2013 IEEE 11th International Conference on Dependable, Autonomic and Secure Computing

A Fault-tolerant Routing Algorithm for NoC Using Farthest Reachable Routers

Junshi Wang, Letian Huang, Guangjun Li
School of Communication and Informaion Engineering
University of Electronic Science
and Technology of China
Chengdu, China
Email: wangjsh@std.uestc.edu.cn, {huanglt, gjli}@uestc.edu.cn

Xiaohang Wang, Terrence Mak
Intelligent Chips and Systems Research Centre
Guangzhou Institutes of Advanced Technology,
Chinese Academy of Sciences
Guangzhou, China
Email: {baikeina, terrencemak}@gmail.com

***Abstract*—As technology scaling, reliability has became one of the key challenges of Network-on-Chip (NoC). Many fault-tolerant routing algorithms for NoC are developed to overcome fault components and provide reliable transmission. But proposed routing algorithms do not pay enough attention to find the shortest paths, which increases latency and power consumption. In this paper, a fault-tolerant routing algorithm using new component states diffusion method based on Farthest Reachable Router (FRR) is proposed. This algorithm can reduce latency by finding the shortest paths between source and destination routers. Experiment results verify that FRR routing algorithm can tolerate 79% fault patterns within 3×3 and reduce latency by 16-44% compared with FON.**

***Index Terms*—network-on-chip; fault-tolerant routing algorithm; farthest reachable router;**

I. Introduction

With the power of Moore's rules, more and more cores can be integrated into a single chip, which gives much challenge to the communication structure of on-chip systems. Networks-on-Chip (NoC) can provide more bandwidth and high scalability than traditional on-chip communication structures. Many-core systems based on NoC have widely researched and developed [1][2]. This work is based on network-on-chip with mesh topology [3].

Since semiconductor technology reached deep submicron era, reliability of NoC is facing more stress [4]. It is quite common for components on chip to be faulty. Faults in NoC can be classified into two types: transient fault and permanent fault. The former is usually caused by power grid fluctuations, particle hits and crosstalk. Most time, transient fault can be recovered by retransmission mechanism or error correcting codes (ECC). In the traditional sense, permanent fault is caused by physical damages. Faults caused by power limitation, thermal management can also be treated as permanent fault. To overcome permanent fault, routing paths of packets should be reorganized to round the fault components. Thus fault tolerant routing algorithms have been widely used.

The main thinking of routing algorithms is finding available paths based on component states known by routers. Usually, component states actually mean the states of paths from one router to another, including output buffers, links and input buffers. Sometimes, crossbar and control logic are included as well. Component state is living means packets can be delivered to the next router through this path successfully.

Conventional fault-tolerant routing algorithms reroute packets rounding faulty regions, either convex or concave, so that the selected paths are not always the shortest. However, rerouting is an expensive solution and considerably increases packet latency and router complexity. In addition, the information about fault components is insufficient or the way of utilizing them is inefficient [5]. If routers know more component states, they can find paths meeting more serious requires or constrains. On the other side, more component states known by routers lead to more complex hardware and more power consumption on component states diffusion. Thus, it should be carefully designed to improve performance by increasing the number of component states known by routers.

In this paper, a novel routing algorithm aimed at reducing latency by achieving the shortest paths is proposed. This routing algorithm uses new component states diffusion mechanism based on Farthest Reachable Router (FRR), and it is called FRR routing algorithm. Experiment verifies that this routing algorithm can 1) tolerate all fault pattern with one or two fault routers and 79% fault patterns within 3×3; and 2) reduce the packet latency by 44% at most compared with FON.

The rest of this paper is organized as follow. Section II reviews some presented work. Section III introduces FRR routing algorithm. FRR component states diffusion method and slope division method are also discussed in Section III. Experiment results are presented in Section IV.

II. Related Work

In last decades, fault tolerant is a hot topic in NoC research with lots of work proposed. Fault-tolerant routing algorithms can be classified into two main types: deterministic and adaptive routing algorithm [6][7][8]. A deterministic routing algorithm uses a fixed path for each pair of nodes resulting in increased packet latency especially in congested networks. In adaptive routing algorithms, packets are not restricted to a single path when traversed from a source router to a destination router. So, they can decrease the probability of routing packets through congested areas and thus improve the performance. For adaptive routing algorithms, according to the region of component states used to calculate paths, they can be further divided into global component states, local component states and partial component states algorithms.

Global component states mean each router knows the states of all components in network directly or equivalently, like Q-routing [9] and dynamic programming [10]. Q-routing uses cost functions to update the routing tables kept by each router. Performance has tight relationship with the quality of cost functions. Convergence is one of the key challenges to this kind of routing algorithms.

In routing algorithms using local component states, routers only know the states of components connecting themselves and their neighbors directly (also called port states), like XY and Gradient [11]. Within this kind of routing algorithms, network is divided into several zones, and paths are selected based on the zone which the destination address belongs to. Performance is determined by partition and selection strategy. Different from XY and even-odd using partition based on two dimensions, Gradient uses slope division.

Except the two classes above, there are some routing algorithms only needing state components of partial network, like FON [12] and MAFA [13]. FON is based on deflection routing algorithm. States of 16 components are known by each router in FON. Nevertheless, FON considers two hops in a route computation, which highly increases hardware complexity and consumption. MAFA is developed based on DyXY routing algorithm. 24 component states known by a router give MAFA the ability to find the shortest paths for more source-destination pairs. The complexity of circuits cannot be ignored when evaluating this kind of routing algorithms.

In summary, compared with routing algorithms only using local component states, partial component states routing algorithms can achieve better routing performance, like lower latency, due to the more information known by routers. But hardware consumption should be treated seriously to optimize system performance.

III. FRR Fault-tolerant Routing Algorithm

In order to find out the shortest paths, two important improvements are applied in proposed routing algorithm. One is slope division which reduces average distance by a more powerful partition and selection strategy [11]. The other is component states diffusion method based on Farthest Reachable Router (FRR) which reduces average distance by increasing component states known by routers.

A. component state diffusion method

Definition 1. *Port (*P*) direction* is the direction which can be reached from port (P) without any turn. For the current router in Fig. 1, the directions of ports are {North, South, West, East}.

Definition 2. Routers which can be reached from port (P) along port (P) direction without any turn are called reachable routers of port (P). *farthest reachable router (FRR) of port (*P*)* is coordinate of the router which is farthest away from current router among reachable routers of port (P). In mesh network,

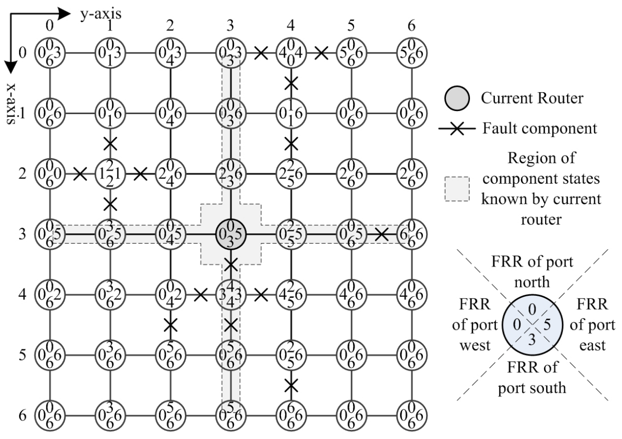


Fig. 1. FRR example

only one dimension of coordinate is needed to store and diffuse. FRRs of port west and port east need the coordinate on y-axis. FRRs of port north and port south need the coordinate of x-axis.

For current router in Fig. 1, because all links on port north direction are not faulty, farthest reachable router from port north is router (0,3), and coordinate on port north direction is 0, which is FRR of port north. As the link (4,3) is faulty, farthest reachable router of port south is current router (3,3) itself, so FRR of port south is 3. FRRs of port west and east are 0 and 5 show that packets can reach router (3,0) and (3,5) along port west direction and port east direction.

From FRRs, current router knows not only the states of fault components next to farthest reachable routers on port directions, but also the states of components from current router to its farthest reachable routers on port directions. Thus, the FRRs contain not only fault states but also living states, which increases the number of component states known by each router sharply. In best case, $\lceil 2\log_2 s \rceil$ bits can contain states of $s-1$ components when all routers on this axis are living. Unfortunately, if a component on port direction is faulty, the states of components behind the fault component can not be known by current router. Thus, in worst case, only state of one component is known by current router, when its neighbor on port direction is faulty, like port south of current router in Fig. 1.

The block diagram of a router is shown in Fig. 2. Each router has 4 FRR channels to store and propagate FRR. It is important that the direction of FRR channel and the direction of FRR is opposite. FRR channels of all routers form a mesh network (Fig. 1). Each FRR channel has one FRR state and propagate circuit to determine and store FRR of current router (upper part of Fig. 2). If the component state is living, FRR of current router is same as FRR came from neighbor, which means packets from this router can reach the same router as its neighbor; If the component state is faulty, FRR of current router is the coordinate of current router.

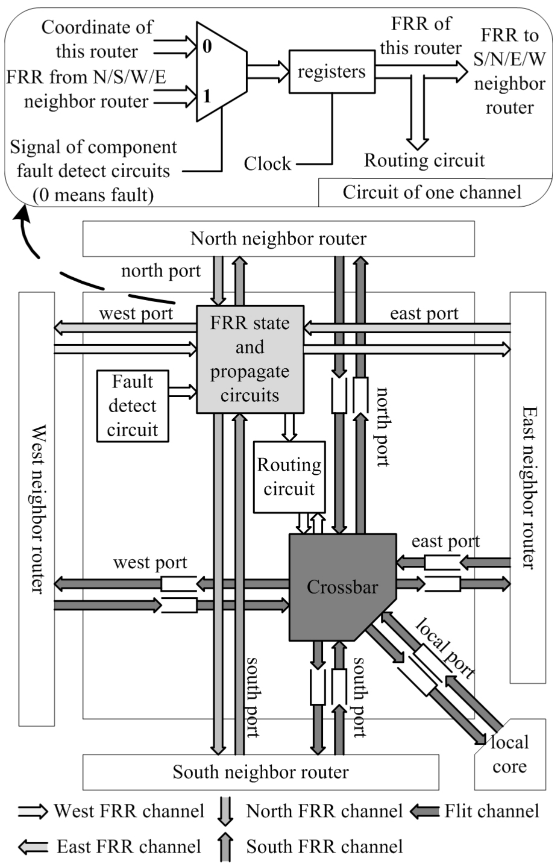


Fig. 2. FRR store and propagate circuit

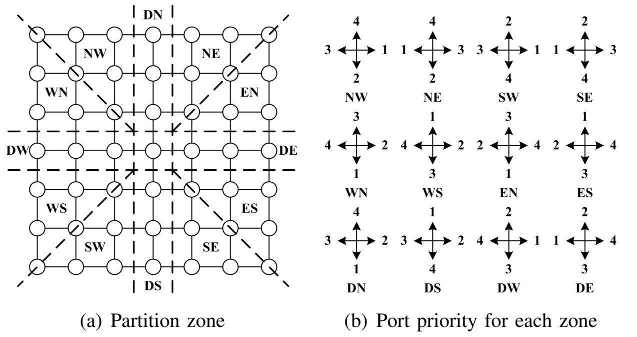


Fig. 3. Partition zone(a) and priority sequence for each zone (b) (4 is highest priority and 1 is lowest)

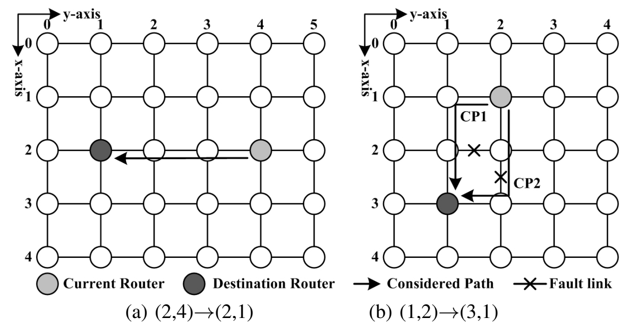


Fig. 4. Two examples about considered paths.

B. slope division method

In NoC routing algorithms, the destination address of packages in network is classified into different zones, and priority sequences will be set to ports according to the zone which address belongs to. Classical fault-tolerant routing algorithms classify destination routers based on vertical and horizontal lines [14] and have advantage for the destination routers on the same line or same row with current router. Slope partition in [11] is more suitable for destination routers in quadrants.

In this work, network is divided into 13 zones (Fig. 3(a)). One zone is current router, 4 zones are on axis and 8 zones are in quadrants. The priority for each zones is shown in Fig. 3(b). Each cross shows four ports of a router and numbers on arrows mean priorities of ports. 4 is the highest priority while 1 is the lowest. This division method can be seen as the combination of slope partition and classical partition, so it has advantages of both partition strategies above.

C. FRR routing algorithm

FRR routing algorithm has three phases (Alg. 1):

Phase 1 (line 3). The priority sequence will be determined according to the destination address of a packet (Fig. 3(b)).

Phase 2 (line 5-14). The priority sequence is adjusted according to the known component states. The strategy is finding out fault components and rounding them earlier. On the other hand, component states known are still not enough for each router to discover all component faults in network. Thus, only increasing priority of ports is allowed to avoid gambling on unknown component states. Based on destination router position, there are two cases.

1) First case is that destination router and current router are on the same row or column (Fig 4(a)). In this case, priorities of ports are not changed. There is only one shortest path (arrow in Fig. 4(a)) in this case, which also goes out from the highest priority port. Increasing priority of the highest priority port means nothing.
2) The other case is that destination router do not have same coordinate with current router (Fig. 4(b)). In this case, there are two shortest paths with the least turns (arrows in Fig. 4(b)) and they start from ports with the highest two priorities. If first turn router of a path is reachable, the priority of port where this path starts will increase, as this path is more likely to be reachable.

Thus, if first turn routers of both two paths are reachable or not reachable, the relationship between the highest two priority ports is not changed. If only first turn router of one path are reachable, the port where this path starts is the highest and packet will go out from this port (Like CP2 in Fig. 4(b)).

Phase 3 (line 16-17). The last phase is used to find out the port with the highest priority among ports with living components. Packages cannot be sent to the port where it

Algorithm 1 N-FRR routing algorithm.

dst: address of destination router; cur: address of current router; frr: FRRs known by current router; $state$: states of the ports of current router; $inputport$: port where packet comes from.

```
procedure $sel$=NFRR($dst$, $cur$, $frr$, $state$)
    // Phase 1 : partition
    Set the priority of each port $pri$ according to the destination address (Fig. 3).
    // Phase 2 : adjusting priority
    if $dst$ on same row or column with $cur$ then
        Nothing to do.
    else
        if The highest priority port $p_1$ is reachable then
            Increase $pri[p_1]$ by $\Delta$.
        end if
        if The second highest priority port $p_2$ is reachable then
            Increase $pri[p_2]$ by $\Delta$.
        end if
    end if
    // Phase 3 : transmission
    Get the living port with the highest priority or $inputport$ as the final selected port $sel$.
    Send this package out from $sel$.
end procedure
```

comes in, except all other ports are faulty.

D. Deadlock avoidance

The key point to avoid deadlock is to break the turning circles. Unfortunately, in order to go around fault components, no turn can be forbidden. In FRR routing algorithm, four virtual channels are employed to deliver packet to different zones. NE, EN and DE belong to VC1; ES, SE, and DS belong to VC2; SW, WS and DW belong to VC3; WN, NW and DW belong to VC4. Thus, turning circles can not be formed in single VC. Each virtual channel is a subnetwork, thus entire network is deadlock free.

IV. Discussion and Verification

A. Simulation configuration

In this section, the latency and power of Gradient, FON and FRR routing algorithms are experimented. The simulation environment is configured as Table I.

In this experiment, only fault routers are considered. One fault router can be seen as four fault components as router (2,1) in Fig. 1. According to whether influence of fault routers will overlap, we define the size in x-axis (y-axis) of fault pattern as the max number of column (row) with fault routers continually (Fig. 5). Take Fig. 5(a) as example. From up to down, line 1 and 2 have fault routers. From left to right row 1 and 3 have fault routers and row 2 without fault separates two lines with faults. Thus, the size of this pattern is 1×2. Due to symmetry of network, $a \times b$ and $b \times a$ are seen as one kind $a \times b \, (a \leq b)$.

TABLE I
Simulation configuration

Argument	Value
Simulator	Popnet
Network size	10×10
Virtual channel	4
Input buffer size (flit)	12
Output buffer size (flit)	12
Flit size (byte)	4
Package length (flit)	1 or 5
Simulation time (cycle)	200000
Traffic pattern	Uniform or PARSEC

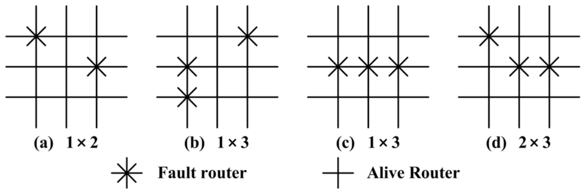


Fig. 5. pattern for simulation.

B. Latency experiment

In this section, Gradient, FON and FRR are simulated by Popnet NoC simulator with configuration in Table I.

Fig. 6 shows the average latency in pattern Fig. 5(b)(c)(d) with uniform traffic. Result verifies that FRR can achieve lower latency than Gradient and FON because of decrease of average distance.

Fig. 7 shows the average latency in pattern Fig. 5(b) with PARSEC [15] traffic. Comparing with Gradient, FRR routing algorithm can reduce latency by 30% at most with canneal benchmark. Comparing FRR with FON, FRR reduces latency to 44% of latency using FON with dedup benchmark at most and 16% with raytrace benchmark at least. Performance of routing algorithm has relationship with fault patterns and source-destination pairs (further experiment in Section IV-D).

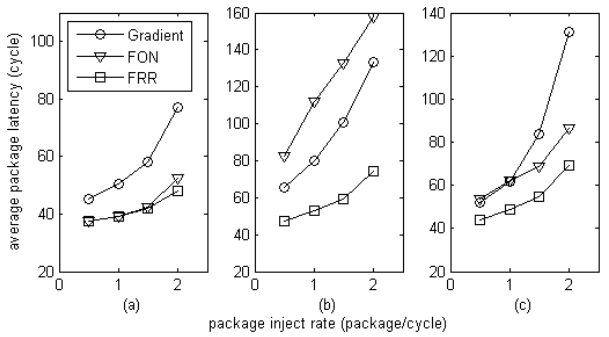


Fig. 6. latency of patterns with uniform traffic ((a)-(c) for patterns in Fig. 5(b)-(d)).

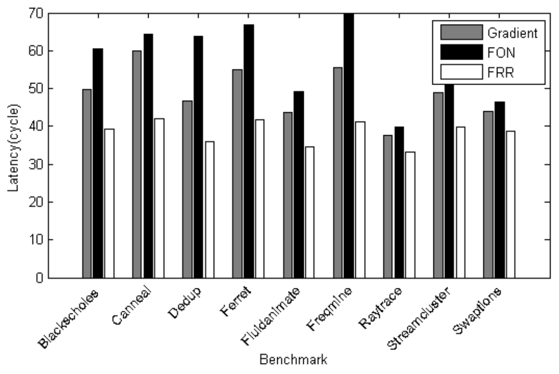


Fig. 7. latency of pattern in Fig. 5(b) with PARSEC traffic.

Even though the fault pattern is constant, different probability distribution of source-destination pairs can lead to different average latency. Thus the performance of FRR routing algorithm for different benchmarks is different.

C. Repair rate experiment

To evaluate the reliability of Gredient, FON and FRR, all fault patterns within 3×3 are tested. Reliability is measured based on two metrics. Using the first reliability metric, we measure the number of combinations with no packet loss over the total number of combinations. For the second metric, the average number of successful packet arrivals at destinations into the total number of delivered packets is calculated. Result in Fig. 8(a) is measured based on first metric. Fault patterns with 2×2 do not loss any packet using FRR. For patterns with 3×3, FRR can tolerant 79% fault patterns, while the percent is only 23% and 71% for Gradient and FON. Fig. 8(b) shows the ratio of successful delivered paths to all paths in 5×5 network. 99% Paths can be delivered successfully using FRR while the percent is 98% and 94% for Gradient and FON.

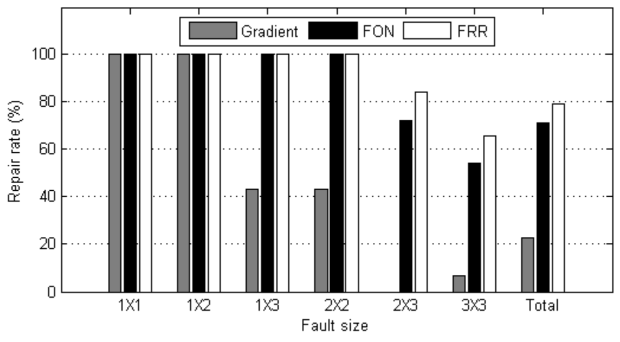
D. Hop count experiment

In this section, we consider the fault patterns in 5×5 network with no faults on boundaries of network. Thus, there are six different sizes for patterns and totally 511 fault patterns. As the target of this work is to reduce latency by decreasing the path length, so we define the increasing rate of redundancy hop to estimate the ability of achieve shorter paths of FRR routing algorithm.

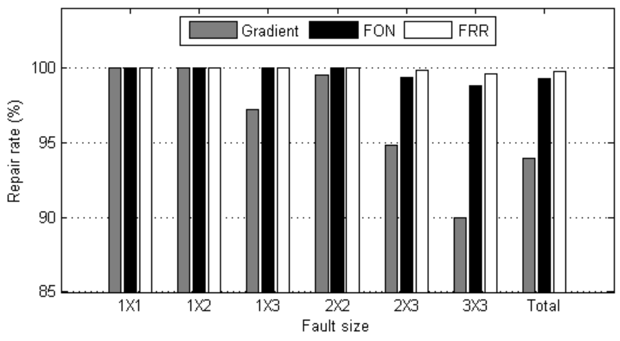
Definition 3. In order to go around fault components, more hops must be added to manhattan distance. These hops is called *redundancy hop*. And *theory limit of redundancy hop* is the redundancy hop of the shortest path. *The increasing rate of redundancy hop (RHIR)* is defined as

$$\text{RHIR} = \frac{\text{redundancy hop} - \text{theory limit}}{\text{theory limit} - \text{manhattan distance}}.$$

In fault-tolerant routing algorithm, we can only reduce redundancy hops, and the hop count cannot be less than manhattan distance. If the increasing rate is 0, the path is the shortest path. Thus, we consider RHIR for different size pattern in this section (Fig. 9).



(a) First reliability metric



(b) Second reliability metric

Fig. 8. Repair rate for different size pattern

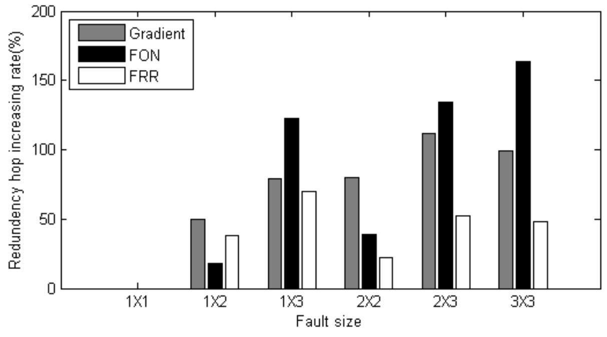


Fig. 9. Redundancy hop increasing rate for different size pattern.

The RHIR is shown in Fig. 9. FRR can reduce redundancy hops to only 48.89%, which is half of Gradient and one third of FON. Sharply reduction of redundancy hops gives FRR great improvement in latency.

Reduction of RHIR is not same for patterns with different size. For 1×1 fault patterns, all three tested algorithm can reduce redundancy hops to 0. For 1×2 fault patterns, FRR is not as effective as FON because component states known using FRR is not suitable to find out this kind of faults, like Fig. 5(a). For other fault patterns, FRR can reduce the redundancy hops and achieve much better performance.

TABLE II
HARDWARE AND POWER CONSUMPTION

Algorithm	Area(mm^2)	Power(mW)
FRR	1.3963	2.2132
Gradient	1.2369	1.6112
FON	1.4714	2.5029

E. Hardware consumption analysis

The FRR routing algorithm circuit are synthesized by Synopsys Design Compiler. Routing circuits of gradient and FON are also synthesized as comparison. For synthesizing, we use the SMIC 0.13μm technology at the operating frequency of 100MHz and supply voltage of 1.08V. For power evaluation, a 5×5 network is considered and packet inject rate is set as 1 packet/cycle with uniform distribution. Area in Table II is the single router area consumption. Power is the average power for each router.

Compare results in Table II. FRR costs more area and power than Gradient, because the added FRR channel circuit and routing logic. FRR costs less area and power than FON due to simplified circuit logic design.

V. CONCLUSION

Farthest Reachable Router (FRR) fault-tolerant routing algorithm for mesh NoC is presented in this paper. It uses Farthest Reachable Router (FRR) to propagate and storage component states, which contains information about fault and living components at the same time. More component states known by routers are the key reason for FRR routing algorithm to reduce average path length. Experiment results verify that FRR can tolerate all fault patterns within 2×2 and reduce the latency by 44% at most comparing with FON routing algorithm with PARSEC benchmark.

In future work, we will explore intelligent routing strategy to handle more complicated fault patterns. Also, we will pay attention to balance the cost of fault information diffusion and routing packets.

ACKNOWLEDGMENT

The research is support by National Natural Science Foundation of China No. 61176025 and the Fundamental Research Funds for the Central Universities No. ZYGX2012J003.

REFERENCES

[1] S. Kumar, et al., “A network on chip architecture and design methodology,” *Proc. IEEE International Symposium on Circuits and Systems*, pp. 105-112, 2005.

[2] Xu, Jiang, et al., “A Methodology for design, modeling and analysis for networks-on-Chip,” *Proc. IEEE International Symposium on Circuits and Systems*, pp. 1778-1781, 2005.

[3] W. Dally, C. Seitz, “Deadlock-free message routing in multiprocess interconnection networks,” *Computers, IEEE Transactions on*, vol. C-36, no. 5, pp. 547-533, 1987.

[4] J. Henkel, et al., “Reliable on-chip systems in the nano-era: Lessons learnt and future trends,” *Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE*, pp. 1-10, 2013.

[5] M. Ebrahimi, et al., “MD: Minimal path-based fault-tolerant routing in on-Chip Networks,” *Design Automation Conference (ASP-DAC), 2013 18th Asia and South Pacific*, pp. 35-40, 2013.

[6] J. Duato, et al., “Interconnection networks: an engineering approach,” *Morgan Kaufmann Publishers*, 2003.

[7] M. Ebrahimi, et al., “CATRA-congestion aware trapezoid-based routing algorithm for on-chip Networks,” *in Proc. 15th ACM/IEEE Design, Automation, and Test in Europe (DATE)*, pp. 320-325, 2012.

[8] M. Dehyadegari, et al., “An adaptive fuzzy logic-based routing algorithm for networks-on-chip,” *in Proc. 13th IEEE/NASA-ESA International Conference on Adaptive Hardware and Systems (AHS)*, pp. 208-214, 2011.

[9] F. Farahnakian, et al, “Q-learning based congestion-aware routing algorithm for on-chip network,” *Networked Embedded Systems for Enterprise Applications (NESEA), 2011 IEEE 2nd International Conference on*, pp. 1-7, 2011.

[10] T. Mak, et al, “Adaptive Routing in Network-on-Chips Using a Dynamic-Programming Network,” *Industrail Electronics, IEEE Transactions on*, Vol.58, No. 8, 2011.

[11] I. Pratomo, S. Pillement, “Gradient - An adaptive fault-tolerant routing algorithm for 2D mesh Network-on-Chips.” *Design and Architectures for Signal and Image Processing (DASIP), 2012 Conference on*, pp. 1-8, 2012.

[12] C. Feng, et al, “FoN: Fault-on-Neighbor aware routing algorithm for Networks-on-Chip,” *SOC Conference (SOCC), 2010 IEEE International*, pp. 441-446, 2010.

[13] M. Ebrahimi, et al, MAFA: Adaptive Fault-Tolerant Routing Algorithm for Networks-on-Chip *Digital System Design (DSD), 2012 15th Euromicro Conference*, pp. 201-207, 2012.

[14] Y. Chen, et al, “A deadlock-free fault-tolerant routing algorithm based on pseudo-receiving mechanism for Networks-on-Chip of CMP,” *Multimedia Technology (ICMT), 2011 International Conference on*, pp. 2825-2828, 2011.

[15] PARSEC benchmark suite, http://parsec.cs.princeton.edu.