Microprocessors and Microsystems 43 (2016) 104–115

Contents lists available at ScienceDirect

Microprocessors and Microsystems

journal homepage: www.elsevier.com/locate/micpro

Tolerating transient illegal turn faults in NoCs

Letian Huang[a,*], Xiaofan Zhang[a], Masoumeh Ebrahimi[b], Guangjun Li[a]

[a] University of Electronic Science and Technology of China, China
[b] KTH Royal Institute of Technology, Sweden and University of Turku, Finland

ARTICLE INFO

Article history:
Received 7 July 2015
Revised 8 November 2015
Accepted 25 January 2016
Available online 15 February 2016

Keywords:
Network-on-Chip
Fault tolerance
Transient faults
Illegal turns

ABSTRACT

Network-on-Chip (NoC) is becoming a competitive solution to connect hundreds of processing elements in modern computing platforms. Under the trend of shrinking feature sizes, circuits are likely to suffer from faults which lead to degraded performance and erroneous behaviour. Compared to permanent faults, transient faults happen even more frequently and seriously while they are hidden within complex on-chip behaviours. One of the serious consequences caused by transient faults is taking illegal turns by the packets after the damage of control logic in on-chip routers which may lead to a deadlock situation and eventually crashing the entire system. To avoid this situation, in this paper, we propose a comprehensive scheme called ODT including an improved router architecture, an illegal-turn-resilient routing algorithm, online fault-detect units and a fault classification method. By applying ODT, more turns are supported on routing level and the deadlock situations can be significantly reduced. Experimental results indicate up to 22% increase of the survived packets in the network when 4% of routing computation units in failure. The extra area overhead and power consumption of ODT method is around 9.22% and 9.63%.

1. Introduction

With the characteristic of distributed and shared computing resource, Network-on-Chip has shown its remarkable advantages over traditional communication platforms [1]. As feature size of integrate circuits decreases aggressively, chips are getting more susceptible to faults and consequently various issues appears, challenging reliable services [2]. It is inevitable that on-chip network will suffer from the same problems. Among the others, faults may occur during the working phase because of the time-dependent variation such as the Negative Bias Temperature Instability (NBTI) [3] or Electromigration [4].

NoC routers generally consist of five stages as routing computation (RC), virtual channel allocation (VA), switch allocation (SA), crossbar (Xbar), and link traversal (LT) [5]. The RC unit unwraps the header of the incoming packets and decides to which direction the packet should be delivered. The VA unit determines the virtual channel in which the packet should be delivered from. In the SA unit, packets are granted to traverse the crossbar (Xbar) unit. Finally, packets pass the output channel (LT) toward the next switch. The RC, VA and SA units are part of the control path by directly affecting the routing decision while buffers, LT and ST are categorized as the units in the data path, determining the path taken by the packets to pass through the router.

When the faults occur in the data path, the most common and obvious phenomenon is that faults affect the data carried by packets during transmission. Fortunately, Error Detecting Code (EDC) and Error Correcting Code (ECC) can be designed to detect and to correct the faults in the packet [6,7] which protect the data path against faults. Also the data redundancy mechanism, such as retransmission, can also be applied [8]. These methodologies would guarantee the protection of the packet's contents, and thus we can assume that flits are well protected. Unlike the faults in the data path, which can be worked out by mature fault-tolerant methods, faults in the control path of NoC routers are hard to be detected and even harder to be tolerated. A fault (e.g. stuck-at-1 or stuck-at-0) in the control path may lead to, among the others, the miscalculation of the routing algorithm or the wrong matching pair between the input and output port in the crossbar unit. When these faults are introduced, even for a short period, packets may be forwarded to a wrong direction and eventually lead to the deadlock or livelock.

In order to enhance the reliability of on-chip networks, we propose a routing-level solution to address the faults in the control path, targeting those transient faults leading to illegal turns. This solution is called ODT, an Online fault Detection and Tolerance mechanism. In our previous work [9], a non-minimal and fault-tolerant routing algorithm is presented which is able to support around 50% of all possible turns, guaranteeing the freedom from deadlock for both permanent and transient faults. In this paper, we make our focus on the remaining turns and try to reduce the

* Corresponding author. Tel.: +46760798377.
E-mail address: hianglt@uestc.edu.cn (L. Huang).

http://dx.doi.org/10.1016/j.micpro.2016.01.016
0141-9331/© 2016 Elsevier B.V. All rights reserved.

negative effects of transient illegal turns, avoiding them to lead any deadlock or livelock. This algorithm practically does not guarantee the freedom from deadlock in all conditions but showing that many of deadlock situations can be avoided by carefully forwarding the affected packets. In the other words, the presented method can significantly reduce the probability of forming deadlock under transient illegal turn faults.

The remainder of this paper is organized as follows. In Section 2, related work is given. In Section 3, the proposed router architecture is presented. In Section 4, the components of the ODT are introduced. The analytical and experimental results are reported in Section 5 while the summary and conclusion are given in the last section.

2. Related work

The functionality of on-chip network can be highly affected by faults generated by time-dependent variation with the continued shrinkage of the semiconductor process feature sizes, resulting in the low noise margin [10], the electromagnetic coupling effects [11], or crosstalk [12]. In addition, aging degradation also decrease the reliability of on-chip network [13]. These negative effects mostly generate stuck-at-1 and stuck-at-0 faults and eventually lead to unreliable network on chips [14]. These faults, considered as transient faults, which occur randomly and do not cause physical damages on circuits make fault detection and tolerant method become two main aspects attracting continuous researches.

In terms of fault detection in on-chip network, built-in-sift-test (BIST) a traditional solution in integrated circuits can be borrowed as a fault-detect scheme to handle the faults in routers, links and network interfaces. Nearly all the components of NoC can be checked by BIST and deactivated after claiming faulty. In [15], faults in crucial parts of on-chip router such as buffering devices and multiplexers are handled by an off-line BIST. More comprehensive fault-detect scheme including router and channel faults is proposed in [16] by using a fully distributed off-line BIST which also provides isolation strategy of the faulty components. However, disconnection of the test subjects is required in BIST and the original traffic flow is influenced which make reliability and performance hardly balanced. To reduce the impact during fault detection, more online fault detection schemes are under investigated [17–19].

After fault detection, we also need to tolerant the existing faults and one of the solutions is restructuring the on-chip network, such as router architecture and topology. Reconfigurable bidirectional channels are applied to avoid the traffic block of faulty channels in [20] named BFT-NoC which supports permanent and transient link faults. When suffering faults in crossbar, one of the crucial components of on-chip router, [21] proposes a solution using bus to connect between the input and output. A router architecture, ReliNoC, is proposed in [22] which utilizes the redundant components as the replacements of the faulty ones to withstand faults and hold the on-chip service. In [23], Vicis is presented combining different schemes as built-in-sift-test (BIST), Error Correcting Code (ECC), port-swapping mechanism and a crossbar bypass bus as a comprehensive scheme aiming at tolerating wear-out induced faults.

Besides, current efforts in literature also focus on tackling these problems on the routing-level using routing algorithm to tolerant faults [24]. In [25], the broken links can be alarmed and one link fault can be tolerant with mesh network in the proposed routing algorithm. Other faulty scenarios are largely based on on-chip router. When faults occur in NoC, one of the most popular solutions is disabling a router and disconnecting the core from the network when a fault is detected [26–29]. In [26], an algorithm is proposed to tolerate a single faulty router in the network without using virtual channels and its main idea is to route packets through a cycle-free contour surrounding a faulty router. Each router in the proposed scheme share information of the faulty status of eight direct and indirect neighbouring routers. The DBP approach [27] uses a default back-up path at each router to connect the upstream to the downstream router in the case of fault. Thereby, besides the underlying interconnection infrastructure, these backup paths connect all routers together in a form of a unidirectional cycle such as a ring. This algorithm is based on taking non-minimal routes. In [28,29], non-minimal routing algorithms have been investigated aiming to achieve fault-tolerant methods by enabling packets to turn around the faulty areas. Similarly, [30,31] design the fault-tolerant routing algorithms with the routing-level reconfiguration guiding the traffic bypass the faulty area and avoid the deadlock situations. Due to the fault-tolerant ability in non-minimal routing addition, it is widely investigated and a scheme named HiPFaR [32] targets tolerating faulty routers by avoiding non-minimal paths as along as possible seeking for better performance.

Although, considerable improvements of reliability have been achieved by routing packets away from the fault area, these proposed methods are based on a common assumption that the entire router is disabled in the case of a single fault. These kinds of solutions try to keep the network working by disconnecting both the faulty router and the core from the network and maintain the connectivity of the remaining nodes. In fact, this assumption is not realistic and may have a severe impact on the functionality of the entire system (i.e. the tasks of the disabled core cannot be transferred to the other cores) or the performance (i.e. traffic highly increases around the faulty area). NoRD [33], on the contrary, takes this issue into account where a separate virtual channel is used to connect all cores through a ring network. This approach provides a solution to connect the powered-off routers with the rest of network. However, as the network size increases, delay rises obviously which deteriorate the on-chip performance.

We narrow the fault-tolerant issue and focus on the faults in control logic of the routers which can be expressed as transient illegal turns. Since the faults occur in control logic and not the whole node, faulty router can be retained and the corresponding core is able to receiving and sending packets. The concept of turn in on-chip network is also showing in [34] where turn model is used to proof deadlock freeness. Some prohibitive turns following the rule of Odd-Even mechanism are supported in this proposal without virtual channels since the additional buffer is available in each router for store the wrong-turn packets. However, this solution still assume the whole router can take away which is a coarse-grained fault-tolerant solution and use the prohibitive turns to bypass the faulty area. Another weakness is that it only suitable for light-weighted applications because of the limited extra buffer.

In this paper, we provide a fine-grained solution and aim at tolerating faults expressed as transient illegal turns and the faulty router is not necessarily turning off. We provide more than one routing direction supported by the non-minimal routing to handle the misrouting situation and introduce a mechanism with the capability of fault detection, fault classification and fault tolerance. It is able to improve the reliability of on-chip network. By applying the proposed mechanism, latency of the network are close to those of using the DyXY routing algorithm [35] which is widely accepted and realized as a high-performance algorithm in NoC-based routing in the absent of faults. Similar to DyXY, only one additional virtual channel along the *Y* dimension is utilized in the proposed mechanism.

3. The proposed router architecture

The router architecture of ODT is shown in Fig. 1, sharing a similar structure as the traditional NoC router [5]. However, three new

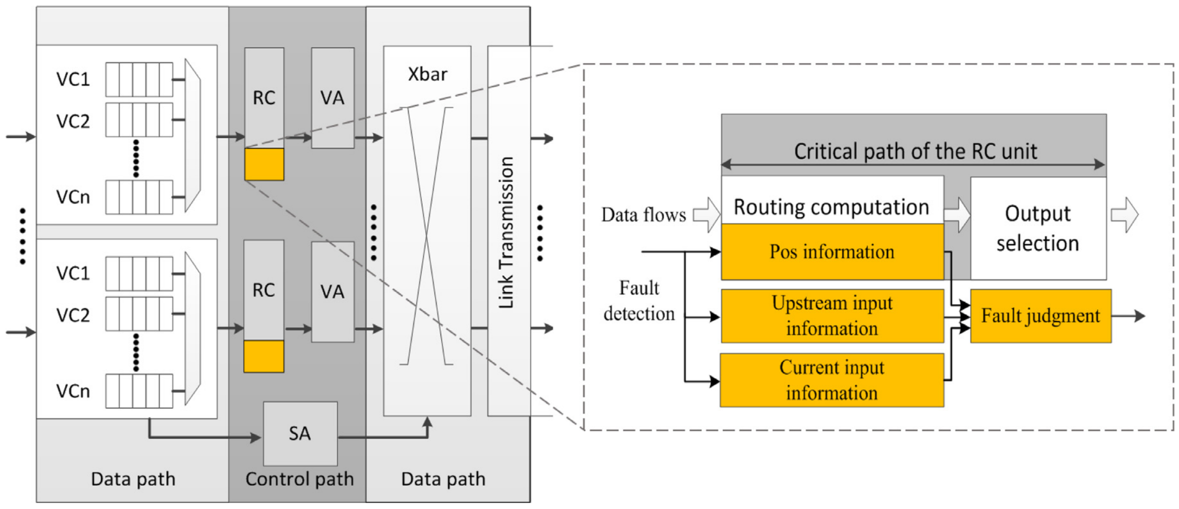


Fig. 1. The proposed router architecture. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

```
IF Pos = {L}                                      THEN   OutCh(L)<=’1’;
IF InCh != {E} AND Pos = {E or NE or SE}          THEN   OutCh(E)<=’1’;
IF InCh = {L or N1 or S1 or E}                    THEN   OutCh(W)<=’1’;
IF InCh = {L or S1 or E}                          THEN   OutCh(N1)<=’1’;
IF InCh = {L or N1 or E}                          THEN   OutCh(S1)<=’1’;
IF InCh != {N2} AND Pos={N or E or NE or SE}      THEN   OutCh(N2)<=’1’;
IF InCh != {S2} AND Pos={S or E or NE or SE}      THEN   OutCh(S2)<=’1’;
```

Fig. 2. The baseline non-minimal routing algorithm [36].

components are embedded in the proposed architecture to achieve efficient fault detection and tolerance. The components include the fault detection unit, the illegal-turn-resilient routing algorithm, and the fault classification unit. In this figure, areas marked as yellow indicate the requisites of fault detection as the Pos information (position between the current and destination node), upstream input information (input port of the previous node) and current input information (input port of the current node) while the fault judgment unit makes a decision whether the occurred fault in the previous router can be tolerated.

4. The proposed mechanism

4.1. The illegal-turn-resilient routing

4.1.1. The basic method of fault detection and tolerance

To tolerate illegal turns without disabling the router, we take advantage of a routing algorithm to provide adaptiveness in routing packets. For this purpose, we utilize a non-minimal routing algorithm proposed in [36], using two virtual channels along the *Y* dimension and no virtual channel in the *X* dimension. Thereby, there are 7 channels in each router called East (E), West (W), North1 (N1), North2 (N2), South1 (S1), South2 (S2), and Local (L). The basic routing algorithm is summarized as the pseudo-code in Fig. 2. As can be seen in this pseudo code, several output channels might be eligible to deliver a packet. For example, when the input port is N1 and the destination is in the NE quadrant, the possible output channels are as E, W, S1, N2, and S2. However, by default, the packet is sent either to the E or N2 according to the congestion information (i.e. the number of free buffer slots in the input ports). The options of output channels in non-minimal directions are utilized to tolerate faults. In other words, if because of faults, the packet is sent to a direction rather than the minimal ones, the packet can successfully reach the destination if the turn is among the supported turns in the non-minimal algorithm. Following the example (input port: N1 and destination: NE quadrant), if there is a fault in the routing unit and the packet is sent to the W direction which is not among minimal options, deadlock will not occur as the turn is still among the eligible non-minimal options. On the other hand, routing the packet to N1 may lead to the deadlock as it is not supported by the non-minimal algorithm. When on faults occur, packets will follow the minimal path to their destination as shown in Fig. 3.

In our previous work [9], the capability of the non-minimal routing algorithm has been investigated, showing that a faulty router can be prevented from being unnecessarily disabled and can continue functioning in the presence of faults. Based on the method presented in [9], all the possible connections between input and output ports are listed in Table 1. The entries marked with √ mean that these turns are supported by the baseline non-minimal routing algorithm and they will not cause any severe negative effects such as the deadlock and livelock. The entries marked × are not supported and thus taking these turns may lead to severe effects on the network. The remaining turns marked with ※ are conditional cases and the destination position is needed in order to (in)validate them. In this paper, we tend to improve the capability of this method to cover more illegal turns and provide a higher reliability. New features are introduced aiming at enhancing the robustness to cope with control path related problems.

4.1.1.1. Spare Routing. One of the potential problems in the basic routing algorithm is the absence of routing options under illegal turns when RC modules cannot come to a conclusion of forwarding direction. While taking an incorrect routing in upstream router, routing rule is broken and the resulting impact is brought to the current router. Packets will appear at the input ports which they

```
Selection strategy in the absence of faults following the minimal path
IF Pos = {L}                                   THEN    OutCh(L)<='1';
ELSE IF Pos = {E}                              THEN    OutCh(E)<='1';
ELSE IF Pos = {W}                              THEN    OutCh(W)<='1';
ELSE IF Pos = {S}
        IF Destination is in the west of Source   THEN    OutCh(S1)<='1';
        ELSE                                      THEN    OutCh(S2)<='1';
ELSE IF Pos = {N}
        IF Destination is in the west of Source   THEN    OutCh(N1)<='1';
        ELSE                                      THEN    OutCh(N2)<='1';
ELSE IF Pos = {NE}                             THEN    OutCh(E or N2)<='1';
ELSE IF Pos = {SE}                             THEN    OutCh(E or S2)<='1';
ELSE IF Pos = {SW}                             THEN    OutCh(W or S1)<='1';
ELSE IF Pos = {NW}                             THEN    OutCh(W or N1)<='1';
```

Fig. 3. The fully adaptive and minimal form of the algorithm, employed in fault-free networks [35].

Table 1
Fault-tolerant capability in baseline non-minimal routing algorithm.

From \ To	E	W	N1	N2	S1	S2
E	×	√	√	※	√	※
W	※	×	×	※	×	※
N1	※	√	×	※	√	※
N2	※	×	×	×	×	※
S1	※	√	√	※	×	※
S2	※	×	×	※	×	×
L	※	√	√	※	√	※

are not supposed to access. The reason of this phenomenon lies in Pos parameter that the position between current node and the destination must be check before routing.

After summarizing all these critical cases in Fig. 4 (where input port indicates which ports the packets are received from and the Pos shows the ideal direction of packets), the Spare Routing is designed to tackle this problem as shown in Fig. 5. When there is no routing option, the spare options are taking place of the default routing algorithm to ensure a fully functional on-chip network. By applying Spare Routing, more illegal marked ※ in Table 1 are supported in ODT.

By applying the Spare Routing showing in Table 2, 15 turns are supported regardless the position information. When packets utilize the Spare Routing, the input port will be blocked during these packets finishing their turns in the current router. The block will prevent the generation of deadlock by reserving an additional resource in the potential cycles.

4.1.1.2.. Shortest Path Priority. In the basic algorithm, illegal turns are forbidden in order to keep on-chip away from deadlock and livelock. However, this approach is rather conservative while some of these turns caused by the malfunction of RC modules will not lead to the persistent blocking of packets. In another word, packets which suffer from some illegal turns will not be involved in the same illegal paths repeatedly. Instead, packets are able to get back on track by taking advantage of the appropriate selection among various routing directions. In ODT, Shortest Path Priority is applied to guide packets which suffer from illegal turns to the correct paths to their destinations while the well-running packets can still stick to the basic routing algorithm.

The core idea of Shortest Path Priority is to choose one of the possible routings which is most closed to the shortest path and five more illegal turn can be supported as the turns from W to N_vc1, from W to S_vc1, from N_vc2 to W, from N_vc2 to S_vc1 and those turns from S_vc2 to N_vc1. Explanations related to these illegal turns have shown in detail in Figs. 6–10. It was noticed that the packet again come back to the same faulty route after taking illegal turn such as in Fig. 6, and will not suffer the previous illegal turn again. Packets can be normally farwarded by the functional conponents of the N_vc2 input in the faulty router since we only assume the failure in RC unit of W input. Similar situations can be found in Figs. 7, 8 and 10.

It should be noticed that the node marked "S" and "D" in figures represent the source nodes and the destinations respectively. The dark node means the router with malfunction in control path which will lead to illegal turns. Besides, there are 3 kinds of arrows which mean the illegal turn (red), the possible directions under the improved algorithm (yellow) and the forbidden selections (black). Integrating the basic routing algorithm, these forbidden turns can effectively prevent deadlock and livelock. Also, we can find dotted lines in these figures which play the role of boundaries, isolationg the possible destinations from other nodes (Figs. 11–13).

By comparing the fault-tolerant ability between the basic routing and the improved one in ODT, all the tolerable turns are listed in Table 3 which indicates more illegal turns are supported.

4.2. The online fault detection unit

The first step in the proposed algorithm is the fault detection which is done at the neighbouring routers of the faulty router. In the other words, upon receiving a packet at a router, it is checked

```
IF InCh = {S_vc2} AND Pos={W or NW or SW or S}      No RC Result;
IF InCh = {N_vc2} AND Pos={W or NW or SW }          No RC Result;
IF InCh = {W} AND Pos = {W or NW or SW or N}        No RC Result;
```

Fig. 4. Critical cases without routing options.

```
WHILE No RC Result;
    IF InCh = {S_vc2}        OutCh(N_vc1)<='1';
    IF InCh = {N_vc2}        OutCh(S_vc1)<='1';
    IF InCh = {W}            OutCh(N_vc1)<='1' or OutCh(S_vc1)<='1';
```

Fig. 5. Spare Routing algorithm.

Table 2
Fault-tolerant capability with Spare Routing under all turns.

To / From	E	W	N1	N2	S1	S2
E	×	√	√	√	√	√
W	√	×	×	√	×	√
N1	√	√	×	√	√	√
N2	√	×	×	×	×	√
S1	√	√	√	√	×	√
S2	√	×	×	√	×	×
L	※	√	√	※	√	※

From W to N_vc1 at the dark node

◆ Packets went E in previous step with InCh != E and Pos = E, NE, SE

◆ After taking illegal turn, packets can take N_vc2, S_vc2 or E. However, W will not be selected according to ***Shortest Path Priority.*** Also, turn S_vc1->W and S_vc1-> N_vc1 are forbidden in the north neighboring router of the dark node to prevent deadlock.

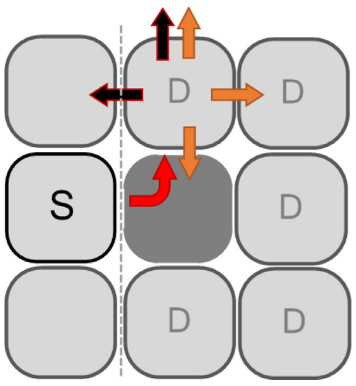


Fig. 6. Illegal turn from W to N_vc1. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

From W to S_vc1 at the dark node

◆ Packets went E in previous step with InCh != E and Pos = E, NE, SE

◆ After taking illegal turn, packets can take N_vc2, S_vc2 or E. However, W will not be selected according to ***Shortest Path Priority***. Also, turn N_vc1->W and N_vc1-> S_vc1 are forbidden in the south neighboring router of the dark node to prevent deadlock.

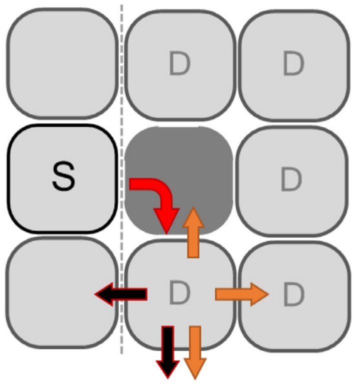


Fig. 7. Illegal turn from W to S_vc1. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

whether the upstream router has performed the routing correctly or not. However, no routing recalculation is done in the receiving router since the router computation module is reused.

Using three factors including the Pos, upstream input and current input information, ODT can judged whether a fault has occurred at the upstream router or not. This above-mentioned information can be simply transferred into 3-bit code and a lightweight hardware is required to run for it. In addition, this step can be done in parallel with the output selection part in the routing com-

From N_vc2 to S_vc1

◆ Packets went S_vc2 in previous step with InCh != S_vc2 and Pos = S, E, NE, SE (In fact, only those packets with InCh != S_vc2 and Pos = S, SE will choose S_vc2 while applying ***Shortest Path Priority***)

◆After taking illegal turn, packets can take N_vc2(packets can be forwarded to east in next step) or S_vc2. However W will not be selected according to ***Shortest Path Priority.*** Also, turn N_vc1->W and N_vc1-> S_vc1 are forbidden in the south neighboring router of the dark node to prevent deadlock.

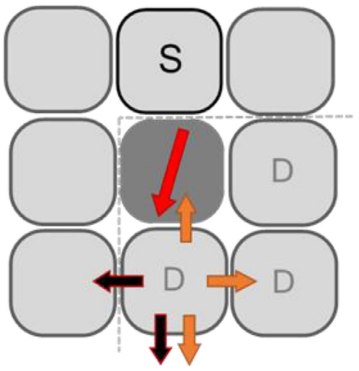


Fig. 8. Illegal turn from N_vc2 to S_vc1. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

putation unit, which is shown in Fig. 1. This implies that the fault detection circuit will not affect the critical path.

Based on the proposed illegal-turn-resilient routing, Fig. 14 shows the fault judgment circuit where a small piece of code is run for the packets received from each input port. Faults are classified into two categories, the ignorable fault and the severe fault which will be explained in the next section. The severe fault is the fault that cannot be supported by the basic routing algorithm. However, by taking advantage of ODT, these faults can be tolerance following the rule of *Spare Routing* as well as *Shortest Path Priority*.

4.3. The fault classification

We classify faults into two groups as ignorable faults and severe faults depending on whether the turn taken by a packet due to faults is among the allowable turns are not. According to the employed illegal-turn-resilient routing, the former can be tolerated as it has no negative effects on the functionality of the on-chip network while the latter cannot be tolerated and may lead to crashing the whole system.

The ignorable fault is the fault that can be tolerated by our proposed mechanism. When faults occur in control path, one of the most obvious cases is sending packets to wrong ports and eventually leads to deadlock and livelock in NoC platforms. Fortunately, by applying ODT, some illegal turns are still supported. The packet, suffering from ignorable faults, will experiences the wrong turns and eventually arrives at the right destination.

The severe fault is the fault that cannot be supported by our mechanism because of the rules of the routing algorithm in which some turns are not supported. For example, when a router receives a packet from its N1 port (i.e. the packet is sent by its north neighbour). If the packet arrives from the specific input ports, such as S1 of the upstream router, severe fault occurs. Under this circumstance, faults cannot be tolerated. However, the *Spare Routing* and

From N_vc2 to W

◆ Packets went S_vc2 in previous step with InCh != S_vc2 and Pos = S, E, NE, SE (In fact, only those packets with InCh != S_vc2 and Pos = S, SE will choose S_vc2 while applying ***Shortest Path Priority***)

◆ After taking illegal turn, packets can take S_vc1, or S_vc2 (packets can be forwarded to east in next step) . However, other ports will not be selected according to ***Shortest Path Priority***. Also, turn E->W, E->N_vc1, E->N_vc2 and E-> S_vc1 are forbidden in the west neighboring router of the dark node to prevent deadlock

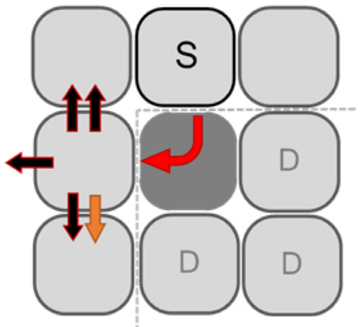


Fig. 9. Illegal turn from N_vc2 to W. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

From S_vc2 to N_vc1

◆ Packets went N_vc2 in previous step with InCh != N_vc2 and Pos = N, E, NE, SE (In fact, only those packets with InCh != N_vc2 and Pos = N, NE will choose N_vc2 while applying ***Shortest Path Priority***)

◆ After taking illegal turn, packets can take N_vc2, E or S_vc2(packets can be forwarded to east in next step). However W will not be selected according to ***Shortest Path Priority.*** Also, turn S_vc1->W and S_vc1-> N_vc1 are forbidden in the north neighboring router of the dark node to prevent deadlock.

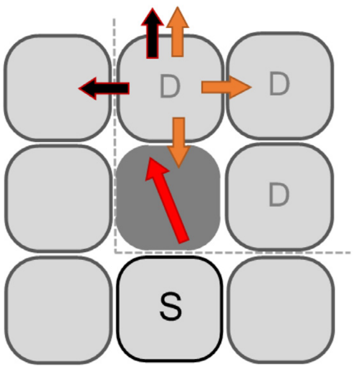


Fig. 10. Illegal turn from S_vc2 to N_vc1. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

From S_vc2 to W

◆ Packets went N_vc2 in previous step with InCh != N_vc2 and Pos = N, E, NE, SE (In fact, only those packets with InCh != N_vc2 and Pos = N, NE will choose N_vc2 while applying ***Shortest Path Priority***)

◆ After taking illegal turn, packets can take N_vc2. However other ports will not be selected according to ***Shortest Path Priority.*** Also, turn E->W, E->N_vc1, E->S_vc1 and W-> S_vc2 are forbidden in the west neighboring router of the dark node to prevent deadlock.

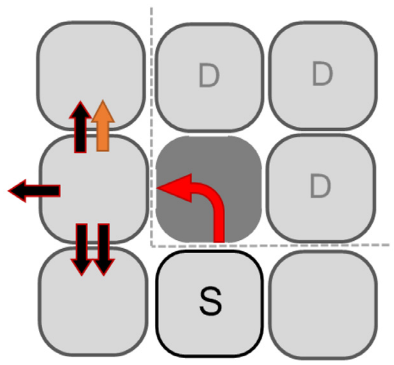


Fig. 11. Illegal turn from S_vc2 to W. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

the *Shortest Path Priority* can be applied in ODT to lower the probability of deadlock significantly.

5. Result and discussion

In this section, average latency, reliability and hardware overhead are evaluated. The average latency is defined as the average time takes for packets to reach from a source node to a destination node while reliability is indicated as the survival rate of injected packets which defined as the ratio of the number of packets successfully reach the destinations over the total number of injected packets.

5.1. Simulation platform

The SimpleScalar [37] which is a wide range platform supported simulator providing architectural modelling is served for our simulation to develop a NoC based multiprocessor architecture. In this simulation platform, 64 processors are connected by the on-chip network with 32KB L1 and 8MB L2 memory. In the aspect of on-chip network, POPNET, a cycle-accurate NoC simulator implemented with C++ is integrated in our simulation platform. 2D 8×8 mesh network is selected as the topology with wormhole switching and packets share the constant size containing 4 flits. The architecture of on-chip router is shown in Fig. 1 where 7 ports can be found as East, South_vc1, South_vc2, West, North_vc1, North_vc2 and Local. Two virtual channels are shared with the same physical channel along the north-south direction. To evaluate the performance and reliability of the proposed mechanism, 4 traffic patterns (Uniform, Transpose1, Transpose2, and Shuffle traffic) and Parsec benchmark (Blackscholes, Canneal, Dedup and Ferret) are running in our platform. Please note that the difference between Transpose1 and Transpose2 is in the orientation of choosing source and destination nodes. Because of the relatively low packet

From N_vc2 to N_vc1

◆ Packets went S_vc2 in previous step with InCh != S_vc2 and Pos = S, E, NE, SE (In fact, only those packets with InCh != S_vc2 and Pos = S, SE will choose S_vc2 while applying ***Shortest Path Priority***)

◆ After taking illegal turn, packets can take E ports. However, other ports will not be selected according to ***Shortest Path Priority***. Also, turn S_vc1->N_vc1, S_vc1->N_vc2, S_vc1->W and S_vc1->S_vc2 are forbidden in the source router.

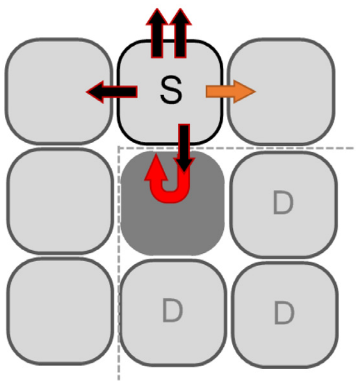


Fig. 12. Illegal turn from N_vc2 to N_vc1. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

From S_vc2 to S_vc1

◆ Packets went N_vc2 in previous step with InCh != N_vc2 and Pos = N, E, NE, SE (In fact, only those packets with InCh != N_vc2 and Pos = N, NE will choose N_vc2 while applying ***Shortest Path Priority***)

◆ After taking illegal turn, packets can take E. However other ports will not be selected according to ***Shortest Path Priority.*** Also, turn N_vc1->S_vc1, N_vc1->S_vc2, N_vc1->W and N_vc1->N_vc2 are forbidden in the Source router.

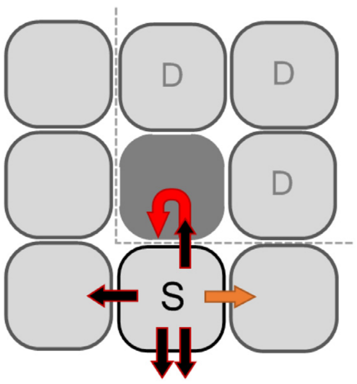
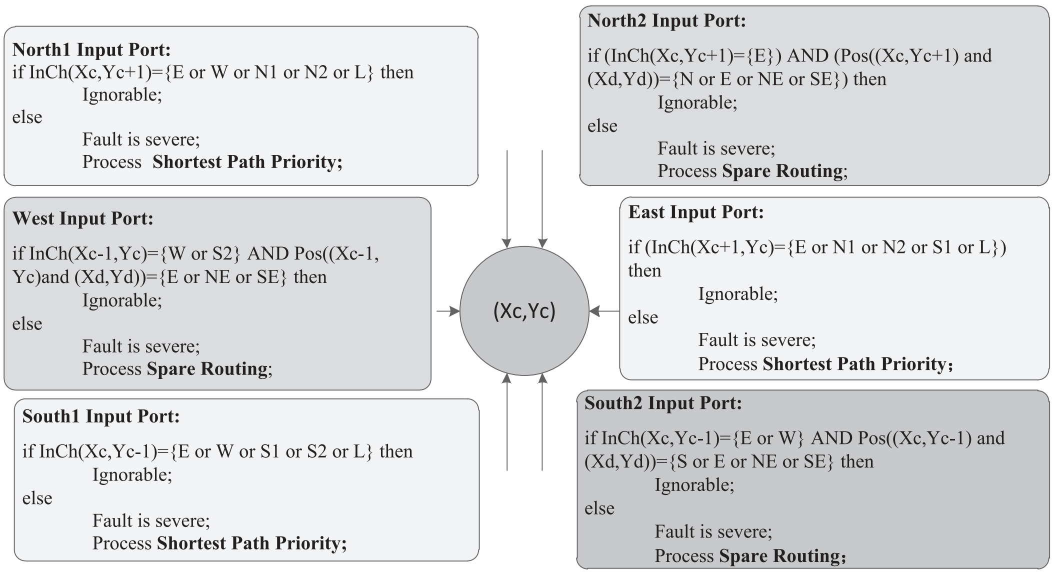


Fig. 13 Illegal turn from S_vc2 to S_vc1

Fig. 13. Illegal turn from S_vc2 to S_vc1. (For interpretation of the references to colour in this figure, the reader is referred to the web version of this article.)

Table 3
Fault-tolerant ability in ODT under all turns.

From \ To	E	W	N1	N2	S1	S2
E	×	√	√	√	√	√
W	√	×	√	√	√	√
N1	√	√	×	√	√	√
N2	√	√	√	×	√	√
S1	√	√	√	√	×	√
S2	√	√	√	√	√	×
L	※	√	√	※	√	※



```
North1 Input Port:
if InCh(Xc,Yc+1)={E or W or N1 or N2 or L} then
        Ignorable;
else
        Fault is severe;
        Process Shortest Path Priority;
```

```
North2 Input Port:
if (InCh(Xc,Yc+1)={E}) AND (Pos((Xc,Yc+1) and
(Xd,Yd))={N or E or NE or SE}) then
        Ignorable;
else
        Fault is severe;
        Process Spare Routing;
```

```
West Input Port:
if InCh(Xc-1,Yc)={W or S2} AND Pos((Xc-1,
Yc)and (Xd,Yd))={E or NE or SE} then
        Ignorable;
else
        Fault is severe;
        Process Spare Routing;
```

```
East Input Port:
if (InCh(Xc+1,Yc)={E or N1 or N2 or S1 or L})
then
        Ignorable;
else
        Fault is severe;
        Process Shortest Path Priority;
```

```
South1 Input Port:
if InCh(Xc,Yc-1)={E or W or S1 or S2 or L} then
        Ignorable;
else
        Fault is severe;
        Process Shortest Path Priority;
```

```
South2 Input Port:
if InCh(Xc,Yc-1)={E or W} AND Pos((Xc,Yc-1) and
(Xd,Yd))={S or E or NE or SE} then
        Ignorable;
else
        Fault is severe;
        Process Spare Routing;
```

Fig. 14. Algorithm to be run at each port for fault detection.

injection rate in Parsec and inconspicuous increase of latency, we only test four traffic patterns in performance evaluation section.

DyXY is selected as the baseline which is a fully adaptive routing algorithm widely accepted by literature and easy to implement in various platforms. DyXY can fit in our platform perfectly since it also supports two virtual channels along the *Y* dimension. By considering the congestion condition in the neighbour node, DyXY dynamically chose the forwarding direction and form a minimal path for each packet to their destinations. By evaluating the performance, running DyXY can achieve the optimal performance in the same network [24]. Also, ODT is compared with DMT which is an illegal turn tolerant mechanism in our previous work [9] to show the further effort achieve in this paper.

5.2. Fault injection scheme

ODT introduces an innovative way of tolerating faults at the control paths of the routers. It implies that the fault injection method is also different from other approaches. In traditional methods, faults are injected directly to the routers by disabling them while in this paper, we simulate the faults which occur in the control path of a router and express as illegal turns taking by packets. Thereby, we force packets to take a random turns despite the decision of the routing computation unit to create faults. This operation is similar to the behaviours when a fault occurs in the circuit level of control logic such as stuck-at-1 and stuck-at-0 and eventually allocates the wrong direction leading to illegal turns. In this simulation platform, 64 routers are involved and there are total 864 RC modules in the whole network (where the routers located at the corner own 3 RC modules, those located along the borderline own 4 or 5 RC module and other have 7 RC modules). We first generate 50 sets of fault pattern and the number of fault on each set is determined by the fault percentage during simulation and these faults are directly injection to the corresponding RC module by referring the router ID and port ID in simulator. When we launch the test, 50 sets of fault pattern are successively loaded by simulator to present a dynamic fault injection scheme since network during each test will suffer from 50 different fault patterns.

5.3. Performance evaluation

In this section, average latency is evaluated which is defined as the average time takes for packets to reach from a source node to a destination node. The performance of ODT, DMT are compared with each other from Figs. 15–17 under 4 traffic patterns The packet injection rates are indicated along the *X* axis while the *Y* dimension shows the average latency.

The performance comparison is shown in Fig. 15 when there is no fault in the whole network. As can be seen in this figure, the average latency of the proposed mechanism as well as the DMT method is nearly the same as those using DyXY. The reason is

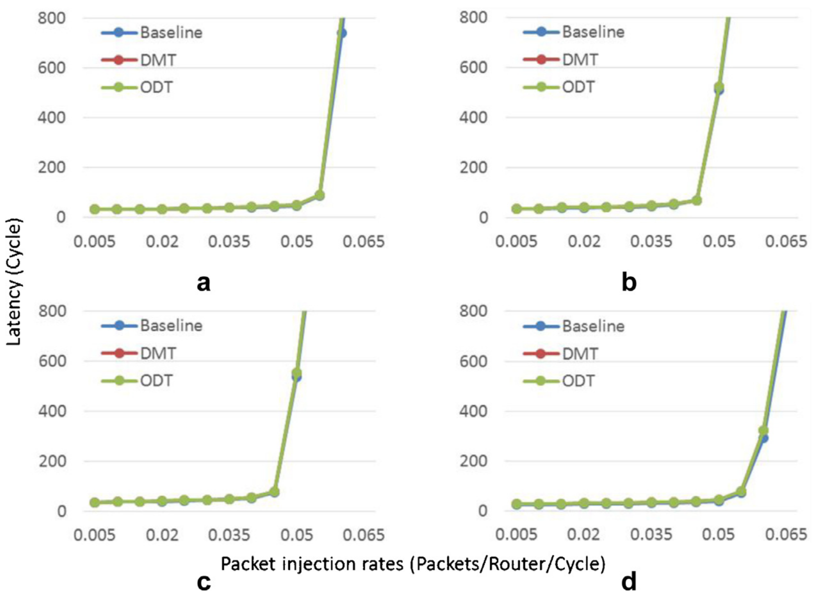


Fig. 15. The latency of ODT and DMT in fault-free cases under four traffic patterns as (a) Uniform, (b) Transpose1, (c) Transpose2 and (d) Shuffle while DyXY without faults is shown as baseline.

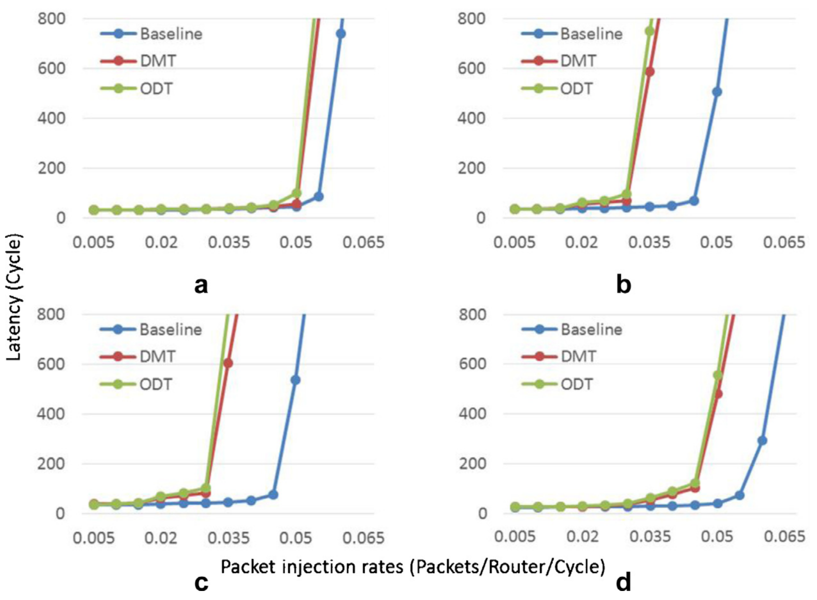


Fig. 16. The latency of ODT and DMT with the fault injection of 2% under four traffic patterns as (a) Uniform traffic, (b) Transpose1 traffic, (c) Transpose2 traffic and (d) Shuffle while DyXY without faults is shown as baseline.

that, in fault-free conditions, all methods follow the minimal paths. Fig. 16 shows the performance results when 2% of router computation modules are faulty in the network. By comparing the equivalent curves of Figs. 15 and 16, it can be observed that the saturation point of the ODT and DMT method shift forward alone the direction of the negative axis. This is because illegal turns are tolerated in these methods and additional non-minimal paths are taken by packets which increase the average latency. We can also find that the ODT will cost slightly more time to finish the packets transmission when compared to the DMT because of more packets that are surviving.

In order to observe the impact of a higher injection rate, we increase the fault injection rate to 4% in Fig. 17. The results show that a little backward of the performance both in ODT and DMT and the difference between these methods meet a further increase. On the other hand, the performance of ODT keeps declining, implying that more packets have saved in the network.

5.4. Fault-tolerant capability

In this set of experiment, we evaluate the reliability of the network for different fault injection percentage and for different traffic patterns and Parsec benchmark. The fault injection percentages are selected between 0% and 4% which means 4% of RC modules are fault in the worst case. Since the number of dropped packets exceeds 10% of the total number of packets, we stop the fault injection percentage at 4%. In fact, this is very severe situation when 4% of the fault happen in network-on-chip which means the processor element are suffering from the same level of fault and that is devastating where processor are hard to remain functional.

The results of running traffic patterns including Uniform, Transpose1, Transpose2, and Shuffle traffic are shown in Figs. 18–21. The statistical information in these experiments includes the total number of packets which can be successfully received by the destination (shown along the *Y*-axis) and the fault injection per-

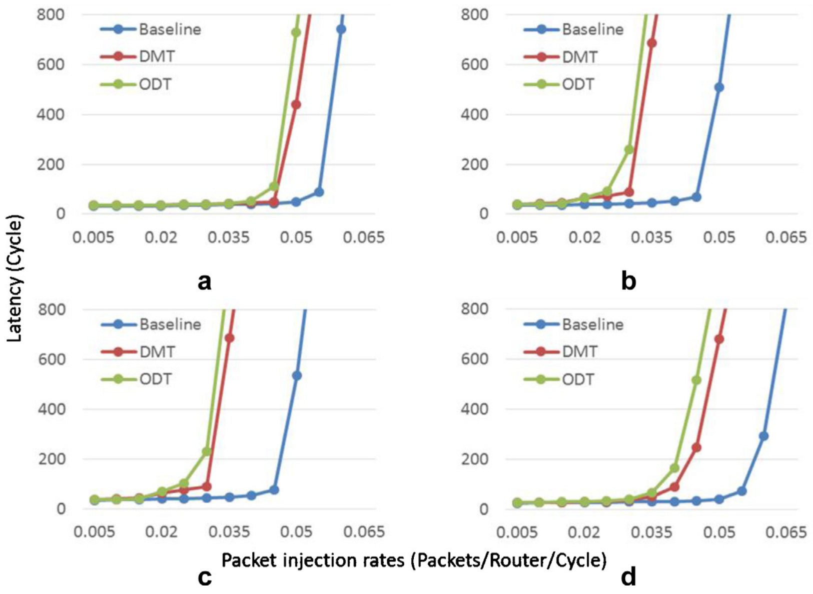


Fig. 17. The latency of ODT and DMT methods with the fault injection of 4% under four traffic patterns as (a) Uniform traffic, (b) Transpose1 traffic, (c) Transpose2 traffic and (d) Shuffle while DyXY without faults is shown as baseline.

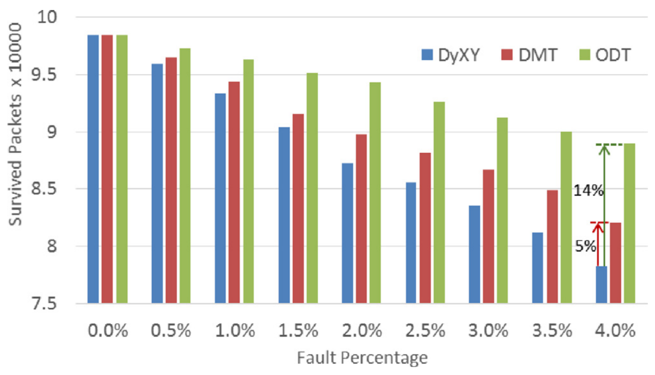


Fig. 18. Fault-tolerant capability under uniform traffic.

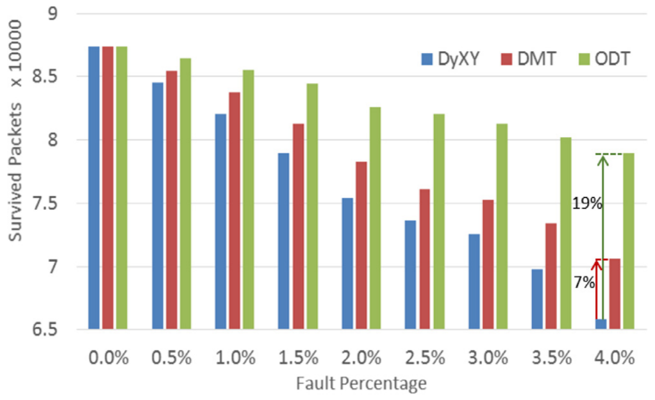


Fig. 19. Fault-tolerant capability under Transpose1 traffic.

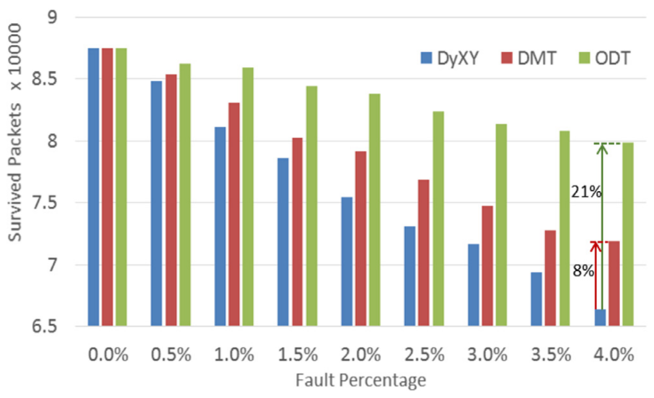


Fig. 20. Fault-tolerant capability under Transpose2 traffic.

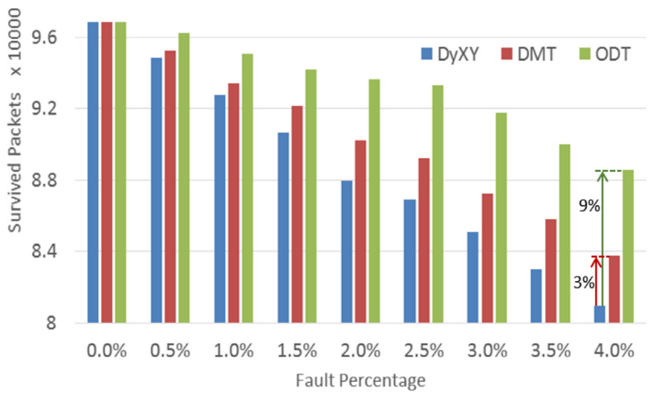


Fig. 21. Fault-tolerant capability under Shuffle traffic.

centages (from 0% to 4% along the *X*-axis). Compared to the opponents, ODT has better capability to tolerate faults in the control path without the prior knowledge on where and why a fault has happened. As can be seen from Figs. 18–21, ODT is able to survive 10,000 more packets than the DyXY algorithm with 4% fault injections. More specifically, 14%, 19%, 21% and 9% growth of surviving packets are achieved in ODT respectively under all four traffic patterns. The contrast between the previous work in DTM and the proposed ODT also indicates the obvious improvement which achieve in ODT. It also means that the deadlock is rarely happen in the on-chip network with ODT.

When evaluating the performance with the Parsec benchmark program, we get the similar trend between the fault percentage and the number of survived packets. Blackscholes, Canneal, Dedup and Ferret benchmarks are shown in Figs. 22–25, where in our

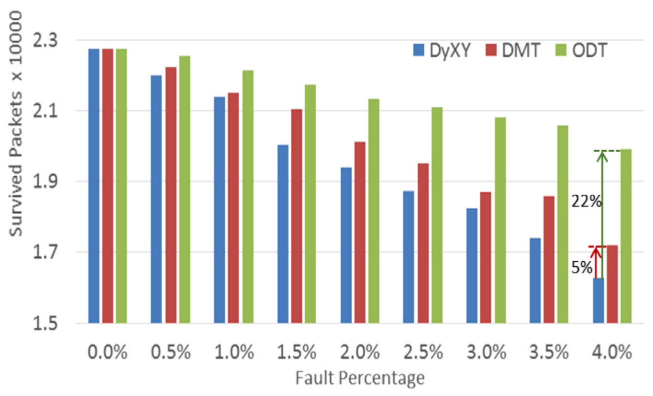


Fig. 22. Fault-tolerant capability under Blackscholes in Parsec.

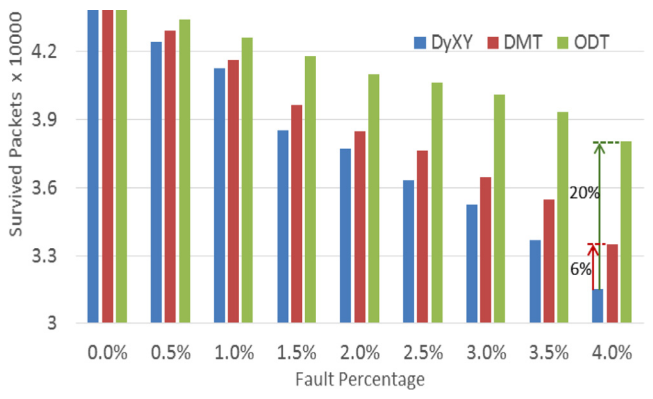


Fig. 23. Fault-tolerant capability under Canneal in Parsec.

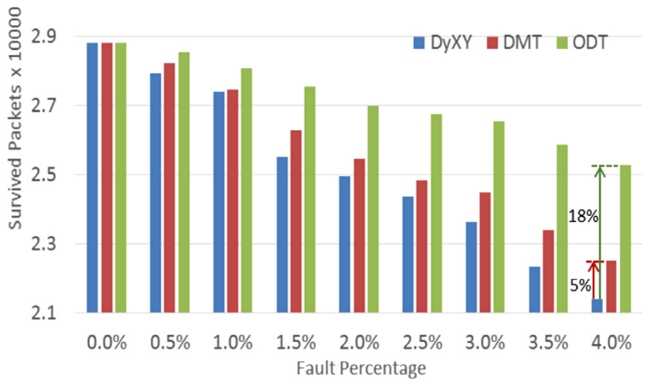


Fig. 24. Fault-tolerant capability under Dedup in Parsec.

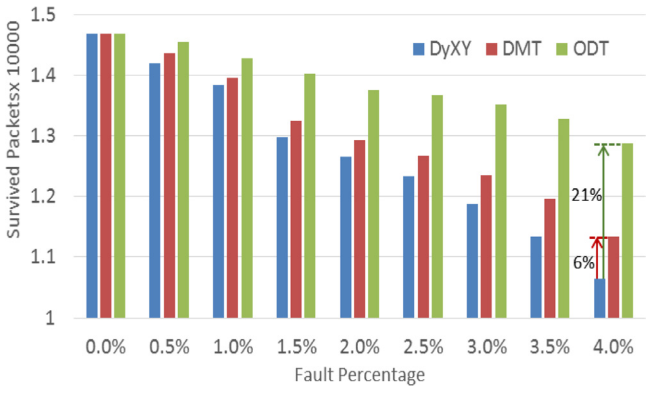


Fig. 25. Fault-tolerant capability under Ferret in Parsec.

Table 4
Area overhead and power consumption.

	Router with DyXY	Router with DMT	Router with ODT
Cell area (μm^2)	1848.46	1961.23	2019.02
Cell power (μW)	264.56	282.73	290.04
vs in area (%)	+0.00	+6.01	+9.22
vs in power (%)	+0.00	+6.87	+9.63

proposed ODT, 22%, 20%, 18% and 21% growth of surviving packets are achieved.

5.5. Hardware overhead

To assess the area overhead and power consumption, an on-chip network router with ODT, DMT and a default router using DyXY algorithm are synthesized using a Synopsys Design Compiler. We compared the area overhead and power consumption of a router in these models. For synthesizing, we use the TSMC45nm technology at the operating frequency of 1 GHz and supply voltage of 0.9 V. As indicated in Table 4, the power consumption and area overhead of these routers are comparable. With additional 9.22% of area and 9.63% of power overhead, the proposed algorithm can offer a more reliable no-chip network.

6. Conclusion

We proposed a mechanism to tolerate faults which can be expressed as illegal turns in the Network-on-Chip, called ODT. Based on the online fault detection unit, the illegal-turn-resilient routing and the fault classification, ODT is able to achieve much higher reliability while the extra hardware is lightweight and do not affect the critical path of the router. Also, routers are not necessary disabled when faults occur. In this paper, ODT introduces a new way of employing non-minimal algorithms in NoCs which is not simply circumventing the faulty zone but taking a better advantage of them by increasing the possible routing selections. According to the experimental results, the average latency of transmission in the proposed mechanism is almost the same as DyXY routing when no faults occur. The deadlock situations represented by illegal turns can be significantly reduced and more than 14%, 19%, 21% and 9% of packets are survived in the network under 4 traffics respectively and average 20% more packets can be survived in four different Parsec benchmarks with 4% of RC failure.

Acknowledgement

This work is supported by the NSFC under Grant nos. 61176025, 61006027 and the Oversea Academic Training Funds (OATF), UESTC. It is also supported by VINNOVA-MarieCurie within the ERoT project and Academy of Finland.

References

[1] A. Jantsch, H. Tenhunen, Networks on Chip, vol. 396, Kluwer Academic Publishers, Dordrecht, 2003.
[2] J.D. Owens, et al., Research challenges for on-chip interconnection networks, IEEE Micro 27 (5) (2007) 96–108.
[3] X. Fu, T. Li, J.A. Fortes, Architecting reliable multi-core network-on-chip for small scale processing technology, in: Proceedings of IEEE IFIP International Conference on Dependable Systems and Networks (DSN), 2010.
[4] K. Bhardwaj, K. Chakraborty, S. Roy, Towards graceful aging degradation in NoCs through an adaptive routing algorithm, in: Proceedings of ACM/EDAC/IEEE Design Automation Conference (DAC), 2012.
[5] J. Howard, et al., A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and DVFS for performance and power scaling, IEEE J. Solid-State Circuits 46 (1) (2011) 173–183.
[6] A. Ganguly, P.P. Pande, B. Belzer, Crosstalk-aware channel coding schemes for energy efficient and reliable NOC interconnects, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 17 (11) (2009) 1626–1639.

[7] L.K. Loo, et al., Packet logging mechanism for adaptive online fault detection on network-on-chip, in: Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), 2014.
[8] G. Schley, N. Batzolis, M. Radetzki, Fault localizing end-to-end flow control protocol for networks-on-chip, in: Proceedings of IEEE Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2013.
[9] X. Zhang, M. Ebrahimi, L. Huang, G. Li, A. Jantsch, A network-level solution for fault detection, masking, and tolerance in NoCs, in: Proceedings of IEEE Euromicro International Conference on Parallel, Distributed and Network-Based Computing, (PDP), 2015.
[10] S. Yang, M. Greenstreet, Noise margin analysis for dynamic logic circuits, in: Proceedings of International Conference on Computer Aided Design (ICCAD), 2005.
[11] I. Erdin, M.S. Nakhla, R. Achar, Circuit analysis of electromagnetic radiation and field coupling effects for networks with embedded full-wave modules, IEEE Trans. Electromag. Compat. 42 (4) (2000) 449–460.
[12] R. Arunachalam, A novel algorithm for testing crosstalk induced delay faults in VLSI circuits, in: Proceedings of International Conference on VLSI Design, 2005.
[13] D.M. Ancajas, K. Bhardwaj, K. Chakraborty, S. Roy, Wearout resilience in NoCs through an aging aware adaptive routing algorithm, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 23 (2) (2015) 369–373.
[14] P.P. Pande, C. Grecu, R. Saleh, G. Demicheli, Design, synthesis, and test of networks on chips, IEEE Des. Test Comput. 22 (5) (2005) 404–413.
[15] S. Lin, W. Shen, C. Hsu, C. Chao, A. Wu, Fault-tolerant router with built-in self-test/self-diagnosis and fault-isolation circuits for 2Dmesh based chip multiprocessor systems, in: Proceedings of International Symposium on VLSI-DAT, 2009, pp. 72–75.
[16] Z. Zhang, et al., On-the-field test and configuration infrastructure for 2-D-mesh NoCs in shared-memory many-core architectures, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 22 (6) (2014) 1364–1376.
[17] J. Liu, et al., Online fault detection for networks-on-chip interconnect, in: Proceedings of IEEE NASA/ESA Conference on Adaptive Hardware and Systems (AHS), 2014.
[18] M.R. Kakoee, V. Bertacco, L. Benini, A distributed and topology-agnostic approach for on-line NoC testing, in: Proceedings of the 5th IEEE/ACM International Symposium on Networks on Chip, 2011.
[19] P.S. Bhojwani, R.N. Mahapatra, Robust concurrent online testing of network-on-chip-based SoCs, IEEE Trans. Very Large Scale Integr. (VLSI) Syst. 16 (9) (2008) 1199–1209.
[20] Wen-Chung Tsai, et al., A fault-tolerant NoC scheme using bidirectional channel, in: Proceedings of the 48th Design Automation Conference, ACM, 2011.
[21] A. DeOrio, D. Fick, V. Bertacco, D. Sylvester, D. Blaauw, J. Hu, G. Chen, A reliable routing architecture and algorithm for NoCs, IEEE Trans. Computer-Aided Des. Integr. Circuits Syst. 31 (5) (2012) 726–739.
[22] M.R. Kakoee, V. Bertacco, L. Benini, ReliNoC: A reliable network for priority-based on-chip communication, in: Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011.
[23] D. Fick, et al., Vicis: a reliable network for unreliable silicon, in: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2009.
[24] M. Palesi, M. Daneshtalab, Routing Algorithms in Networks-on-Chip, Springer, 2014.
[25] H.S. Mehrizi, E. Zeinali, A load balancing method for improving fault tolerance in mesh based networks on chip, in: Proceedings of the 7th International Conference on Application of Information and Communication Technologies (AICT), 2013.
[26] Z. Zhang, A. Greiner, S. Taktak, A reconfigurable routing algorithm for a fault-tolerant 2D-mesh network-on-chip, in: Proceedings of ACM/IEEE Design Automation Conference (DAC), 2008.
[27] M. Koibuchi, H. Matsutani, H. Amano, T.M. Pinkston, A lightweight fault-tolerant mechanism for network-on-chip, in: Proceedings of ACM/IEEE International Symposium on Networks-on-Chip, 2008.
[28] F. Chaix, D. Avresky, N.E. Zergainoh, M. Nicolaidis, A fault-tolerant deadlock-free adaptive routing for on chip interconnects, in: Proceedings of Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011.
[29] M. Ebrahimi, M. Daneshtalab, J. Plosila, H. Tenhunen, MAFA: adaptive fault-tolerant routing algorithm for networks-on-chip, in: Proceedings of Euromicro Conference on Digital System Design (DSD), 2012.
[30] D. Fick, A highly resilient routing algorithm for fault-tolerant NoCs, in: Proceedings of ACM/IEEE Conference on Design, Automation and Test in Europe (DATE), 2009, pp. 21–26.
[31] S. Rodrigo, et al., Addressing manufacturing challenges with cost-efficient fault tolerant routing, in: Proceedings of ACM/IEEE International Symposium on Networks-on-Chip (NoCS), 2010, pp. 25–32.
[32] M. Ebrahimi, M. Daneshtalab, J. Plosila, High performance fault-tolerant routing algorithm for NoC-based many-core systems, in: Proceedings of Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2013.
[33] L. Chen, T.M. Pinkston, NoRD: node-router decoupling for effective power-gating of on-chip routers, in: Proceedings of IEEE/ACM International Symposium on Microarchitecture, 2012.
[34] Wen-Chung Tsai, et al., TM-FAR: turn-model based fully adaptive routing for networks on chip, in: Proceedings of VLSI System on Chip Conference (VLSI-SoC), 2010.
[35] M. Li, Q.A. Zeng, W. B Jone, DyXY – a proximity congestion-aware deadlock-free dynamic routing method for network on chip, in: Proceedings of Design Automation Conference, 2006.
[36] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, H. Tenhunen, LEAR – a low-weight and highly adaptive routing method for distributing congestions in on-chip networks, in: Proceedings of Euromicro International Conference on Parallel, Distributed and Network-Based Processing (PDP), 2012.
[37] T. Austin, E. Larson, D. Ernst, SimpleScalar: an infrastructure for computer system modeling, Computer 35 (2) (2002) 59–67.

Letian Huang was born in Sichuan Province, China, in 1984. He received the B.S. and M.S. degrees from the University of Electronic Science and Technology of China (UESTC), Chengdu, where he is currently working toward the Ph.D. degree. He received B.S. and M.S. in 2006 and 2009, in Communication and Information System. From 2013 to 2014 he is a visiting researcher in ICT Royal Institute of Technology (KTH), Sweden. He is also a Lecturer of UESTC. His research interests include power efficient computing and communication system, mult-core/many-core embedded system for signal processing, and mixed signal IC design.

Xiaofan Zhang was born in Guangzhou, China in 1990. He received the B.S. degree in information engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu in 2013. He is currently pursuing the M.S. degree with the Department of Communication and Information Engineering, UESTC. His research interests include reliability of network-on-chip and parallel and network-based computing.

Masoumeh Ebrahimi received the Ph.D. degree with honors from the University of Turku, Finland, with over 60 high-level journal and conference paper publications, including three book chapters. Her areas of expertise include interconnection networks, networks-on-chip, 3D integrated systems, and systems-on-chip. During her doctoral studies, she received a 4-year scholarship from the Graduate School in Electronics, Telecommunications and Automation (GETA). In addition, she has received grant awards from the Elisa foundation (2011), Kaute foundation (2013), University of Turku foundation (twice 2011 and 2013), and Nokia foundation (twice 2012 and 2013). She also received a 1-year grant for doing postdoctoral research abroad from the TES foundation. Currently, she holds a VINNOVA Marie-Curie fellowship, conducting her research ICT Royal Institute of Technology (KTH), Sweden and the University of Turku, Finland, as a senior researcher. She is a member of the IEEE.

Guangjun Li received M.S. degree from the University of Electronic Science and Technology of China (UESTC), Chengdu, in 1985. Since then, he has been with UESTC. From 1991 to 1992, he was a visiting scholar with RETH Aachen University, Aachen, Germany. He is currently the Chair of the Communication Integrated Circuits and Systems Engineering Center of UESTC. He has published various papers in the area of communication systems, wireless communication networks, SoC and NoC for wireless communication systems.