University of Rochester
Rochester, NY, USA 14627
paul.ampadu@rochester.edu

Abstract— **High-end MPSoC systems with built-in high-radix topologies achieve good performance because of the improved connectivity and the reduced network diameter. In high-end MPSoC systems, fault tolerance support is becoming a compulsory feature. In this work, we propose a combined method to address permanent and transient link and router failures in those systems. The LBDRhr mechanism is proposed to tolerate permanent link failures in some popular high-radix topologies. The increased router complexity may lead to more transient router errors than routers using simple XY routing algorithm. We exploit the inherent information redundancy (IIR) in LBDRhr logic to manage transient errors in the network routers. Thorough analyses are provided to discover the appropriate internal nodes and the forbidden signal patterns for transient error detection. Simulation results show that LBDRhr logic can tolerate all of the permanent failure combinations of long-range links and 80% of links failures at short-range links. Case studies show that the error detection method based on the new IIR extraction method reduces the power consumption and the residual error rate by 33% and up to two orders of magnitude, respectively, compared to triple modular redundancy. The impact of network topologies on the efficiency of the detection mechanism has been examined in this work, as well.**

Keywords-Networks-on-chip; transient error; permanent error; information redundancy; arbiter; reliability; fault tolerant

I. Introduction

Current multi-processor systems-on-chip (MPSoCs) are usually formed by tens of microprocessors and other components (such as memories) in the same chip. As technology advances, more and more components are included in MPSoCs; hundreds of components are expected to integrate into a single chip in the near future. One side effect of increasing component densities is the need of long interconnections, which creates a well-known communication bottleneck. Networks-on-Chip (NoCs) have demonstrated potential to manage this communication problem [1, 2].

In MPSoCs, it is worth differentiating between application-specific MPSoCs and high-end MPSoCs. In the former (e.g. Spidergon STNoC [3]), fully irregular topologies are derived as the system design is totally customized for the applications. In contrast, in high-end MPSoCs (e.g. Tilera [4]), regular structures (2D mesh-based topologies) are used. In this paper we focus on high-end MPSoCs.

Reliability is one of the most critical emerging challenges, caused by ever increasing chip densities [5]. Nanoscale fabrication processes inevitably result in defective components, which lead to permanent errors. Without a proper solution, a simple failure may render the chip useless, impacting the yield and cost of manufacturing. One common approach to handle permanent errors in NoCs is to use fault-tolerant routing algorithms, which assume that the permanently unusable links and/or routers are isolated at the testing stage [6]. One recent solution that offers compact routing implementation and high fault tolerance is the Logic-Based Distributed Routing (LBDR) approach [7]. Transient errors are becoming equally important. As the critical charge of a capacitive node decreases with technology scaling, the probability that a high-energy particle strike flips the value in a storage node increases [8]. Moreover, transient error rates in logic gates are expected to increase because of higher frequencies and lower supply voltages [9]. Recently, inherent information redundancy (IIR) has been used to manage transient errors in a NoC based on XY routing [10].

These two approaches combined (LBDR and IIR) can thus be envisioned as a good solution for addressing both permanent and transient errors in NoCs. However, as they were originally proposed, both were not extensively applied to the topologies and configurations for realistic applications. On one hand, the LBDR approach is designed for 2D meshes where routers are connected to one neighbor on each dimension and direction. On the other hand, because the uniqueness of XY routing used in [10], that approach is not suitable for the router designs with more advanced routing solutions. Indeed, recently, different topologies have been presented. 2-D meshes suffer from an increasing diameter and a limited bisection bandwidth as the system size increases. To address these issues, the new topologies add extra links connecting routers in the 2D mesh, thus providing additional connectivity and lower diameters. Three advanced topologies are shown in Fig. 1. The initial 2D mesh is the underlying topology.

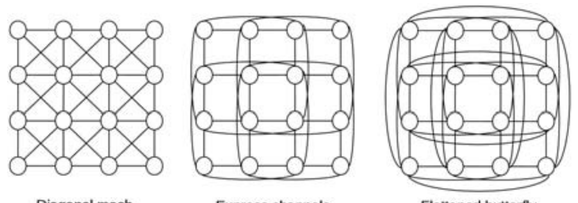


Fig. 1 Advanced solutions for 2D-meshes.

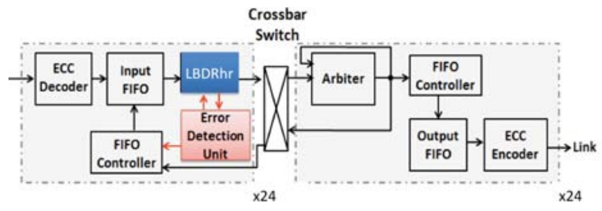


Fig. 2 Block diagram of proposed router.

In this work, we address fault tolerance for advanced topologies by redesigning the LBDR mechanism. The proposed LBDRhr (LBDR for high-radix networks) contains two virtual channels and includes an adaptive routing algorithm. We prove that LBDRhr is deadlock-free for any of the high-radix topologies even when certain links and/or routers permanently fail. We further couple the LBDRhr mechanism with a new error control method to detect the transient errors in LBDRhr logic. This new error control method exploits the inherent information redundancy in LBDRhr to significantly reduce the error control overhead. The overview of the proposed router is shown in Fig. 2.

The remainder of this paper is organized as follows. The related work is discussed in Section II. In Section III we describe the new routing mechanism LBDRhr. The new method to extract inherent information redundancy in the LBDRhr approach is proposed in Section IV. Performance, area and power consumption are evaluated in Section V. Conclusions are provided in Section VI.

II. Related Work

A. *Transient Error Detection for NoC Router*

Because of its temporary behavior, transient link errors can be managed with various error control coding methods—such as forward error correction (FEC), error detection combined with automatic repeat request (ARQ) retransmission, and hybrid ARQ (HARQ). ARQ and HARQ are not suitable for the route computation unit because no retransmission is allowed after the packet arrives in input FIFOs. Transient errors in NoC links can be managed using error control coding (ECC) [11-13]. FEC methods are also used in the data path of NoC routers [14, 15]. Unfortunately, FEC techniques are only suitable for linear systems. In a NoC router, the control path (in charge of the routing of packets and the allocation of resources to packets) is not a linear system. Triple-modular redundancy (TMR) duplicating the unit under protection and selecting the output through majority voting is attractive for its simplicity and it has been applied to the router control path [16, 17]. Theoretically, TMR functions correctly when up to one-third of the received copies are wrong. Furthermore, the potential errors in the majority voter further reduce the effectiveness of the TMR approach, particularly when the number of units being protected is small [18]. In [10], the inherent information redundancy in the router using XY routing algorithm is investigated to reduce the packet loss and misrouting without significantly sacrificing power and area overhead. It has been shown that the residual error rate achieved by the inherent information redundancy based error control method is smaller than that of TMR.

B. *Permanent Error Management*

Permanent fault support in NoCs can be achieved by means of three main approaches: fault-tolerant routing algorithms, reconfiguration techniques and component redundancy. Routing algorithms for faulty NoCs have been extensively investigated to support irregular topologies derived from regular ones after permanent link failures. Some solutions [19, 20] are implemented using routing tables, suffering the scalability problems and routing costs derived from them. On the contrary, other solutions are logic-based. FDOR [21] is based on the idea of dividing an irregular mesh topology into regular sub-meshes. Although a large variety of irregular mesh topologies are supported, it does not offer full coverage. The uLBDR approach [7], however, is proposed as an efficient logic-based mechanism offering 100% coverage to faulty 2D-meshes. The routing information is condensed into a small set of bits distributed over the switches. Reconfiguration techniques are focused on adequately updating the routing tables whenever failures occur. ARIADNE [22] is a distributed reconfiguration solution for agnostic NoCs capable of circumventing large numbers of simultaneous faults. Compared with other solutions [23, 24], it guarantees higher performance during normal operation since no virtual channels are restricted to deterministic routing. Component redundancy is the easiest way for providing fault tolerance, but needs to consider the extra cost derived from redundant components.

To the best of our knowledge, there are no solutions that are able to allow distributed routing implementation in faulty high-radix topologies with no routing tables and minimum logic. The LBDRhr approach we proposed in this work replaces the routing tables with a small and bounded set of bits that are finally hardwired, offering support for a great number of irregular shapes derived from high-radix topologies. We also exploit the inherent information redundancy in LBDRhr logic to manage transient errors in high-end MPSoC systems.

III. LBDRhr Mechanism for High-Radix Topologies

A. *LBDRhr Description*

In this section we describe the LBDRhr mechanism, an extension of LBDRx [25], which is able to tolerate permanent link and router failures. We assume that links are *bidirectional* in this work. As shown in Fig. 3, this method is implemented with three basic logic blocks, each addressing different port types: 3-hop, 2-hop and 1-hop ports. 3-hop ports are those connected to long-range links that connect two routers with a distance of 3-hops (through the basic 2D-mesh). 2-hop ports are connected to links that connect to routers at 2-hop distance. 1-hop ports are connected to 2D-mesh links.

LBDRhr uses a few configuration bits to *store local information about the neighboring links*: 8 configuration bits for routing purposes (Rne, Rnw, Ren, Res, Rwn, Rws, Rse, and Rsw); 2 bits (DR1 and DR0) for two deroute options (special cases) at every input port; and one connectivity bit per output port (Cx for port X), which can be hardwired. With this compact information per router, LBDRhr addresses fault tolerance without using routing tables, achieving important savings in critical aspects such as area, latency and power.

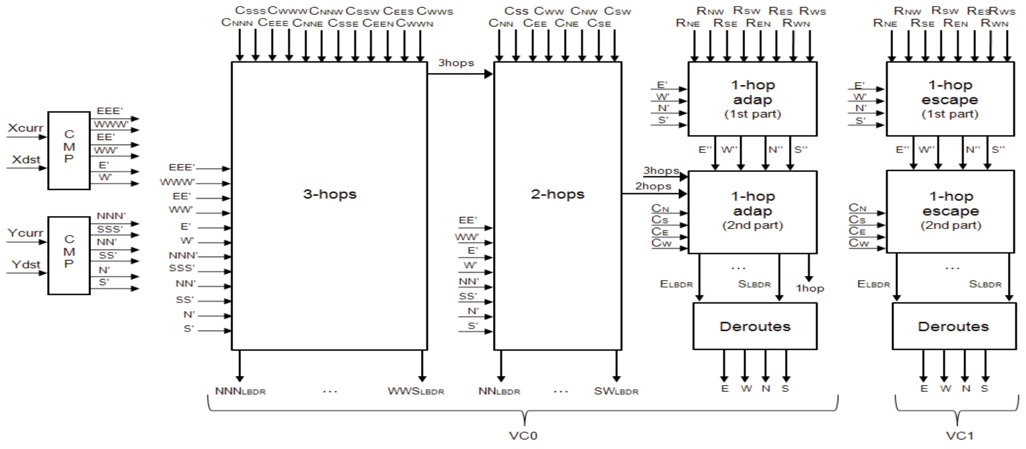


Fig. 3 Diagram of LBDRhr

First of all, the logic computes the relative position of the message's destination. Note that packets forwarded to the local port are excluded from the routing logic. With this logic, the directions to take for the packet are computed. Signals NNN', SSS', EEE' and WWW' indicate whether the destination is at least 3 hops in the give direction. Similarly, the signals of 2 hops and 1 hop are computed.

In the first logic block, if the packet's destination is at least three hops along the same direction away from the current router, then the corresponding 3-hop port is selected. A simple AND gate per 3-hop output port is needed to AND both the proper comparator signal and the connectivity bits (the presence) of the 3-hop port (e.g., NNN' signal and the Cnnn bit). Notice that the logic may select more than one output port as a candidate for routing. For example, in a NoC with the flattened butterfly topology, a packet at router 0 (located in the upper left corner) with destination at router 15 (located in the lower right corner) can use both EEE and SSS output ports.

The second logic block deals with 2-hop ports. In this case, 3-hop ports have priority over 2-hop ports. Therefore, the 3-hop signal is considered in this block (the 3-hop signal is easily computed by ORing all the output signals from the 3-hop logic block). The 2-hop block behaves similarly to the previous one: 2-hop ports are selected if the destination is at least two hops away along the intended direction. Directions that are covered in this block are NN, EE, WW, and SS, for the mesh with express channels, and NE, ES, WS, and WN, for the diagonal mesh topology. A simple 4-entry AND gate is required for every output port (ANDing the 3-hop signal, the availability of the output port, and the combined outputs of the comparators). Notice that 3-hop ports and 2-hop ports will never be selected at the same time for the same packet.

Finally, the third logic block deals with 1-hop ports. In this case, complexity is higher. The logic is replicated in two blocks. The first block (*1-hop-adap*) is filtered by the 3-hop and 2-hop signals. Thus, selecting 3-hop or 2-hop ports prevents using 1-hop ports. The second block (*1-hop-escape*), however, is not filtered out by the previous blocks and works independently. The 1-hop-adap and 1-hop-escape blocks consider both routing and deroute bits in the same manner they are configured and used in [7]. Notice that these bits are only used in the underlying 2D-mesh to provide non-minimal path support. In particular, the logic forces messages to take a non-minimal port whenever the second part of the logic fails in routing the message.

B. *Deadlock-free Routing*

With the provided logic and with no further support, deadlocks can easily form. In order to avoid deadlocks, and still keeping the LBDRhr approach simple, we combine the mechanism with a routing algorithm using two virtual channels. The first virtual channel (VC0) is used to route packets freely using minimal paths through long-range links (3-hop and 2-hop ports). Also, packets can be routed through 1-hop links but taking into account both routing bits and deroute bits. The second virtual channel, however, can be used only for routing packets through 1-hop links and by taking into account routing and deroute bits. Therefore, the underlying 2D-mesh topology must provide a deadlock-free routing implementation.

The arbiter needs also to be codesigned with LBDRhr. Notice that for some packets different routing options may be available at the same time: some output ports through VC0 and some outputs through VC1. To avoid deadlocks, the arbiter must filter out those routing options that could allow packets moving from VC1 to VC0 again. That is, once a packet moves to VC1 it will stay in VC1 along its route until it reaches its destination. Therefore, VC1 implements an escape path through the underlying 2D-mesh topology. When the packet is at VC0, adaptive and escape options are available because of the LBDRhr logic. The arbiter should give higher priority to adaptive output ports. In the case the adaptive output port is busy, the escape output port must be selected. Finally, packets will be injected always through VC0 (to maximize the flexibility).

In order to clarify how the arbiter and the virtual channels are used, an example is shown in Fig. 4. The right part of the figure represents VC1 (the escape layer) through a 2D-mesh topology with some faulty links. On the other hand, the left part represents VC0 (escape and adaptive layers) through the previous faulty 2D-mesh topology along with express channels, where there also exist faulty express channels.

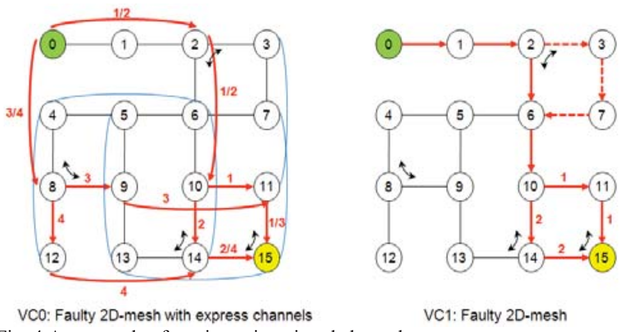


Fig. 4 An example of routing using virtual channels.

As an example we route a packet from router 0 to router 15. At a first sight, we observe that there exist several options in both layers for reaching our destination.

On VC1, four escape paths are possible. From router 0 to router 2 there is only one valid option, but at router 2 two options are possible (through router 6 and through routers 3, 7 and 6). At router 10, there are two new options again (through router 11 and through router 14). By combining these options, up to four paths are possible. It is important to note that, when router 7 is reached (through router 3), there is no minimal path available for reaching router 15 because there is a faulty link between routers 7 and 11. Therefore, at Router 7, a deroute option is needed (in this case going West to router 6). At router 6, it is possible to reach the destination without using deroutes (through a minimal path) with the options previously commented. Note that the sub-path using the deroute option is marked with dashed line. Note also that routing in VC1 is deadlock-free based on LBDR.

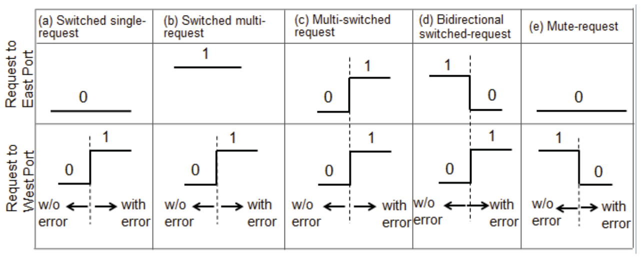
On VC0, the previous four escape paths and new four adaptive paths are available. Concerning the adaptive paths, the first two options share the path until router 10, but in this case using 2-hop links (express channels). Options 3 and 4 share the path until router 8 using a 2-hop link. Therefore, in this example the arbiter will select between the eight options available through VC0. Remember that options with longer links have priority over shorter ones, so at the end, the arbiter will choose between the four adaptive paths available. Note that a change between VC0 and VC1 directly depends on the dynamic conditions of the network (explained previously) that does not affect the conclusions drew from our example.

IV. Proposed IIR-based Transient Error Detection Method for LBDRhr Logic

A. *Inherent Information Redundancy (IIR) Extraction*

In this work, the *forbidden signal patterns* in routers are regarded as inherent information redundancy. LBDRhr logic determines the next-hop on the path for the packet at the current node. If one or more logic gates in the LBDRhr logic fail because of transient errors, the packet may be routed to wrong directions, thus causing either additional latency or deadlocks. For a router using XY routing, the route path determination unit is simple: only one output request is valid per each packet. As a result, detecting the number of valid output requests can identify transient errors in the CMP unit.

TABLE 1. Five request failures



	(a) Switched single-request	(b) Switched multi-request	(c) Multi-switched request	(d) Bidirectional switched-request	(e) Mute-request
Request to East Port	0	1	0 1	1 0	0
Request to West Port	0 1 w/o error ← → with error	0 1 w/o error ← → with error	0 1 w/o error ← → with error	0 1 w/o error ← → with error	1 0 w/o error ← → with error

Sigma-and-branch error detection [10] has been successfully applied to routers using the XY routing algorithm.

To facilitate fault-tolerant routing, LBDRhr logic may generate multiple valid requests to use the output ports. Consequently, the method in [10] is not feasible. However, the principle of inherent information redundancy extraction is applicable to the LBDRhr-based routers. Different with [10], we classify the forbidden signal patterns existing in the output port requests into four general categories:

- *Mute-Request:* None of the output ports is requested when a packet header flit arrives at the input port. As shown in Table 1-case (e), the valid request (high logic value) is muted by logic gate failures.
- *Multiple-Request:* One or more non-valid requests are flipped to be 'valid' requests, resulting in accessing two opposite directions, as shown in Table 1-cases (b) and (c). Those are illegal signal combinations, which can be used to detect the malfunction of the LBDRhr logic.
- *Switched-Single-Request:* A non-header flit arrives but there is a request to build a input port-output port connection. Or, the destination and the current nodes have the same horizontal or vertical coordinate, nevertheless the next-hop route deviates from horizontal or vertical direction. As shown in Table 1-case (a), the transient error in LBDRhr logic causes an non-reasonable request to access the output port.
- *Bidirectional-Switched-Request:* The request to the intended output port is muted while another opposite output port is requested, as shown in Table 1-case(d).

The summary of Table 1 is based on the routing phenomena that two opposite directions cannot be enabled at the same time. Without losing generality, we use the pair of east and west ports to introduce the error detection approach based on the different error scenarios shown in Table 1.

B. *IIR-Based Error Detection for LBDRhr Logic*

We propose to detect transient errors in each function unit. As shown in Fig. 5, a CMP error detection unit is used to detect the forbidden signal patterns in the coordinator comparison unit; the multi-hop logic (MHL) error detection unit is used to examine the transient errors in the 24 output port request; the deroute error detection unit is used to identify the wrong deroute path. Any of the detected errors invokes to re-compute the direction for the next hop. The details for these three error detection units are discussed in the following subsections.

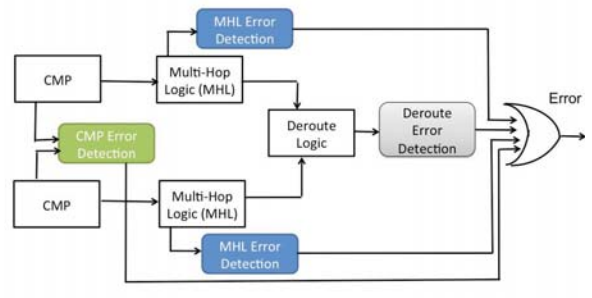


Fig. 5 Overview of proposed error detection method for LBDRhr logic.

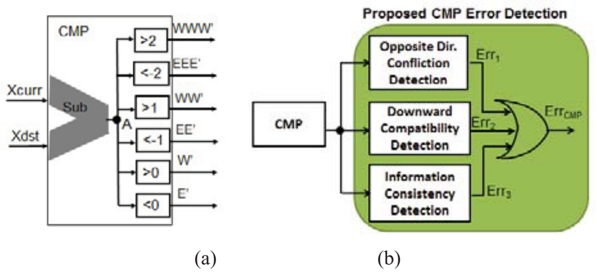

CMP
Xcurr
Xdst
Sub
A
>2
<-2
>1
<-1
>0
<0
WWW'
EEE'
WW'
EE'
W'
E'
Proposed CMP Error Detection
CMP
Opposite Dir. Confliction Detection
Downward Compatibility Detection
Information Consistency Detection
Err1
Err2
Err3
ErrCMP
(a)
(b)

Fig. 6 Coordinate comparison unit (a) without error detection and (b) with the proposed error detection.

1) *Error Detection for CMP Unit*

The coordinate comparison module (CMP) is implemented with a subtraction unit and six threshold comparison units (>2, <-2, >1, <-1, >0 and <0), as shown in Fig. 6(a). *Xcurr* and *Xdest* are the x coordinates for the current and destination nodes, respectively. This CMP provides the prime signals for 3-hops, 2-hops and 1-hop units in the LBDRhr logic (shown in Fig. 3). Similar to the CMP for XY routing, the opposite directions cannot be enabled at the same time. As a result, multiple-request error detection is still applicable in LBDRhr. We use the expressions (1)-(3) to detect multiple-requests.

$$Err_{1.1} = WWW' \;\&\; EEE' \tag{1}$$

$$Err_{1.2} = WW' \;\&\; EE' \tag{2}$$

$$Err_{1.3} = W' \;\&\; E' \tag{3}$$

The second inherent information redundancy in the three-stage LBDRhr logic is the information downward compatibility. If the subtraction output of *Xcurr* and *Xdest* is greater than 2 (i.e. *WWW'*=1), *WW'* and *W'* should be true, as well; otherwise, it indicates that the threshold comparison units have errors. Meanwhile, we keep the forbidden signal patterns of the opposite directions in mind to further constrain the signal patterns among *E'*, *W'*, *EE'*, *WW'*, *EEE'* and *WWW'*, as expressed in (4)-(7):

$$Err_{2.1} = WWW' \;\&\; !(WW' \;\&\; W' \;\&\; EE'\_n \;\&\; E'\_n) \tag{4}$$

$$Err_{2.2} = EEE' \;\&\; !(EE' \& E' \;\&\; WW'\_n \;\&\; W'\_n) \tag{5}$$

$$Err_{2.3} = WW' \;\&\; !(W' \;\&\; E'\_n) \tag{6}$$

$$Err_{2.4} = EE' \;\&\; !\,(E' \;\&\; W'\_n) \tag{7}$$

in which, *EE'_n*, *E'*_n, *WW'_n* and *W'_n* are the inverted values of *EE'*, *E'*, *WW'* and *W'*, respectively. With expressions (1)-(7), we can successfully identify the multiple requests, rather than the switched single-request, bidirectional switched-request and mute-request.

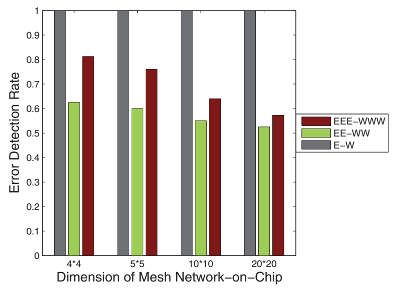


Fig. 7 Error detection rate breakdown for each signal combination in CMP

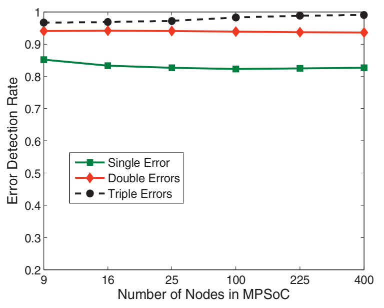


Fig. 8 Error detection rate of the proposed CMP.

Fortunately, we discovered the third type inherent information redundancy to detect the rest of request failure: the **information consistency** between the internal node *A* (in Fig. 6) and the CMP outputs. If the subtraction is not zero, *E'* and *W'* cannot both be zero because the value of *A* is either larger than zero or less than zero. When the subtraction is zero, neither *E'* nor *W'* can be zero because of the nature of '>2', '<-2', '>1', '<-1', '>0' and '<0' comparison units.

$$Err_3 = (A\_n \;\&\; (E' \mid W')) \;\mid\; (A \;\&\; E'\_n \;\&\; W'\_n) \tag{8}$$

The three IIR-based error detection outputs are ORed to obtain the overall error detection decision, Err_{CMP}, as shown in Fig. 6(b). We verify the error detection coverage of the error detection logic by flipping logic values of some gates in CMP unit. The error detection rate is the ratio of the number of error-detectable cases over the number of the total error injection cases. As shown in Fig. 7, no matter how the NoC size changes, the error detection rate for *E'* and *W'* is 100% because of the use of the internal node. Since the zero/non-zero subtraction output does not contribute to detect the errors causing wrong *EE'*, *WW'*, *EEE'* and *WWW'*, only the occurrence of opposite direction pairs helps to detect errors in *EE'*, *WW'*, *EEE'* and *WWW'*. Therefore, the error detection rate for *EE'*, *WW'*, *EEE'* and *WWW'* is less than 1. A larger NoC size is more likely to obtain a nonzero subtraction output (i.e. *A*!=0). As a result, the proposed method achieves a higher error detection rate in smaller NoCs, as shown in Fig. 7.

We exhaustively examined the impact of the number of failure gates on the CMP error detection rate. As shown in Fig. 8, the proposed method can successfully captures most of the single errors. As the number of failure gates increases, more errors occur on EE'-WW' and EEE'-WWW'; consequently, the error detection rate slightly decreases.

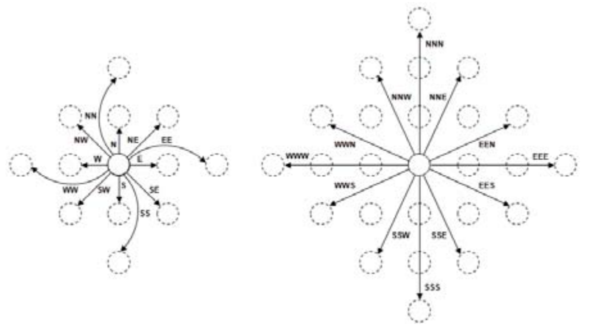


Fig. 9 Possible directions in LBDRhr (a) 1- and 2-hop (b) 3-hop directions

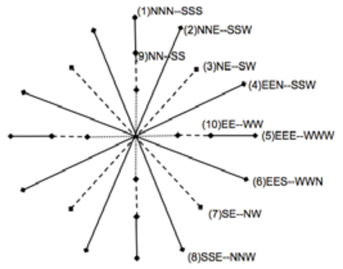


Fig. 10 Pairs of the forbidden concurrent routing directions. The center point is where the packet is currently located. Dash line and dot line are overlapped with solid line on the N/S/W/E directions.

2) *Error Detection for Hop Logic*

The LBDRhr is designed to facilitate the multi-hop routing, as shown in Fig. 9. In the XY routing, we found only two pairs of opposite directions; however, in the LBDRhr routing logic, we found ten pairs of opposite directions, each one on the 180-degree line shown in Fig. 10. Although adaptive routing algorithms can provide multiple routing paths, two directions opposite to each other are not likely offered at the same time. Consequently, we obtain *three* new inherent information redundancies in the multi-hop logic (MHL):

The opposite direction pairs (1)-(8) are eight *strong* opposite cases. For example, in the pair of *NNE-SSW*, N is opposite to S and E is opposite to W, therefore these two constraints further ensure the impossible occurrence of *NNE*=1 and *SSW*=1.

i. As shown in Fig. 3, *NN'*, *SS'*, *WW'* and *EE'* are generated with the use of the 3-hop signals. As a result, pair (1) should inherently *match* to pair (9); similarly, pair (5) should inherently *match* to pair (10) in Fig. 10.
ii. Other than the 180 degree line, the symmetric directions belonging to different quadrants are also exclusive, because part of the logic is exclusive. For example, *NNE* and SSE cannot be true as the same time. To enable *NNE*, the 2-hop north direction (i.e. *NN'*) has to be true, which means 2-hop south direction (i.e. *SS'*) is not enabled. As a result, *NNE* and *SSE* cannot be selected in the same router port. Consequently, we have more forbidden signal patterns to detect errors.

The inherent information redundancy (i) inspires us to use the expression (9) to detect errors in the 3-hops logic.

$$Err_{3\text{-}hops} = (NNE \,\&\, SSW) \mid (EEN \,\&\, SSW) \mid (EES \,\&\, WWN) \mid (SSE \,\&\, NNW) \mid (NNN \,\&\, SSS) \mid (EEE \,\&\, WWW) \mid (NNE \,\&\, NNW) \mid (SSW \,\&\, SSE) \mid (EEN \,\&\, EES) \mid (EES \,\&\, EEW) \mid (WWN \,\&\, WWS) \qquad (9)$$

However, this logic can only detect the errors for the multiple request scenarios. As shown in Fig. 3, the connection vector *Cdest* also plays an important role to determine the possible route for the packet's next hop. Given a network topology, we can eliminate some 3-hops paths. With this prior knowledge, we further detect the errors in the 3-hops logic. For the 4x4 diagonal and mesh with express lanes topologies, all outputs of the 3-hops logic are zero. This simplifies the error detection and improves the error detection rate.

The inherent information redundancy (ii) and (iii) inspire us to use the expression (10) to detect errors in the 2-hops unit. Again, we also use the prior-knowledge of *Cdest* to detect switched-request and mute-request.

$$Err_{2\text{-}hops} = (NN \,\&\, SS) \mid (EE \,\&\, WW) \mid (NE \,\&\, SW) \mid (SE \,\&\, NW) \mid (NE \,\&\, SE) \mid (SW \,\&\, SE) \qquad (10)$$

Whenever any of the previous error detection is true, it will invoke to re-compute the output port request. Fig. 11 shows the error detection rate of the proposed method is above 0.8, which maintains as the number of injected error increases. We also examine the impact of different topologies on the error detection rate. As shown in Fig. 11, the proposed method has minor variation on the error detection rate. This is because an all-zero Cdest vector simplifies the error detection circuit and thus reduces the probability of detection circuit failure. The error detection efficiency for the 2-hops is slightly less than that for 3-hops logic. The interesting phenomena we observed was that the proposed method improves the error detection rate of 2-hops logic as the number of error injected to the logic increases. A similar approach is applied to detect the error in 1-hop logic. Because of the sufficient IIR, we achieve the error detection rate of 1 in the 1-hop logic unit.

3) *Error Detection for Deroute Logic*

The deroute logic provides a non-minimal path in case the minimal path is not available, because of permanent errors. Based on the DR vector, the alternative paths are constructed with regular logic. For example, the $N_{deroute}$, $S_{deroute}$, $E_{deroute}$ and $W_{deroute}$ are expressed in (11)-(14).

$$N_{deroute} = \sim DR[0] \,\&\sim DR[1] \qquad (11)$$

$$E_{deroute} = \sim DR[0] \,\&\, DR[1] \qquad (12)$$

$$W_{deroute} = DR[0] \,\&\sim DR[1] \qquad (13)$$

$$S_{deroute} = DR[0] \,\&\, DR[1] \qquad (14)$$

The fact that the four directions are exclusive is regarded as a new inherent information redundancy to detect errors in the deroute logic. The simulation performed also considers the impact of failures in the error detection circuit. As shown in Fig. 13, the proposed error detection method can even tolerate the errors in the error detection circuit, as long as the number of errors injected is no more than four. Fig. 13 also shows that the decrease of the error detection rate is negligible.

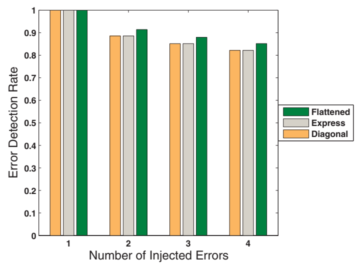


Fig. 11. 3-hops logic error detection rate.

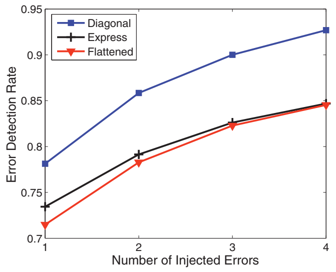


Fig. 12. 2-hops logic error detection rate.

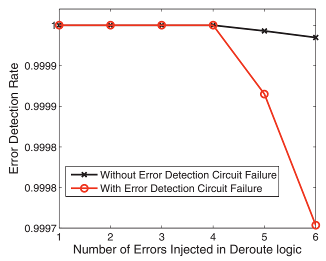


Fig. 13. Error detection rate of the deroute logic.

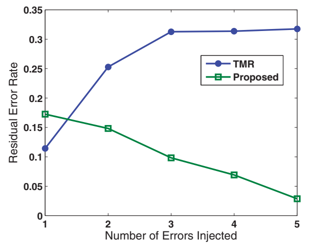


Fig. 14 Residual error rate comparison.

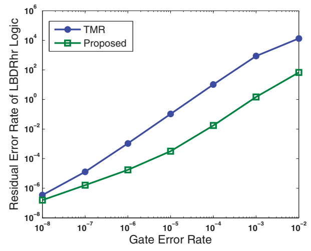


Fig. 15 Impact of gate error rate on LBDRhr reliability.

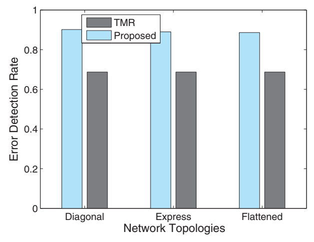


Fig. 16 Impact of network topologies on reliability.

V. Experimental Results

A. Experimental Setup

We evaluated the performance and overhead of the routers without protection, using TMR, and using proposed methods. The experiments were performed on topologies derived from a 4x4 mesh topology, TMR was applied on the entire LBDRhr logic; each output port uses a majority voter to guess the most likely output. The residual error rate is simulated on the gate-level description of the LBDRhr logic. Each data point was obtained from few million random simulations. Area, power and delay are obtained based on the synthesized netlist from Synopsys Design Compiler using a TSMC 65nm typical technology and 1 GHz clock frequency.

B. Reliability

In this experiment, we randomly flipped the logic gate value to simulate the transient error in LBDRhr logic. The residual error rate is the ratio of the number of cases that the error detection method cannot identify the error over the total number of the error injection cases. We vary the number of the error injected in this experiment. As shown in Fig. 14, the residual error rate of the proposed method for 1-bit error is higher than that of TMR. This is because TMR can successfully tolerate 1-bit error, unless the error happens to the majority voter. In contrast, the proposed method has limited capability to detect the switched single-request and mute-request. However, the error detection circuit is not error free; if the number of injected errors increases, our method reduces the residual error rate by up to 90%. If the ratio of failed gates over the total logic unit is fixed, our approach is better than TMR. Because TMR has three times the logic gates as the original design, more logic gates potentially encounter more gate failures. As shown in Fig. 15, our method reduces the residual error rate by two orders of magnitude over TMR.

In Fig. 16, we show the impact of network topologies on the error detection rate. TMR has negligible impact by the network topologies; our approach slightly varies the error detection rate because we use the *Cdest* vector as redundancy information to detect errors. The proposed method also achieves a 30% higher error detection rate than the TMR.

C. Flit Throughput and Latency

For the next experiments we studied the behavior of the 3 proposed topologies using the gNoCsim simulator (an in-house cycle accurate simulator). We first performed the experiments without considering link failures, and then incrementally add faulty links to each topology up to obtain the underlying 2D-mesh. Fig. 17 compares the traffic generated with the maximum mean value of traffic actually received. In all of the cases, the flit throughput decreases with the number of failures up to 0.33 flit/cycle/nic. We also compared the traffic generated with the average network flit latency at the first traffic point (low network load). As shown in Fig. 18, a larger number of link failures leads to a higher latency. Finally, we redid the experiments and additionally considered a faulty underlying 2D-mesh. Similar results were obtained. After examining all the failure combinations of long-range links for the 3 proposed topologies, we found that the LBDRhr logic successfully facilitates all nodes to reach any other nodes in the network; the topologies with a higher radix offer better performance for a given number of failures. In addition, as demonstrated in [7], 80% of the failure patterns of the underlying 2D-mesh were covered by the LBDR approach.

D. Area

The TMR approach need three copies of the core elements in LBDRhr, so consumes more than two times the area compared to the LBDRhr logic without error detection. Our method exploits the inherent information redundancy and does not dramatically induce overhead, only causing a 6.1% area overhead, as shown in Table 2.

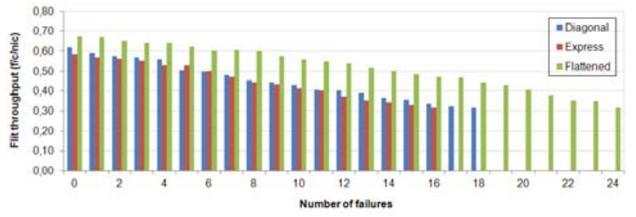


Fig. 17 Flit throughput.

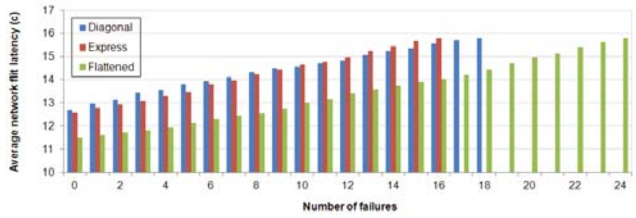


Fig. 18 Flit latency.

TABLE 2 Area, Power and Delay

		LBDRhr without Error Detection	LBDRhr with Proposed Error Detection	LBDRhr with TMR
Area (μm^2)		342 (100%)	363 (106.1%)	806 (235.7%))
Delay (ns)		0.495 (100%)	0.54 (109.1%)	0.51 (103%)
Power	Dyn.(μW)	199.97 (100%)	207.27 (103.7%)	267.39 (133.7%)
	Leak(μW)	1.8084 (100%)	1.8405 (101.8%)	4.0969 (226.5%)

E. *Power and Delay*

As shown in Table 2, the proposed error detection method exploits the inherent information redundancy to reduce the hardware overhead. In the new LBDRhr logic, we find the appropriate internal nodes and discover the exclusive signal pairs, as well as other logical forbidden signal patterns. Error detection based on the inherent information above only causes the dynamic power and leakage power increase by 3.7% and 1.8%, respectively. Compared to TMR, our method reduces the power by 30%. The error detection circuit detects the conflicts between signals, resulting in the increase on the critical path. Although increased by 9.1%, the total worst-case delay 0.54 ns is still below the desired clock period of 1ns.

VI. Conclusions

We proposed a combined method to address both permanent and transient failures in high-end MPSoC systems. For the former, the LBDRhr mechanism is proposed. For the latter, we exploit the inherent information redundancy (IIR) in LBDRhr logic. The high complexity of the LBDRhr logic limits the application of the previous IIR detection approach for XY routing. Compared to the previous work, we discover more inherent information redundancy extraction rules to detect the transient errors in the LBDRhr. This is complement to the methods that use forward error correction for transient errors on the link. Analysis and simulation results show that the proposed method is able to cover permanent failures of all the long-range links and 80% of the failure combinations of the 2D-mesh links. For transient errors, our method reduces the residual error rate and the average power consumption by up to 200x and 30%, respectively, over triple modular redundancy. In the future work, we will extend our solution to manage the errors in the arbiter and crossbar logic.

Acknowledgments

This work was supported by the Spanish MICINN, Consolider Programme, as well as European Commission FEDER funds, under Grant CSD2006-00046. It was also partly supported by the project NaNoC (project label 248972), which is funded by the European Commission within the Research Programme FP7.

References

[1] L. Benini and G. De Micheli, Networks on Chips: Technology and Tools. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2006.

[2] J. Flich and D. Bertozzi, Designing Network On-Chip Architectures in the Nanoscale Era. Boca Raton, FL, USA: CRC Press, Inc., 2010.

[3] M. Coppola, M. D. Grammatikakis, R. Locatelli, G. Maruccia, and L. Pieralisi, Design of Cost-Efficient Interconnect Processing Units: Spidergon STNoC. Boca Raton, FL, USA: CRC Press, Inc., 2008.

[4] D. Wentzlaff, et al. "On-chip interconnection architecture of the tile processor," *IEEE Micro*, vol. 27, no. 5, pp. 15–31, September/October 2007.

[5] J. Owens, W. Dally, R. Ho, D. Jayasimha, S. W. Keckler and L. Peh, "Research challenges for on-chip interconnection networks," *IEEE Micro*, vol. 27, no. 5, pp. 96–108, Sept.-Oct. 2007.

[6] A. Strano, "Exploiting Network-on-Chip structural redundancy for A cooperative and scalable built-in self-test architecture, in *Proc. DATE'11*, pp. 1-6, Mar. 2011.

[7] S. Rodrigo, J. Flich, A. Roca, S. Medardoni, D. Bertozzi, J. Camacho, F. Silla, and J. Duato, "Addressing manufacturing challenges with cost-efficient fault tolerant routing, " in *Proc. NOCS'10*, pp. 25–32, May 2010.

[8] R. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies, "*IEEE Trans. Device & Materials Reliability*, vol. 5, pp. 305–316, 2005.

[9] N. N. Mahatme, *et al.*, "Analysis of soft error rates in combinational and sequential logic and implications of hardening for advanced technologies, " in *Proc. IRPS*, pp. 1031–1035, Apr. 2010.

[10] Q. Yu, M. Zhang and P. Ampadu, "Exploiting inherent information redundancy to manage transient errors in NoC routing arbitration," in *Proc. NOCS'11*, pp.105-112, May 2011.

[11] S. Murali, T. Theocharides, N. Vijaykrishnan, M.J. Irwin, L. Benini, G. De Micheli, "Analysis of error recovery schemes for networks on chips," *IEEE Design & Test of Computers*, vol. 22, no. 5, pp. 434–442, 2005.

[12] S. Sridhara and N. R. Shanbhag, "Coding for system-on-chip networks: a unified framework," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 6, pp. 655–667, Jun. 2005.

[13] Q. Yu, P. Ampadu, "Dual-layer adaptive error control for Network-on-Chip links," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, DOI: 10.1109/TVLSI.2011.2156436.

[14] A. Dutta and N. A. Touba, "Reliable Network-on-Chip using a low cost unequal error protection code," in *Proc. DFT'07*, pp. 3–11, 2007.

[15] Q. Yu, B. Zhang, Y. Li and P. Ampadu, "Error control integration scheme for reliable NoC," in *Proc. ISCAS'10*, pp. 3893–3896, May 2010.

[16] K. Constantinides, *et al.*, "BulletProof: a defect-tolerant CMP switch architecture, " in *Proc. HPCA'06*, pp. 5–16, Feb. 2006.

[17] A. Yanamandra, *et al.*, "Optimizing power and performance for reliable on-chip networks, " in *Proc. ASP-DAC'10*, pp. 431–436, Jan. 2010.

[18] R. E. Lyons and W. Vanderkulk, "The use of triple-modular redundancy to improve computer reliability," *IBM J Res. & Development*, vol. 6, no. 2, pp. 200–209, Apr. 1962.

[19] M. K. F. Schafer, T. Hollstein, H. Zimmer, and M. Glesner, "Deadlockfree routing and component placement for irregular mesh-based networks-on-chip," in *Proc. ICCAD '05*, pp. 238–245, 2005.

[20] W. Song, *et al.*, "Adaptive stochastic routing in fault-tolerant on-chip networks," in *Proc. NOCS '09.*, pp. pp. 32–37, 2009.

[21] T. Skeie, F. Sem-Jacobsen, S. Rodrigo, J. Flich, D. Bertozzi, and S. Medardoni, "Flexible dor routing for virtualization of multicore chips," in *Proc. SOC'09*, pp. 73–76, 2009.

[22] K. Aisopos, A. DeOrio, L.-S. Peh, V. Bertacco, "ARIADNE: Agnostic Reconfiguration In A Disconnected Network Environment," in *Proc. PACT*, Oct. 2011.

[23] D. Fick, et al. "Vicis: a reliable network for unreliable silicon," in *Proc. DAC*'09, pp. 812–817, 2009.

[24] V. Puente, J. A. Gregorio, F. Vallejo, and R. Beivide, "Immunet: A cheap and robust fault-tolerant packet routing mechanism," ACM SIGARCH Computer Architecture News, 32(2), 2004.

[25] J. Cano, J. Flich, J. Duato, M. Coppola and R. Locatelli, "Efficient routing implementation in complex Systems-on-Chip," in *Proc. NOCS'11*, pp. 1–8, May 2011.