PAPER

A Scalable and Reconfigurable Fault-Tolerant Distributed Routing Algorithm for NoCs**

Zewen SHI^{†a)}, Xiaoyang ZENG[†], Nonmembers, and Zhiyi YU^{†*}, Member

SUMMARY Manufacturing defects in the deep sub-micron VLSI process and aging resulted problems of devices during lifecycle are inevitable, and fault-tolerant routing algorithms are important to provide the required communication for NoCs in spite of failures. The proposed algorithm, referred to as scalable and reconfigurable fault-tolerant distributed routing (RFDR), partitions the system into nine regions using the concept of divide-and-conquer. It is a distributed algorithm, and each router guarantees fault-tolerance within one's own region and the system can be still sustained with multiple fault areas. The proposed RFDR has excellent scalability with hardware cost keeping constant independent of system size. Also it is completely reconfigurable when new nodes fail. Simulations under various synthetic traffic patterns show its better performance compared to Extended-XY routing algorithm. Moreover, there is almost no hardware overhead compared to Logic-Based Distributed Routing (LBDR), but the fault-tolerance capacity is enhanced in the proposed algorithm. Hardware cost is reduced 37 % compared to Reconfigurable Distributed Scalable Predictable Interconnect Network (R-DSPIN) which only supports single fault region.

key words: fault-tolerant routing, network-on-chip (NoC), deadlock-free, divide-and-conquer, system partition

1. Introduction

Due to the phenomenon of the Moore's Gap, the performance evolution of single core processor has tapered off even as the number of transistors integrated in the chip has continued to increase exponentially. Multicore processor has shown significant performance and power advantages over single core processor and we expect to see one thousand core chips in a few years [28]. Although the number of commercial processing cores is rather small, it is predicted to be doubled by each generation. Several representative works of NoCs are reported, like Tile64TM from Tiler [1], 80-core Teraflops from Intel [2], and the 167-processor computational platform by UC Davis [3].

2D mesh topology is usually preferred for network-on-chips (NoCs) design such as aforementioned work [1]–[3] due to its regularity and good layout and it is adopted in this work. For deep sub-micron (DSM) VLSI process, increasingly higher integration makes it difficult to guarantee cor-

rect fabrication with an acceptable yield. Moreover, effects like electro-migration [4] and other wear out effects all result in reliability issues. As previously predicted in [5], within a decade we will see 100 billion transistor chips. This is good news for that thousands of cores can be integrated. The bad news is that 20 billion of those transistors will fail in manufacture and a further 10 billion will fail in the first year of operation. While some regions of the chip are defective, the remaining may be fully functional. In order to improve the reliability of NoCs, design techniques have to take into account of these defects and fault tolerant interconnect infrastructures are becoming increasingly important [6].

Table-based method is an intuitive option for fault-tolerant routing. However, tables do not scale in terms of latency, power consumption, and area, thus being impractical for NoCs with large number of nodes. Logic-Based Distributed Routing (LBDR) proposed in [7] gives a new perspective of scalable and efficient implementation, but it is limited to some specific topologies that fault nodes can only be located at four corners, or else pairs of end-nodes cannot communicate through a minimal path defined in the original 2D mesh. Reconfigurable Distributed Scalable Predictable Interconnect Network (R-DSPIN) proposed in [8] can support one fault node or be extended to support one fault area no matter where it is located, but it is limited to only one, so when new node fails, the chip may fail, and the re-configurability is greatly reduced.

The proposed scalable and reconfigurable fault-tolerant distributed routing (RFDR) algorithm is perfectly scalable with hardware cost keeping constant independent of system size. It can support fault node locating at corner, on the edge or in the center of the mesh, and also it is completely reconfigurable when new node fails. The system is divided into nine regions using the concept of divide-and-conquer and three types of routers are needed. Routers within each region are of the same kind and each region guarantees fault-tolerance of one's own and the system can be sustained. Evaluation under various traffic patterns shows its better performance compared to other routing algorithm. And it is also cost effective with almost no hardware overhead compared to LBDR and the area reduces up to 37 % compared to R-DSPIN.

The rest of this work is organized as follows. In Sect. 2, related work and motivation is described. The proposed scalable and reconfigurable fault-tolerant distributed routing (RFDR) algorithm is introduced in detail in Sect. 3, and in Sect. 4, deadlock-freedom and reconfiguration mech-

Manuscript received November 1, 2010.

Manuscript revised February 22, 2011.

[†]The authors are with State Key Laboratory of ASIC & System, Fudan University, Shanghai, P.R. China, 201203.

*The corresponding author.

**This work was supported in part by a grant from Science and Technology Commission of Shanghai Municipality 10706200300, and State Key Laboratory of ASIC & System (Fudan University) 09ZD002.

a) E-mail: dreamic123456@gmail.com

DOI: 10.1587/transinf.E94.D.1386

anism are demonstrated. We compare performance of fault-tolerant routing algorithms under several synthetic traffic patterns in Sect. 5. And hardware cost, fault-tolerance capability, re-configurability, and scalability are further evaluated in Sect. 6. Finally, this paper is concluded in Sect. 7.

2. Related Work and Motivation

Many fault-tolerant routing algorithms in 2D meshes have been proposed. Some use virtual channels [9], [10], however, they increase buffer area and need extra logic for buffer allocation. Alternative way is using routing table. The main advantage of table-based routing is that any topology and any routing algorithm can be adapted, including fault-tolerant routing algorithms, but it lacks scalability in terms of latency, power consumption, and area, thus being impractical for NoCs with large number of nodes. Table minimization is proposed in [11] based on a fixed routing function combined with minimal deviation tables. These tables are used only when the routing decisions for a given destination deviate from the predefined routing function. Routing table compression scheme is adopted in [12], but it is limited to Application Specific Routing (APSR) and loses some generality. In [13], stochastic communication is proposed to achieve fault tolerance in NoC architectures. The nodes communicate using a probabilistic broadcast: data packet is forwarded to a randomly chosen neighboring node until the entire network becomes aware of it. Directed flooding, probabilistic flooding and random walk are derived stochastic routing algorithms [14], [15]. The principal limitation of these algorithms is that they can only sustain a very low traffic injection rate. The concept of region has been proposed to handle cores with larger size than the regular tile size of the mesh and it can be extended to deal with fault areas. Design issues and possibilities of region are pointed out in [16] and several fault-tolerant routing algorithms adapted for region have been presented. Region-based Routing (RBR) proposed in [17] groups destinations into network regions allowing an efficient implementation with logic blocks. However, the number of regions required depends on the regularity of the topology (the number of failures and their positions), and each region needs a complete set of routing logic which also limits its scalability.

The concept of fault ring and fault chain is given in [9]. Fault ring denotes fault region located in the center of mesh structure with all adjacent nodes along the edge of the region healthy. Fault chain means fault region touching one or more edges of the network. Chen and Chiu propose a fault-tolerant routing algorithm [18] (the deadlock problem is corrected in [19]), which can support fault chain and fault ring, however, there is a possibility that Form-Ring procedure [18] might partition the network into unconnected parts and make routing between some pairs of nodes impossible. Traffic-Balanced routing proposed in [20] makes traffic loads evenly spread on the network and shortens the average paths of packets. It can also support multiple fault chains and fault rings, but it is implemented by look-up ta-

ble and lacks scalability. Glass and Ni propose a turn-based fault-tolerant routing algorithm in [21] based on modification of the Negative-First routing. This routing algorithm is made for n -dimensional meshes and can tolerate up to $(n - 1)$ faults, but each routing function depends on the size of the mesh, so it also lacks scalability. Moreover, the fault-tolerance capability of this routing algorithm is limited which only copes with one faulty node for 2-D mesh, but cannot handle one fault region containing several faulty nodes. Negative-First routing is also evaluated with other adaptive routing for fault-tolerance in NoC fabrics [15]. Jie Wu proposes a novel method based on simple dimension order routing (DOR) and odd-even turn model in [22], but it supports fault ring only. Logic-based distributed routing (LBDR) algorithm proposed in [7] gives a new perspective of efficient implementation of NoCs, but it only supports routing of minimum path defined in the original regular mesh topology and can only tolerate fault areas happening at four corners of the mesh structure, like form 'p', 'q', 'd', 'b', and '+' presented in [7]. Here 'p' means fault area happening at the bottom right corner of the mesh structure, and 'q' means fault area happening at the bottom left corner, etc. R-DSPIN proposed in [8] combines X-first routing with turn model and gives a reconfigurable version of fault-tolerant routing. It can support one fault node or be extended to be one fault region (including several fault nodes), but it limits to only one region. From this point of view, the R-DSPIN scheme is not completely reconfigurable if node of new position fails. Scalable and fault-tolerant distributed routing algorithm (SFDR) in our previous work [27] gives an outline of system partition, but does not show the detail of routing restriction sets exploring, reconfiguration mechanism and performance comparison with other routing algorithm. Reconfigurable routing in [29] outlines the configuration procedure, but it lacks performance and fault-tolerance capability evaluation.

Virtual channel routing, table-based routing, Negative-First routing, Extended-XY routing, LBDR, R-DSPIN all have some limits. It is encouraging to develop a routing algorithm that supports fault region wherever it is located with good scalability, re-configurability, and little hardware overhead, and this is our main contribution. The proposed RFDR algorithm inspired by divide-and-conquer concept is to overcome the limitations of LBDR, R-DSPIN and other aforementioned routing algorithms.

3. The Scalable and Reconfigurable Fault-Tolerant Distributed Routing (RFDR) Algorithm

The proposed RFDR algorithm is demonstrated in this section and the key concept of RFDR is system partition and divide-and-conquer. We divide fault regions into three categories including: corner fault chain, boundary fault chain, and fault ring. And they are defined as follows:

Definition 1: Corner fault chain.

Corner fault chain means fault region touches two edges of

the 2D mesh and it is located at corner of the topology as shown in Fig. 1. There are four corner fault chains in total including NE corner chain, SE corner chain, NW corner chain, and SW corner chain.

Definition 2: Boundary fault chain.

Boundary fault chain means fault region touches only one edge of the 2D mesh. As shown in Fig. 1, there are four kinds of boundary fault chains in total including east boundary fault chain, south boundary fault chain, west boundary fault chain, and north boundary fault chain.

Definition 3: Fault ring.

Fault ring means fault region is located in the center of mesh structure (not touching any edge) and all adjacent nodes along the edge of the region are healthy.

Extended-XY routing proposed in [22] supports fault ring, but it does not support corner fault chain and boundary fault chain. LBDR in [7] only supports corner fault chain and does not provide a mechanism to support boundary fault chain and fault ring. R-DSPIN in [8] can tolerate one fault region located anywhere in the mesh structure, but it does not provide a way of reconfiguration with multiple fault regions. Using the concept of divide-and-conquer, the proposed RFDR partitions the system into nine regions. Each region guarantees fault-tolerance of one's own and these nine regions make the system sustain its operation even under several fault regions.

3.1 System Partition

Since fault areas can be refined to three kinds shown in Fig. 1, we are inspired by divide-and-conquer concept to partition the system into nine regions. Each region guarantees to work properly in spite of fault nodes and the whole system can be sustained. The details are shown as follows.

The mesh topology is divided into nine regions using

Fig. 1 All possible fault regions including fault rings, boundary fault chains and corner fault chains.

the concept of 'divide' as shown in Fig. 2. Each region handles fault nodes within one's area and the system holds even with fault node, and this is 'conquer', to make the system fault-tolerant. Specifically, routers in the west boundary region must tolerate fault nodes happening in the west, and the other eight regions treat themselves respectively. If packets are blocked in some direction, some turn is taken to detour around the fault region and transfer packets to the desired destination.

It is noted that proposed RFDR is a distributed routing scheme compared to source routing of which the source node stores the entire path in the packet header. In the proposed solution, the packet header contains the address of the destination node. Routing function integrated in the router is activated and takes effect when packets arrive at current position from other region. In the proposed RFDR, three kinds of routers distribute in nine regions. Each region handles fault nodes within its area, and works together to tackle several fault regions. During routing each router chooses the proper output port to transfer the packet to get closer to the destination and avoid fault region. Once getting out of fault region, packets can advance freely to reach the destinations.

In system design perspective, we can design a router library, and three kinds of routers need to be designed individually. The first kind of router is to be positioned at the four corner regions, the second class is to be located on the four boundary regions, and the third kind is for the center region. This matches well with the system partition. And the overview design of NoC is illustrated in Fig. 3 including router library design and system integration. How to choose the scope of each region has much freedom although different widths of the boundary region have some impact on the fault-tolerance. Detail evaluation is beyond the scope of this paper due to limited space.

The proposed RFDR have three forms including corner RFDR, boundary RFDR, and ring RFDR. If no node fails, all routers within each region behave the same as corner RFDR function which will be illustrated in Sect. 3.3 and 3.4. Nine regions can tolerate fault patterns located within

Fig. 2 Partitioning the system into four corner regions, four boundary regions and one center region. Each region is in a rectangle form.

Fig. 3 Overview of the NoC design flow including router library design and system integration with various sizes.

each region respectively, and they cooperate well to transmit packets to the desired destination nodes even though packets may have to travel around several fault regions.

3.2 Corner RFDR

For routers in four corner regions, corner RFDR is taken and it behaves the same as LBDR proposed in [7]. Each port of the four output ports only needs three bits: one connection bit and two routing restriction bits, and their values are determined by specific topology and routing restriction sets.

Definition 4: Routing restriction.

Routing restriction bits are referred to as R_{xy} , with x and $y \in \{E, S, W, N\}$, and (x, y) are in different orientations. R_{xy} indicates whether packets routed through some ports could make a turn or not at next hop. The two bits at S output port are denoted as R_{se} and R_{sw} , and they indicate whether packets routed through S port could take an east or west turn at next switch.

Definition 5: Connection.

Connection bits are referred to as C_x (one of C_n , C_e , C_s , and C_w), and indicate whether a neighbor node is connected through this port.

Routing logic of LBDR takes three steps. First, pre-

liminary results $N1$, $E1$, $S1$ and $W1$ are computed based on the relative position between the current node and the destination node. For example, if destination node is in the northeast direction of current node, $N1$ and $E1$ are both assigned to '1'. Second, candidate output ports ($N2$, $E2$, $S2$, and $W2$) are calculated based on the preliminary results and the routing restriction (referred to as R_{xy}) sets. And they are calculated based on the following formulas:

$$\begin{aligned} N2 &= N1 \cdot \overline{E1} \cdot \overline{W1} + N1 \cdot E1 \cdot R_{ne} + N1 \cdot W1 \cdot R_{nw} \\ E2 &= E1 \cdot \overline{N1} \cdot \overline{S1} + E1 \cdot N1 \cdot R_{en} + E1 \cdot S1 \cdot R_{es} \\ S2 &= S1 \cdot \overline{E1} \cdot \overline{W1} + S1 \cdot E1 \cdot R_{se} + S1 \cdot W1 \cdot R_{sw} \\ W2 &= W1 \cdot \overline{N1} \cdot \overline{S1} + W1 \cdot N1 \cdot R_{wn} + W1 \cdot S1 \cdot R_{ws} \end{aligned} \quad (1)$$

Third, candidate directions are weighted by connection bits (C_x). After this filter, final output port is calculated.

$$\begin{aligned} N &= N2 \cdot C_n \\ E &= E2 \cdot C_e \\ S &= S2 \cdot C_s \\ W &= W2 \cdot C_w \end{aligned} \quad (2)$$

This is the description of LBDR introduced in [7]. In the proposed boundary RFDR and ring RFDR which will be discussed in Sect. 3.3 and 3.4, the equations for the final output ports (N , E , S , and W) are enhanced with some extra terms to handle different issues that corner RFDR cannot cope with.

3.3 Boundary RFDR

Original LBDR fails if some nodes on four edges of the mesh are faulty since it only supports minimal path routing defined in the original 2D mesh. Boundary RFDR is discussed in this subsection for routers in the four boundary regions. Intuitively, it can make a 90 degree turn if it is blocked in some direction to detour around the fault region as illustrated in Fig. 4.

Without loss of generality, we assume fault nodes take place on the south edge of the mesh (forming south boundary fault chain) as shown in Fig. 4(a). For packets transmitted from node S with destination node D , when packets come to node M , candidate output port $E2$ is active, but C_e is disconnected and the packet is blocked in east direction. In the proposed scheme, packets can behave smartly enough to make a north turn when packets are blocked at east port for the south boundary region, based on equations in Fig. 4(b). In this special case, term $E2 \cdot \overline{C_e}$ takes effect and makes a north turn. When packets get out of the fault chain, they can reach destination nodes freely. Term $W2 \cdot \overline{C_w}$ denotes horizontal mirror scenario of Fig. 4(a) where destination is on the west of south boundary fault chain and packets are blocked in west direction, and term $C_n \cdot N2$ takes effect under normal conditions when no node fails. It is noted that when there are no fault nodes, C_x will never become '0' except nodes along the four edges and these extra terms such

Fig. 4 (a) Detouring around the south boundary chain when packets are blocked, (b) Formulas to calculate N port for south boundary chain.

as $W2 \cdot \overline{Cw}$ and $E2 \cdot \overline{Ce}$ are disabled permanently. Boundary RFDR degrades to corner RFDR if no nodes fail, and routing function integrated in boundary region and corner region behaves the same.

Routing of other three boundary regions can be explained in the same way. Some difference is that in north region, south turn is taken when blocked in east or west direction, while in east (west) region west (east) turn is taken when blocked in north or south direction to detour around the boundary fault chain.

There are some cases that may result in 180-degree turn which is not allowed in the proposed RFDR. As illustrated in Fig. 5, nodes in region I will communicate with nodes in region II. Since destination nodes are in the direction of northwest, candidate output ports N2 and W2 are both active, and there is a possibility that packets may bump into the west edge directly below the fault region and have to take a 180 degree turn to arrive at the desired destination region.

In the proposed scheme, routing is executed in a speculative way. Assuming packets transmitting from southeast of node R1 in Fig. 5, to prevent making extra turns to reach destination nodes in region II, a special west-north unitary restriction is set at node R1 (west-north restriction $R_{wn} = 0$ can be explained that W output port is disabled in advance even though destination is in northwest and W1 is active), and N is the final choosing output port. Similarly, west-north restriction at node A1, west-south restriction at node

Fig. 5 Setting special unitary restrictions to avoid 180-degree turn for the diagonal nodes of the boundary chains.

R2 and A2, north-east at node R3 and B1, and north-west restriction at node R4 and B2 are also set.

3.4 Ring RFDR

For routers in the center region of the mesh structure, if node fails and forms a fault ring, LBDR fails again to work properly which only supports minimal path routing defined in the original 2D mesh. However, the proposed RFDR can make a turn to detour around the fault region.

Definition 6: Wrapper.

Wrapper denotes a set of healthy nodes in a rectangle form which are most close to the fault ring.

As shown in Fig. 6, we set a Res restriction at node B and a Rnw restriction at node E respectively if there is a fault ring (X node fails). Without loss of generality, we take calculation of south output port for example, and it is calculated based on four terms which will be described one by one in detail in the following.

If node E wants to send or forward messages to nodes in its northwest direction, 'W' port is blocked and 'N' is forbidden since there is a Rnw restriction for deadlock-freedom which will be shown in Sect. 4.1, and it cannot forward packets from node E, through node C and to node B or other nodes in the northwest region. And 'S' port is the choice based on term $N1 \cdot W1 \cdot \overline{Cw}$.

For node D with destination nodes in east region blocked at X node with y dimension no lower than the current node (destination nodes in direct east or southeast direction). 'E' is forbidden because it is disconnected in the east direction. And if 'N' is chosen, it cannot forward packets from node A to B, through C and to E since there is a Res restriction at node B. This condition is activated based on term $E2 \cdot \overline{N2} \cdot \overline{Ce}$. As a result, 'S' is the only choice. If packets come from node A, it is easy to infer, and if packets come from node F, it is further discussed based on some judgment mechanism shown in the following. And if packets come from D or west of D and will go to southeast region, 'S'

Fig. 6 Fault ring and formulas to calculate each output port. Each one is decided based on several terms.

port is chosen.

Similarly, term $W2 \cdot \overline{N2} \cdot \overline{Cw}$ indicates node E blocked at X node with y dimension no lower than the current node (destination node in direct west or southwest region), and it also has to take the ‘S’ output port.

Term $Cs \cdot S2$ takes effect under normal conditions when there are no fault nodes in the center region of the mesh.

Each term of other output port’s calculation can be explained in the similar way.

For packets with destination in northwest region, to prevent packets bump to C node to cross the forbidden restriction, routing function is also executed in a speculative way by setting a unitary Rnw restriction at node H, so ‘N’ output direction is disabled even though preliminary result ‘N1’ is true. Similarly, a unitary Res restriction at node A is set to disable the ‘E’ output port in advance.

As to node F, there are no special unitary or bidirectional restrictions, and message coming from west, south or local direction with destination in northeast region can take both north and east output port. This could make some trouble (crossing some forbidden restriction) if destination is node B or more north but choosing east port, or with destination node E or more east but choosing north port. To solve this confusion, we add some judgment mechanism.

Delta_x denotes spacing between destination node and current node in x dimension (if x dimension of destination node is bigger, delta_x is calculated, or else it is set to be ‘0’) and term delta_y denotes spacing in y dimension and it is set in the same way. And here flagx_y denotes whether delta_x is equal to or greater than delta_y. If it is true, flagx_y is set to be ‘1’, or else it is ‘0’.

Routing algorithm can be abstracted as routing function and selection function. Identification flagx_y can be used as selection metric if both ‘N2’ and ‘E2’ are true as shown in Fig. 6 in calculation of north and east output ports. Using this mechanism to select the proper output port also balances the traffic in some way. Only a single fault node is illustrated in Fig. 6, but it can be easily extended to support a fault region by adding some bits to record the position of the upper-right corner of the fault region.

It is noted that when there are no fault nodes, Cx will always be true for nodes in the center region, and extra terms in Fig. 6 compared to formula (2) all become ineffective. And then ring RFDR function degrades to corner RFDR function under this condition.

3.5 Routing Restriction Sets Exploring

SR (Segment Routing) provides a fast computation of deadlock-free routing restriction sets as described in [23], [24], and it is agnostic to the topology of the network. SR is based on a segmentation process of the network by placing routing restrictions at each segment in order to guarantee deadlock-freedom. The computation of a SR-derived routing algorithm follows this routine: divide a topology into subnets, and subnets into segments, and place bidirectional turn restrictions locally within each segment. SR methodology offers different degrees of flexibility when computing routing restriction. First, the starting node can be selected among all the possible nodes in the network. Another main flexibility is that you can compute each segment at will and set routing restrictions anywhere in each segment.

The proposed algorithm assumes all fault nodes are known in advance. This assumption is reasonable for a faulty node can be detected by a dedicated build-in-self-test mechanism, which won’t be focused in this paper. The exploring of routing restriction sets takes a two-step approach. First divide a topology into subnets, and subnets into segments, and second, place bidirectional turn restrictions locally within a segment. There are some principles that need to follow to explore the routing restrictions sets:

1. Start setting bidirectional routing restriction at near NE corner of the wrapper of the fault ring to avoid potential cycle if there is any fault ring as Fig. 6 shows.
2. Set unitary restriction Res at top-left node of the wrapper and Rnw at bottom-right node to disable some output port direction in a speculative way if there is any fault ring.
3. If there are some boundary fault chains, set unitary restrictions at diagonal nodes to avoid 180-degree turn as

Fig. 5 shows.

4. Make sure each cycle has a bidirectional routing restriction to break up the channel dependency using the SR mechanism described in [23], [24].

Figure 7 (a) is an illustration of routing restriction sets of a 6×6 mesh with a one-node fault ring and a one-node boundary fault chain. And we set bidirectional and unitary routing restrictions for different nodes.

As segments are independent, we gain the freedom to place turn restrictions within a segment. This results in a large degree of freedom by placing turn restrictions compared to other fixed strategies like West-First (WF), North-Last (NL), Negative-First (NF), odd-and-even turn, and Extended-XY [22].

Fig. 7 (a) Routing restriction sets exploring including bidirectional restrictions and unitary restrictions, (b) Routing restriction sets reconfiguration with a new boundary fault chain and corner fault chain.

4. Deadlock-Freedom and Reconfigure Procedure

In this section, we illustrate the deadlock-freedom of the proposed RFDR algorithm. Also reconfiguration procedure is expressed to deal with failing nodes due to manufacture defect or aging resulted problems during chip's lifecycle.

4.1 Deadlock-Freedom

Routing restriction sets exploring is proposed in Sect. 3.5 by dividing a topology into subnets, and subnets into segments and placing bidirectional turn restrictions within each segment. The algorithm is deadlock free if no packets cross a forbidden routing restriction. And this promises the routing algorithm to be deadlock free as it proves in [23]. Therefore, RFDR must ensure that no packet pass through any routing restriction set.

Corner fault chain proves to be deadlock free in [26] and it is extended that boundary fault chain is also deadlock-free. It can also be explained in another point of view using the channel dependency graphs (CDG) theory. CDG theory can be used to prove that four corner fault chains and four boundary fault chains are intrinsic deadlock-free following Dally's condition in [25], since in no way a waiting cycle would form, and the escaping paths for remaining nodes to communicate are illustrated in Fig. 8 (a) and Fig. 8 (b) respectively.

The potential cycle can be wrapper of fault ring and other nodes combined with the wrapper nodes. In the proposed scheme, there is a bidirectional turn restriction at NE corner of the wrapper nodes, and this breaks the channel dependency graphs. And the escaping paths for remaining nodes are shown in Fig. 8 (c) for fault ring. For other nodes in the mesh, any possible cycle is also broken through routing restrictions exploring in Sect. 3.5 by setting bidirectional routing restrictions at each segment.

Since routing restriction bits decide the candidate output port, if restriction bit R_{xy} is set to '0', some output port is disabled in advance and no packets will pass through this artificial fence. And this promises the proposed RFDR to be deadlock free.

4.2 Reconfigure Procedure

Some of the nodes may be not correctly manufactured and other nodes may fail due to aging or other wearing out problems such as electro-migration, time-dependent-dielectric-breakdown (TDDB), and negative bias temperature instability (NBTI) [4]. Fault-tolerant and reconfigurable design is especially challenging and inspiring due to these non-perfect effects.

The proposed routing algorithm can be reconfigured, to adapt to the modification of the micro-network topology caused by manufacture defect and hard error through lifetime. When new node is detected to fail permanently (how to detect faulty node needs dedicated BIST mechanism

Fig. 8 Escaping path illustration for (a) corner chain, (b) boundary chain, and the contours are natural cycle free; (c) Fault ring is also cycle free by setting a bidirectional turn restriction at near NE corner of the wrapper.

which will not be addressed in this work), the reconfiguration procedure is done following three steps:

1. New failing node to be merged in a rectangle form, and when fault region crosses more than one region (for example boundary region and corner region), form a

Fig. 9 Connection bit reconfiguration.

big corner fault chain instead.

2. Reset the connection bits of nodes around the fault regions, i.e. change C_s , C_e , C_w , and C_n in the north, west, east, and south direction of the fault node or fault region to be '0' (if it is originally connected) respectively as shown in Fig. 9.
3. Re-execute the routing restriction sets exploring procedure including bidirectional and unitary routing restrictions as shown in Sect. 3.5. An example of the routing restriction sets re-exploring is shown in Fig. 7 (b) when new nodes fail with a boundary fault chain and a corner fault chain.

Of these three steps, step 1 is optional, and step 2 and 3 are imperative during each reconfiguration procedure. Some software procedure within a robust node monitoring the network status can calculate the routing restriction bits and connection bits when new nodes detect to fail, and send configuration packets to the healthy nodes around the fault region. It takes about some hundred cycles and this mainly depends on the number of fault nodes, the time transferring the configuration bits and relative position between the monitoring node and the faulty node.

5. Performance Evaluation

In this section we evaluate the performance of RFDR under various traffic patterns and number of available nodes under various faulty rates and compare it to other fault-tolerant routing algorithm. Also network latency under various fault patterns is evaluated.

5.1 Simulation Environment and Traffic Scenarios

To evaluate the performance of the proposed algorithm, we have developed a RFDR simulator in SystemC language. 2D mesh is assumed with wormhole switching and there are two unidirectional channels between each pair of neighboring routers. Input buffer is adopted with buffer depth of four flits and round-robin is the arbitration scheme. One

flit passes through one router in one cycle and each packet contains up to 10 flits. For each run of the simulation, we simulate 20,000 cycles and the first 1,000 cycles are used as warm-up time. After the simulation, we obtain an average latency of the network versus packet injection rate. Several synthetic traffic patterns are considered, including uniform, transpose I, transpose II, and shuffle traffic.

5.2 Simulation Results

An 8×8 mesh network is constructed to evaluate the performance of the network. Assume fault rings (2, 2) and (3, 5) exist in the center and the network is evaluated under synthetic traffic. Extended-XY [22] routing algorithm which can also support multiple fault rings is selected for fair comparison. Average latency of the network is evaluated.

The latency versus packet injection rate under uniform, transpose I, transpose II and shuffle traffic is shown in Fig. 10 (a), Fig. 10 (b), Fig. 10 (c) and Fig. 10 (d) respectively.

The RFDR performs better than Extended-XY routing [22] in latency under aforementioned traffic scenarios. When the packet injection rate is rather small, the latency is almost the same. But Extended-XY saturates at a lower packet injection rate and the proposed RFDR can sustain a much higher injection rate of the same latency. This is mainly due to traffic balance scheme adopted and the flexibility of setting routing restrictions introduced in Sect. 3.

Also we present the performance penalty of the network under various fault patterns. Fault patterns with 3 holes, 2 holes, one hole in different places of the mesh, and without hole are created. And the latency versus packet injection rate under uniform traffic is shown in Fig. 11.

When the packet inject rate is small, the latency is almost of the same under various fault patterns, but when the inject rate gets higher, the latency disparity is enlarged. Also it shows that fault node located in the centre plays a more important role in impacting the performance of the network than nodes located at the corner.

To evaluate the fault-tolerance capability, we set another experiment to get the number of fully available nodes which can be used as source node and sink node under various faulty rates and compare it with Extended-XY routing using Monte Carlo simulation method.

For simplification, 10×10 mesh network is adopted to get integer fault nodes under 1%~5% faulty rates. And the number of available nodes is shown in Fig. 12.

The number of available nodes of Extended-XY under various fault rates is smaller than that of the proposed RFDR routing for some limitations. First, when the source is in an odd column and Δx is non-zero, packet has to be sent to its west neighbor in an even column as to Extended-XY, however, if the west even column node is faulty as shown in [22]. Second, the destination should not be a boundary node of any faulty block. That is if the destination is at the east side of a faulty block and it is on an even boundary line which is closer to the block than the odd boundary line, and

Fig. 10 Latency comparison under (a) uniform traffic, (b) transpose I traffic, (c) transpose II traffic and (d) shuffle traffic with fault ring.

Fig. 11 Latency of the network under kinds of fault patterns in 8×8 mesh.

Fig. 12 Number of available nodes under variable fault rates in 10×10 mesh network.

also the destination is at the west side of a faulty block and it is on an odd boundary line as described in [22]. However, as to the proposed RFDR, these limitations do not exist and it enjoys bigger number of available nodes than Extended-XY in [22].

6. Evaluation of Cost, Fault-Tolerance Capability, Re-configurability, and Scalability

In this section, the proposed RFDR is implemented in SMIC 0.13 μm CMOS process and hardware cost is compared with several other works. Also it is evaluated in terms of fault-tolerance capability, re-configurability, and scalability.

6.1 Hardware Cost

Three types of routers (designed for corner regions, boundary regions, and center region) are needed adopting the system partition and divide-and-conquer concept. Router designed for the corner region integrates the corner RFDR function, and Boundary RFDR is designed for four boundary regions and ring RFDR is for routers in the center region of the mesh. It is found that these three kinds of routers are almost of the same size from Table 1, and they can be embedded quite well into the tile structure.

Table 1 Area of three types of routers.

Routing Function	Area (k gates)
Corner RFDR	16.98
Boundary RFDR	17.11
Ring RFDR	17.22
Average RFDR	17.11 ^a

Table 2 Area comparisons of selected routing functions.

Routing Function	Area (k gates)
Deterministic DOR	15.84
Table-based (4×4 mesh)	17.40
Table-based (16×16 mesh)	19.59
Table-based (32×32 mesh)	29.45
Fixed-priority LBDR[26]	17.00
R-DSPIN[8]	27.2
Proposed RFDR	17.11

From Table 2, it is found that compared to simple deterministic DOR algorithm, the proposed RFDR has only 1270 gates (8 %) overhead, but DOR is totally without any fault-tolerance capability. The gap between table-based routing and RFDR is enlarged as the number of nodes increases. The proposed RFDR is cost efficient not only for hardware cost keeps constant independent of system size, but also for implementation of RFDR is almost without hardware overhead compared to fixed priority LBDR routing algorithm [26]. LBDR in [26] is synthesized in 65 nm process. When converted to 130 nm process, the area is estimated to be about 17 K gates, which is similar to the proposed RFDR. But it is only partially fault-tolerant which only supports fault regions happening at four corners (corner fault chains). And hardware cost of RFDR reduces 37 % compared to R-DSPIN proposed in [8]. Extended-XY is not implemented due to its limitations listed in Sect. 5.2, and to compensate these restrictions, it has to add extra hardware (like virtual channels) or software cost which will undoubtedly exceed the proposed RFDR.

6.2 Fault-Tolerance Capability

Table-based routing is topology-agnostic and any routing algorithm can be adopted including fault-tolerant routing algorithms, but it lacks scalability in terms of latency, power consumption and area, thus being impractical for NoCs with large number of nodes. A fault-tolerant routing algorithm without using of virtual channels or routing tables is attractive. Several works are only partially fault-tolerant, like Extended-XY proposed by [22] not supporting boundary fault chains and corner fault chains, LBDR proposed in [7], [26] not supporting fault ring and boundary fault chains, and R-DSPIN proposed in [8] tolerating any one fault region, but not providing a mechanism to support several fault regions. The proposed RFDR gives a novel scheme that not only supports fault ring, boundary fault chain, and corner fault chain, but also supports multiple fault regions.

6.3 Re-Configurability

It is no doubt that table-based routing is best reconfigurable of all fault-tolerant routing algorithms. Once there is a path between two nodes, it is configurable no matter what the routing algorithm and topology is. R-DSPIN proposed in [8], can be reconfigured to be one of nine different situations based on relative position of the fault node or fault region when some node fails, but the re-configurability is limited in some way because R-DSPIN does not provide a mechanism to support multiple fault regions. After merging, connection bits reconfiguration around fault region, and routing restriction sets exploring procedure, the proposed RFDR is completely configurable.

Moreover, the configuration bits are far fewer than routing tables especially when the chip contains large number of nodes. Only twelve bits are needed at each node, and each output port of the four (N, E, S, and W) has one connection bit and two routing restriction bits respectively. If it is connected, connection bit C_x is '1', or else it is '0', and if there is a routing restriction at some port, routing restriction bit R_{xy} is '0', or else it is '1'. Once some new node is detected to fail, what to do is merging to form fault region, modifying the connection bits and the routing restriction bits described in Sect. 4.2.

6.4 Scalability

As to table-based routing, each table needs entries up to all possible destinations within a single router although there are some reduction schemes. Thus routing table size grows dramatically with system size and so does the power consumption, and access latency. As to RFDR, no matter what size the system is, all needed at each output port of the four are several information bits and a few logic gates. It is highly scalable compared to table-based implementation, especially when system size is large enough. It is noted that Table 2 is only cost of a single router. Imagine the gap of area cost when the system contains hundreds of and up to one thousand nodes in the near future [28].

And a summary of fault-tolerance, re-configuration capability and scalability of the selected routing algorithms is shown in Table 3.

Table 3 Comparisons of fault-tolerance, re-configurability, and scalability of selected routing algorithms.

Routing Function	Fault-tolerance	Reconfigurable	Scalable
Deterministic DOR	Without	N	Good
Table-based Routing	With	Y	Bad
Extended-XY [22]	Partial	-	Good
LBDR [26]	Partial	Partial	Good
R-DSPIN [8]	Partial	Partial	Good
Proposed RFDR	With	Y	Good

7. Conclusion

This paper presents the RFDR mechanism, a hardware cost efficient, enhanced fault-tolerant, highly scalable, and fully reconfigurable fault-tolerant routing algorithm. Inspired by divide-and-conquer concept, system is partitioned into nine regions. Each region promises fault-tolerance of one's own when packets bound into its area and works cooperatively to guarantee the fault-tolerance. SystemC based simulation results show its less latency compared to other fault-tolerant routing algorithm under various synthetic traffic patterns. Also RFDR is implemented and synthesized results show almost no hardware overhead compared to Logic-Based Distributed Routing, but the fault-tolerance capability is enhanced, and there are only 1270 gates increment (8 %) compared to DOR which is totally without fault-tolerance. Constant area requirement of RFDR shows its excellent scalability. Also it is easy to reconfigure and configuration bits are far fewer than routing tables. Only twelve bits are needed at each router. And the reconfigure procedure is highly simplified. Moreover, the re-configurability of RFDR is better than R-DSPIN which is only partially reconfigurable to some extent. The enhanced reconfiguration capability is beneficial to improve the yield issues and can prolong the chip lifetime in some ways.

References

- [1] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. Mackay, M. Reif, L.-W. Bao, J. Brown, M. Mattina, C.-C. Miao, C. Ramey, D. Wentzlaff, W. Anderson, E. Berger, N. Fairbanks, D. Khan, F. Montenegro, J. Stickney, and J. Zook, "TILE64TM processor: A 64-Core SoC with mesh interconnect," IEEE International Solid-State Circuits Conference ISSCC2008, pp.588–598, Feb. 2008.
- [2] S.R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar, "An 80-tile Sub-100-W TeraFLOPS processor in 65-nm CMOS," IEEE J. Solid-State Circuits, vol.43, no.1, pp.29–41, Jan. 2008.
- [3] D.N. Truong, W.H. Cheng, T. Mohsenin, Z.-Y. Yu, A.T. Jacobson, G. Landge, M.J. Meeuwsen, C. Watnik, A.T. Tran, Z.-B. Xiao, E.W. Work, J.W. Webb, P.V. Mejia, and B.M. Baas, "A 167-processor computational platform in 65 nm CMOS," IEEE J. Solid-State Circuits, vol.44, no.4, pp.1130–1144, April 2009.
- [4] J. Srinivasan, S.V. Adve, B. Pradip, and J.A. Rivers, "Lifetime reliability: Toward an architectural solution," IEEE Micro, vol.25, no.3, pp.70–80, May-June 2005.
- [5] S. Furber, "Living with failure: Lessons from nature?" (Invited paper), Eleventh IEEE European Test Symposium ETS'06, 2006.
- [6] A. Hosseini, T. Ragheb, and Y. Massoud, "A fault-aware dynamic routing algorithm for on-chip networks," Proc. IEEE International Symposium Circuits and Systems, pp.2653–2656, May 2008.
- [7] J. Flich and J. Duato, "Logic-based distributed routing for NoCs," IEEE Comput. Architecture Lett., vol.7, pp.13–16, Jan. 2008.
- [8] Z. Zhang, A. Greiner, and S. Taktak, "A reconfigurable routing algorithm for a fault-tolerant 2D-mesh network-on-chip," 45th ACM/IEEE Design Automation Conference, pp.441–446, June 2008.
- [9] R.V. Boppana and S. Chalasani, "Fault-tolerant wormhole routing algorithms for mesh networks," IEEE Trans. Comput., vol.44,

- pp.848–864, July 1995.
- [10] W.J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," *Proc. IEEE Design Automation Conference*, pp.684–689, 2001.
 - [11] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Routing table minimization for irregular mesh NoCs," *Proc. Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp.1–6, April 2007.
 - [12] M. Palesi, S. Kumar, and R. Holmark, "A method for router table compression for application specific routing in mesh topology NoC architectures," *Proc. SAMOS*, pp.373–384, 2006.
 - [13] T. Dumitras, S. Kerner, and R. Marculescu, "Towards on-chip fault-tolerant communication," *Proc. Asia and South Pacific Design Automation Conference (ASP-DAC'03)*, pp.225–232, Jan. 2003.
 - [14] M. Pirretti, G.M. Link, R.R. Brooks, N. Vijaykrishnan, M. Kandemir, and M.J. Irwin, "Fault tolerant algorithms for network-on chip interconnect," *IEEE Computer society Annual Symposium on VLSI*, pp.46–51, Feb. 2004.
 - [15] H. Zhu, P.P. Pande, and C. Grecu, "Performance evaluation of adaptive routing algorithms for achieving fault tolerance in NoC fabrics," *Proc. IEEE ASAP'07*, pp.42–47, July 2007.
 - [16] R. Holmark and S. Kumar, "Design issues and performance evaluation of mesh NoC with regions," *Proc. IEEE Norchip Conference*, pp.40–43, Nov. 2005.
 - [17] A. Mejia, M. Palesi, J. Flich, and S. Kumar, P. López, R. Holmark, and J. Duato, "Region-based routing: A mechanism to support efficient routing algorithms in NoCs," *IEEE Trans. Very Large Scale Integration (VLSI) Syst.*, vol.17, no.3, pp.356–369, March 2009.
 - [18] K.-H. Chen and G.-M. Chiu, "Fault-tolerant routing algorithm for meshes without using virtual channels," *J. Information Science and Engineering*, pp.765–783, 1998.
 - [19] R. Holmark and S. Kumar, "Corrections to Chen and Chiu's fault tolerant routing algorithm for mesh networks," *J. Information Science and Engineering*, pp.1649–1662, 2007.
 - [20] S.-Y. Lin, C.-H. Huang, C.-H. Chao, K.-H. Huang, and A.-Y. Wu, "Traffic-balanced routing algorithm for irregular mesh-based on-chip networks," *IEEE Trans. Comput.*, vol.57, no.9, pp.1156–1168, Sept. 2008.
 - [21] C. Glass and L. Ni, "Fault-tolerant wormhole routing in meshes," *23rd International Symposium on Fault-Tolerant Computing FTCS-23*, pp.240–249, 1993.
 - [22] J. Wu, "A fault-tolerant and deadlock-free routing protocol in 2D meshes based on odd-even turn model," *IEEE Trans. Comput.*, vol.52, no.9, pp.1154–1169, Sept. 2003.
 - [23] A. Mejia, J. Flich, J. Duato, S.-A. Reinemo, and T. Skeie, "Segment-based routing: An efficient fault-tolerant routing algorithm for meshes and tori," *20th International Parallel and Distributed Processing Symposium*, p.10, April 2006.
 - [24] A. Mejia, J. Flich, and J. Duato, "On the potentials of segment-based routing for NoCs," *37th International Conference on Parallel Processing (ICPP)*, pp.594–603, Sept. 2008.
 - [25] W. Dally and C. Seitz, "Deadlock-free message routing in multiprocessor interconnection networks," *IEEE Trans. Comput.*, vol.36, no.5, pp.547–553, 1987.
 - [26] S. Rodrigo, S. Medardoni, J. Flich, D. Bertozzi, and J. Duato, "Efficient implementation of distributed routing algorithms for NoCs," *IET Computers & Digital Techniques*, vol.3, pp.460–475, Sept. 2009.
 - [27] Z.-W. Shi, K.-D. You, Y. Ying, B. Huang, X.-Y. Zeng, and Z.-Y. Yu, "A scalable and fault-tolerant routing algorithm for NoCs," *Proc. IEEE International Symposium Circuits and Systems*, pp.166–168, May 2010.
 - [28] S. Borkar, "Thousand core chips—a technology perspective," *Proc. ACM/IEEE Design Automation Conference*, pp.746–749, June 2007.
 - [29] Z.-W. Shi, Y.-M. Yang, X.-Y. Zeng, and Z.-Y. Yu, "A reconfigurable and deadlock-free routing algorithm for 2D mesh network-on-chip,"

accepted in *Proc. IEEE International Symposium Circuits and Systems*, May 2011.

Zewen Shi received his B.S. degree in Electronic of Science and Technology from Huazhong University of Science and Technology in 2008, and he is currently working towards the MS degree at State Key Lab of ASIC & System, Fudan University, Shanghai, P.R. China. His research fields include the platform-based VLSI architecture modeling and evaluation, fault-tolerant micro-architecture, routing algorithm, and reliability design for on-chip networks.

Xiaoyang Zeng received the B.S. degree from Xiangtan University in 1996 and the Ph.D. degree from Changchun Institute of Optics and Fine Mechanics, Chinese Academy of Sciences in 2001. From 2001 to 2003, he worked as a post-doctor researcher at the State-Key Lab of ASIC and System, Fudan University, P.R. China. Then he joined the faculty of Department of Micro-electronics at Fudan University as an associate professor. His research interests include information security chip design, VLSI signal processing, and communication systems. Prof. Zeng is the Chair of Design-Contest of ASP-DAC 2004 and 2005, also the TPC member of several international conferences such as ASICON 2005 and A-SSCC 2006, etc.

Zhiyi Yu received the B.S. and M.S. degrees in electrical engineering from Fudan University, Shanghai, China, in 2000 and 2003, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, Davis in 2007. From 2007 to 2008, he was with IntellaSys Corporation, CA, USA, where he participated in the design of the many-core SEAForth chips which utilize stack-based processors with extremely small area and low power consumption. In January 2009 he joined the State Key Laboratory of ASIC & System, Microelectronics Department, Fudan University, China, where he is now an associate professor. His research interests include high-performance and energy-efficient digital VLSI design with an emphasis on many-core processors. He serves as a member of the Technical Program Committee of the IEEE Asian Solid-State Circuits Conference (ASSCC) and the IEEE International Conference on ASIC. He has published 1 book and over 30 papers, and has applied 3 patents.