Minimal-Path Fault-Tolerant Approach Using Connection-Retaining Structure in Networks-on-Chip

Masoumeh Ebrahimi, Masoud Daneshtalab, Juha Plosila, Hannu Tenhunen
Department of Information Technology, University of Turku
{masebr, masdan, juplos, hanten}@utu.fi

Abstract—**There are many fault-tolerant approaches presented both in off-chip and on-chip networks. Regardless of all varieties, there has always been a common assumption between them. Most of all known fault-tolerant methods are based on rerouting packets around faults. Rerouting might take place through non-minimal paths which affect the performance significantly not only by taking longer paths but also by creating hotspot around a fault. In this paper, we present a fault-tolerant approach based on using the shortest paths. This method maintains the performance of Networks-on-Chip in the presence of faults. To avoid using non-minimal paths, the router architecture is slightly modified. In the new form of architecture, there is an ability to connect the horizontal and orthogonal links of a faulty router such that healthy routers are kept connected to each other. Based on this architecture, a fault-tolerant routing algorithm is presented which is obviously much simpler than traditional fault-tolerant routing algorithms. According to this algorithm, only the shortest paths are used by packets in the presence of fault. This results retains the performance of NoCs in faulty situations. This algorithm is highly reliable, for an instance, the reliability is more than 99.5% when there are six faulty routers in an 8×8 mesh network.**

I. Introduction

As the number of processing elements integrated into a single chip is increasing, traditional bus-based architectures in Many-Core Systems-on-Chip (MCSoCs) are inefficient and a new communication infrastructure is needed. Networks-on-Chip (NoC) has emerged as a promising solution for on-chip interconnection in MCSoCs due to its scalability, reusability, flexibility, and parallelism [1][2][3].

Transient and permanent faults are two different types of faults that can occur in on-chip networks [4]. Transient faults are temporary and unpredictable. They are often difficult to be detected and corrected. Permanent faults are caused by physical damages such as manufacturing defects and device wear-out. These faults should be recovered or tolerated in a way that the network continues functioning. Routing techniques provide some degrees of fault tolerance in NoCs. They can be categorized into deterministic and adaptive approaches [5][6]. 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 traversing from a source to a destination router. So, they can decrease the probability of routing packets through congested areas and thus improve the performance. In minimal adaptive routing algorithms, the shortest paths are used for transmitting messages between routers. In contrast, performance can severely deteriorate in non-minimal methods due to taking longer paths by packets. Moreover, non-minimal methods are usually more complex with a larger number of virtual channels. Deadlock-free non-minimal routing schemes are the most common approaches to tolerate faults in the network [7]. They are used in order to reroute packets around faults [8]. Some fault-tolerant algorithms are proposed to support special cases of faults, such as one-faulty routers, convex or concave regions. These algorithms either disable the healthy components or require a large number of virtual channels to avoid deadlock. There are other fault-tolerant approaches [9][10][11] which do not require any virtual channels. However, these algorithms are partially adaptive and very limited in supporting faults. In addition, fault-tolerant algorithms are relatively complex due to considering different fault models in order to find a path for a packet. A common behavior in fault-tolerant approaches is that packets are routed normally in the network until they face a fault. At this time, turn models or other techniques are used to reroute packets around the faults such that no cycles be formed in the network. The performance analysis in [11] indicates that in a 4×4 mesh network, the average packet latency can be increased by almost twice when a single router is faulty in the central part of the network. This is due to the fact that the latency is usually forgotten when designing a fault-tolerant algorithm. Faults might exist forever or it might be recovered after a long period of time. Now, imagine how many packets should be misrouted around the fault and how it can affect the performance. What if there are several faults in the network, existing for a long time?

In this paper, we present a method named MiCoF (**Mi**nimal-path **Co**nnection-retaining **F**ault-tolerant approach). It is a lightweight fault-tolerant approach that maintains the performance of NoCs in the presence of faulty routers. In this approach, packets are never misrouted around the faulty routers since an alternative path is chosen prior reaching the fault. To keep the connectivity and avoid misrouting packets, when a router becomes faulty, the belonging links will be connected in appropriate directions. In other words, a faulty router can be seen as a wire, connecting the surviving routers to each other. The distinguishing feature of MiCoF from the other proposals is in considering the performance as the main goal besides tolerating faults.

The rest of this paper is organized as follows: Section II reviews the related work. Preliminaries are given in Section III. The proposed fault-tolerant routing algorithm is presented in Section IV. Multiple faulty cases are discussed in Section V. The results are investigated in Section VI while we summarize and conclude in the last section.

II. RELATED WORK

A number of studies present solutions to tolerate faulty links [12][13] or routers [11][14] in the network. A large numbers of faults can be supported by the proposed method in [15] without using virtual channels. However, this method takes advantage of a routing table at each router and an offline process to fill out the tables. FTDR [16] and HARAQ [17] are Q-learning approaches requiring routing tables. The routing tables in HARAQ are used to collect the congestion information while the routing tables in FTDR indicate the possibility of routing packets through the neighboring routers to different destination nodes. The presented algorithm in [18] does not require any routing tables, but packets take unnecessary non-minimal paths. In this algorithm, an output hierarchy is defined for each position in the network. According to the current and destination positions of a packet, the routing algorithm scans the hierarchy in a descending order and selects the highest available direction. BFT-NoC [19] presents a different perspective to tolerate faulty links. This approach tries to maintain the connectivity between the routers through a dynamic sharing of surviving channels. Zhen Zhang et al. present an algorithm [11] to tolerate a single faulty router in the network without using virtual channels. The main idea of this algorithm is to route packets through a cycle free contour surrounding a faulty router. Each router should be informed about the faulty or healthy status of eight direct and indirect neighboring routers. This algorithm is extended in [20] to tolerate two faulty links or routers in the network. It utilizes two virtual channels along both X and Y dimensions. Each router needs to know the fault statuses of twelve surrounding links or eight neighboring routers. DBP [21] approach uses a lightweight approach to maintain the network connectivity among non-faulty routers. In this approach, besides the underlying interconnection infrastructure, all routers are connected to each other via an embedded unidirectional cycle (e.g. a Hamiltonian cycle or a ring along a spanning tree). A default back-up path is used at each router to connect the upstream to the downstream router. This algorithm is based on taking non-minimal routes.

In sum, some of the proposals require to know the fault statuses of all routers or links in the network. To collect this knowledge, online or offline techniques are employed. In some approaches, for making a correct decision, the fault statuses of direct and indirect neighboring routers or links are needed. In few numbers of approaches, the routing decision is based on the minimum knowledge about the faults in the network (i.e. four neighboring routers or connected links). As the fault might occur on additional resources, the simpler approach is always preferred.

The fault-tolerant algorithms based on turn models are very limited in rerouting packets and thus they are usually unable to tolerate multiple faults in the network. Deactivations of healthy resources or employing virtual channels are commonly used solutions to support multiple faults.

Fault-tolerant algorithms are usually very complicated due to considering various conditions, such as the location of faults, the number of faults, turn model rules, etc. That is why almost all of them are based on deterministic method, to be able to control different conditions. Most importantly, the best known algorithms are based on rerouting packets around faults which may result in taking unnecessary non-minimal paths. These approaches affect the performance significantly not only by taking longer paths but also by creating hotspot around a fault.

The presented algorithm (MiCoF) has many advantages over traditional methods: 1- it maintains the performance level of NoCs by choosing only the shortest paths for each pair of source and destination router in the presence of faults. 2- the routing unit only requires the fault status of the adjacent routers, that is the minimum knowledge needed by a fault-tolerant routing algorithm. 3- the algorithm is very simple such that it can be implemented in few lines of code. This small piece of code covers all positions of faulty routers and does not set any exceptional rule for borderline routers. 4- it requires only one and two virtual channels along the X and Y dimensions, which is the minimum amount of virtual channels to design a fully adaptive routing algorithm. Moreover, it does not require any routing table. 5- unlike traditional methods, when a router becomes faulty, its belonging links can still be utilized. 6- it is highly reliable such that on average 99.5% of packets successfully reach their destinations when there are six faulty routers in the network. By another metric of reliability, when six faults occur in the network, with the probability of more than 50%, the network functions normally without any message loss. Reachability is also another highlighting point of this method as a router with all neighbors faulty is still reachable.

Besides the mentioned advantages, we need to consider that MiCoF may result in a formation of long wires in the network when supporting a large number of faults. In addition, Non-minimal methods or adding virtual channels are necessitated when multiple faults are going to be supported by 100%. We will investigate these issues in our future work.

III. PRELIMINARIES

Fig. 1(a) shows a typical router in the XY network. In this figure, each input channel is paired with a corresponding output channel. By adding two virtual channels per physical channel, a double-XY network is obtained (Fig. 1(b)). The virtual channels in each dimension are differentiated by vc1 and vc2. Fig. 1(c) shows the double-Y network in which one and two virtual channels are used along the X and Y dimensions, respectively. Each router in the double-Y network has seven pairs of channels, i.e. East(E), West(W), North-vc1(N1), North-vc2(N2), South-vc1(S1), South-vc2(S2), and Local(L). The idea of this paper is developed upon a double-Y network. However, it can be implemented on a double-XY network or a network with more virtual channels.

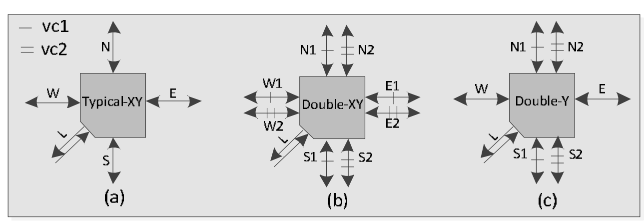

Fig. 1. A router in (a) XY (b) double-XY (c) double-Y network

The proposed fault-tolerant routing algorithm is based on a fully adaptive routing algorithm. In this paper, one and two virtual channels are used along the X and Y dimensions. This is the minimum number of virtual channels that can be employed to provide fully adaptiveness. In double-Y networks, commonly the following method is used to guarantee the deadlock freeness. The network is partitioned into two sub-networks called +X and –X, each having half of the channels in the Y dimension. Eastward packets are routed through +X sub-network (i.e. by using the first virtual channel (vc1) in the Y dimension) while westward packets are propagated within –X sub-network (i.e. by using the second virtual channel (vc2) along the Y dimension).

IV. THE MICOF APPROACH

A. The Router Architecture of MiCoF

In NoCs, faults may occur in cores (such as processing elements and memory modules), links or routers. When a core is faulty, the connected router and links can continue functioning. When a link is faulty, the approaches similar to BFT-NoC [19] retain the connectivity by a dynamic sharing of surviving channels. Thereby, cores and routers perform functioning normally. The most severe case causes by a faulty router. In this case, not only the connected core cannot send or receive packets, but also the packets from the other cores cannot be transmitted through this router. The most common solution is to reroute packets around the faulty router or faulty region. This may imply a non-minimal routing algorithm based on turn models. However, turn models are very limited to tolerate multiple faulty routers. In sum, when a router is faulty, the corresponding core and links are also tagged as faulty and become out of use. The core cannot start working until the fault is recovered.

In this work, we show that the links can still be used to retain the performance by a simple modification in the router architecture. Fig. 2 shows the router architecture in the MiCoF approach. Normal router architecture includes input buffers, a routing unit, a virtual channel allocator, a switch allocator and a crossbar switch. In our modified architecture, in a case of faults, the east input channel is directly connected to the west output channel while the west input channel is connected to the east output channel. Similarly, the packets coming from the north or south input channels are directly connected to the south or north output channels, respectively. So, no processing takes place in the router and packets are not stored in the input buffer. So, the flow control is normally performed between the surviving routers. In other words, the whole router acts as a wire, connecting the input channels to output channels in specific directions. Compared with normal router architecture, this architecture needs a few amounts of multiplexers and de-multiplexers at input and output ports plus a small wiring overhead.

Fig.3 (a) shows a 4×4 mesh topology with five faulty routers. A faulty router itself and the core connected to it are disconnected from the network while the links are used to connect the neighboring routers in appropriate directions. Using the MiCoF architecture, the resulted network is illustrated in Fig. 3(b).

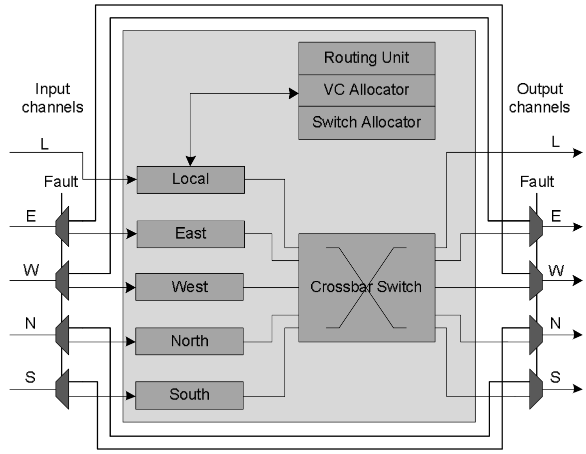

Fig. 2. Router architecture using MiCoF

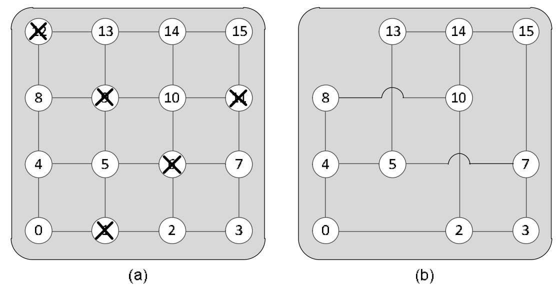

Fig. 3. (a) five fautly routers in a 4×4 mesh topology (b) the resulted network by using the MiCoF approach

B. Fault-Tolerant Routing Algorithm Supporting the MiCoF Architecture

In this subsection, we present a fault-tolerant routing algorithm based on the presented architecture. This algorithm is able to support all locations of a single faulty router without taking non-minimal routes. Using the characteristics of fully adaptiveness, packets usually have an alternative choice when facing a fault in one direction. Let us follow a northeast-ward packet when there is a single faulty router in the network. As shown in Fig. 4(a), when the destination is to the east or north of the current router, the packet can bypass the faulty router and reach the destination router without taking any non-minimal path. This is achieved as the faulty router can be considered as a wire connecting the links in the horizontal and orthogonal directions. In Fig. 4(b), the packet is one hop away from the destination router in both the X and Y dimensions (*Delta_x=1 and Delta_y=1*). By default, the packet is sent to the Y direction (patterns 1 and 2). However, when the neighboring router in the Y direction is faulty, the packet is delivered to the X direction instead (pattern 3). Fig. 4(c) indicates the cases where the distances are two (or it can be greater than two) hops and one hop along the X and Y dimensions (*Delta_x>=2 and Delta_y=1*), respectively. Similar to the previous case, the packet is sent to the Y direction unless the neighboring router in this direction is faulty (patterns 1, 2, and 3). By delivering the packet to the Y direction, the packet is placed in the same row as the destination router where it can reach the destination regardless of faults in the path. When the neighboring node in the Y dimension is faulty (pattern 4), the packet is delivered to

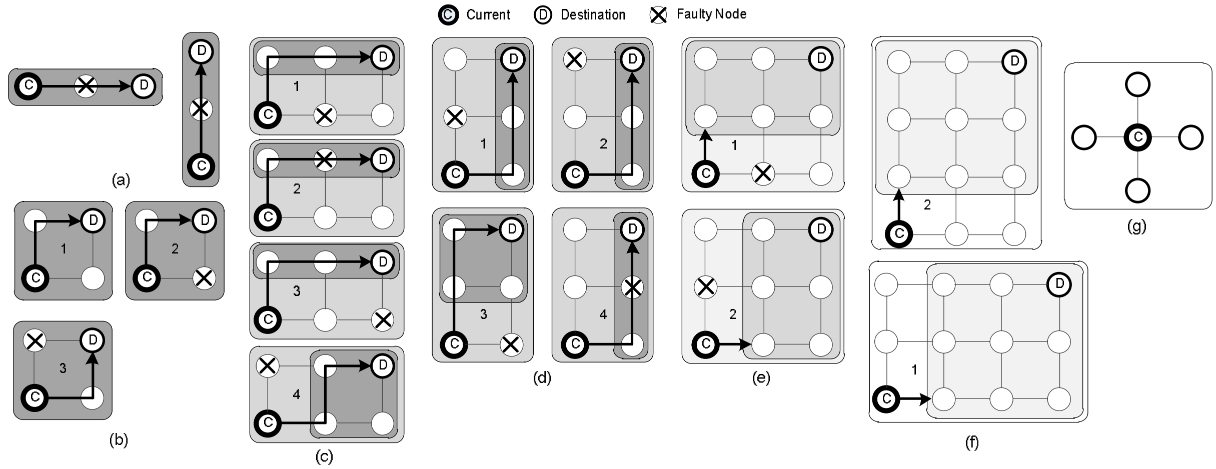

Fig. 4. Tolerating all one-faulty routers using only shortest paths

the X direction. In the next hop, the packet stands in one of the positions of Fig. 4(a) or Fig. 4(b). In other words, in all positions of a faulty router, the packet could reach the destination by using the shortest paths. In Fig. 4(d), the distance is one and two (or greater than two) hops along the X and Y directions (*Delta_x=1 and Delta_y>=2*), respectively. The rule is as simple as avoiding to send the packet to the Y direction when the neighboring router in the X direction is non-faulty. Using this rule, all positions of faults can be covered by patterns 1, 2, 3, and 4. In the next hop, the packet stands in one of the positions of Fig. 4(a) or Fig. 4(b). When the distances along both directions are two or greater than two hops (*Delta_x>=2 and Delta_y>=2*), the packet should be sent to a non-faulty neighboring router (Fig. 4(e)). By routing the packet with this policy, the packet reaches one of the positions of Fig. 4(c) or Fig. 4(d). In a case where both neighboring routers are non-faulty, the packet is sent through the less congested direction. If the network is designed to tolerate a few numbers of faults, packets can be routed adaptively in the network as long as the remaining distance along both directions is equal or greater than two hops. To sum up, it is guaranteed that all situations of a single faulty router are covered by MiCoF taking only the shortest paths between each pair of source and destination routers. In addition, fully adaptiveness is provided when the packet is not close to the destination router (i.e. the packet is close to the destination when the distance along one dimension is at most one hop). However, for better reliability, the adaptiveness of MiCoF can be limited. The limitation is applied to a situation when the distances along both directions are two or greater than two hops. In this case, a packet is sent to a (non-faulty) larger-distance direction (Fig. 4(f)). The packet can be sent through either direction when the distances along both directions are equal. Obviously, this adaptivity limitation does not affect the behavior of the algorithm in supporting all single faulty routers without misrouting packets. Another interesting point is that, as illustrated in Fig. 4(g), this algorithm only requires the fault information of four neighboring routers (i.e. it is normally provided for any fault-tolerant method). Thereby, this algorithm does not impose any area overhead due to collecting the fault information throughout the network. Taking into account that faults might occur on additional resource, the less used resources offers the more reliable method.

As it is already mentioned, fault-tolerant routing algorithms are usually very complicated. In contrast with them, the MiCoF algorithm is very simple with a negligible area overhead. The whole MiCoF routing algorithm is shown in Fig. 5. In this figure, the adaptiveness is limited to support more faulty switches. However, the adaptiveness can be increased when the performance is more preferred than the high reliability.

```
Definitions:        Xd,Yd: X and Y of destination switch
                     Xc,Yc: X and Y of current switch
                    ngbr: neighboring router
          **---------------------------------------------**
x_dir <= E when Xd>Xc else W;
y_dir <= N when Yd>Yc else S;
Delta_x <= (Xd-Xc) when Xd>Xc else (Xc-Xd);
Delta_y <= (Yd-Yc) when Yd>Yc else (Yc-Yd);

if (Delta_x =0 and Delta_y =0) then select <= local;
elsif (Delta_x >=1 and Delta_y =0) then select <= x_dir;
elsif (Delta_x =0 and Delta_y >=1) then select <= y_dir;
elsif (Delta_x >=1 and Delta_y =1) then
      if ngbr(y_dir)=healthy then select <= y_dir;
      else select <= x_dir; end if;
elsif (Delta_x =1 and Delta_y >=2) then
      if ngbr(x_dir)=healthy then select <= x_dir;
      else select <= y_dir; end if;
else
      if (ngbr(x_dir)=faulty and ngbr(y_dir)=faulty) or
         (ngbr(x_dir)=healthy and ngbr(y_dir)=healthy) then
            if (Delta_x > Delta_y) then select <= x_dir;
            elsif (Delta_x < Delta_y) then select <= y_dir;
            else select <= x_dir or y_dir; end if;
      elsif ngbr(y_dir)=healthy then select <= y_dir;
      elsif ngbr(x_dir)=healthy then select <= x_dir;
      end if;
end if;
```

Fig. 5. Pseudo VHDL code of MiCoF

V. RELIABILITY UNDER MULTIPLE FAULTY ROUTERS

A. Two Faulty Routers

By the MiCoF approach, all positions of one faulty router can be tolerated using only the shortest paths. The MiCoF algorithm does not change in order to support more faults. In this subsection, we investigate how two faulty routers can be

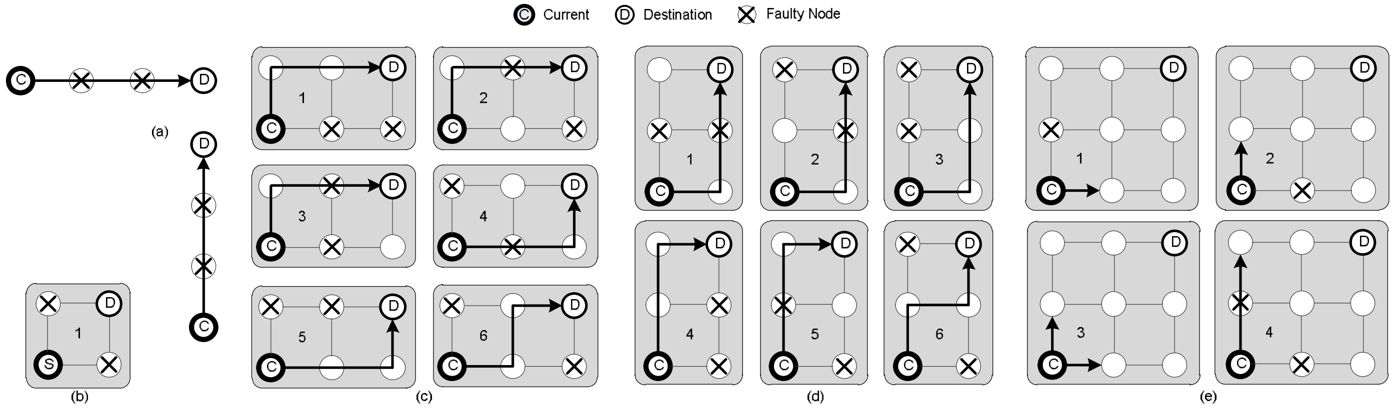

Fig. 6. Tolerating two faulty routers by the MiCoF approach

tolerated using the same routing algorithm. As shown in Fig. 6(a), when the current and destination routers are located in the same row or column, the packet can reach the destination regardless of the faulty routers in the path. Fig. 6(c) indicates all the six positions of two faulty routers when the distances are two and one hops along the X and Y dimensions. According to the MiCoF algorithm, by default, the packet is sent to the Y direction. If the neighboring router in the Y direction is faulty, the packet is sent to the X direction. In patterns 1, 2, and 3 of Fig. 6(c), the packet is sent to the neighboring router in the Y direction as it is healthy. In the next hop, the packet faces to the similar situation as in Fig. 4(a). Patterns 4, 5, and 6 cover the cases when the neighboring router in the Y direction is faulty, and thus the packet is sent to the X direction. In the next hop, the packet stands in one of the positions of Fig. 4(b) or it simply passes through the faulty router. A similar approach is applied when the distance is one and two hops along the X and Y dimensions (Fig. 6(d)). Fig. 6(e) shows the cases in which the distance is two hops along both directions. If the neighboring router in one of the directions is faulty, the packet is sent through the non-faulty direction (pattern 1 and 2). From the next hop to the destination router, there might be at most one faulty router in the path that can be supported according to Fig. 4(c) or Fig. 4(d). If both neighboring routers are healthy (position 3 in Fig. 6(e)), in the remaining path from the next hop to the destination router, the packet might face two faulty routers. As indicated in Fig. 4(c), Fig. 4(d), Fig. 6(c), and Fig. 6(d), all one and two faulty routers are supported by MiCoF. If both neighboring routers are faulty, the packet can be sent through either direction. This packet will not face any fault in the remaining path as both faults are already bypassed.

So far, all two faulty routers are supported by MiCoF using only the shortest paths. There is only one position in which two faulty routers cannot be supported using the shortest paths. This is the case when the distance from the source to the destination router is one along both dimensions while the neighboring routers in both directions are faulty (Fig. 6(b)). These positions of faults are called diagonal positions. The source router still can send and receive packets to/from every other router in the network except the destination router. In other words, only the packets from this specific source router position cannot reach to this specific destination router position (or vice versa). All the other packets can be normally routed in the network. If the source router is farther away from the destination router, the packet never stands in this unsupported position as the packet already chooses other routes prior reaching this position (e.g. similar to the pattern 2 of Fig. 6(c) and Fig. 6(d)).

B. Reliability Analysis of Two Faulty Routers

In this paper, we use two reliability metrics named reliability1 and reliability2 in our measurements. Reliability1 shows the probability that the network can successfully deliver any packet under the existence of fault. Reliability2 is the probability that a packet can be successfully delivered under fault. These metrics can be calculated as follows:

- Reliability1:

  According to MiCoF, if two faults are located in diagonal positions, the network may fail. At first, we calculate the number of total combinations of two faulty switches in the network. Then, we measure the number of combinations in which two faults occur in diagonal positions. By dividing these two numbers, the reliability value is obtained. The number of different combinations of two faulty switches in an $n \times n$ mesh network can be measured by:

$$N_{all\_combinations} = \binom{n^2}{2} = \frac{n^2(n^2-1)}{2}$$

  Fig. 7 shows all combinations in which two faulty switches are located in diagonal positions. By extending the idea to an $n \times n$ mesh network, the number of diagonal combinations can be calculated by:

$$N_{diagonal\_combinations} = 2(n-1)^2$$

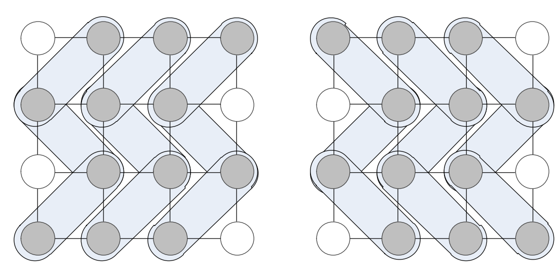

Fig. 7. A couple indicates a diagonal position (i.e. nine diagonal positions in each figure)

Finally, the Reliability1 can be calculated by:

$$R1 = 1 - \frac{N_{diagonal\_combinations}}{N_{all\_combinations}} = 1 - 4\frac{(n-1)^2}{n^2(n^2-1)}$$

According to this formula, for example in an 8×8 mesh network, with the probability of 95.2%, two faults will not be located in diagonal positions, and thus the network functioning normally without dropping any message.

- Reliability2:

The second definition is mostly used in literature to report the reliability value. Let us assume that the network is examined under all combinations of two-faulty switches. Thereby, the number of examinations is equal to the combinations of two faulty switches ($N_{all\text{-}combinations}$). Per examination, each healthy switch delivers one packet to every other healthy switch in the network (i.e. total of n^2-3 packets, except itself and two faulty switches). As faulty switches do not send or receive any packets (i.e. total of n^2-2 switches are able to deliver packets), the total number of delivered packets per combination is:

$$N_{delivered\_per\_combination} = (n^2-2)(n^2-3)$$

Therefore, the total number of delivered packets in the whole examinations is:

$$Total = N_{delivered\_per\_combination} \times N_{all\_combinations}$$

On the other hand, per diagonal position, two packets must be dropped (those from the source to the destination switch or vice versa), so that the total number of defeated packets is calculated by:

$$Defeated = 2 \times N_{diagonal\_combinations} = 4(n-1)^2$$

Therefore, the reliability2 can be measured by:

$$R2 = 1 - \frac{Defeated}{Total}$$

According to this formula, in an 8×8 mesh network, 99.998% of packets reach their destination considering all combinations of two faulty routers.

C. Three Faulty Routers

In this subsection, we take a quick look at three faulty routers in the network. If three faults are well distributed over the network, there are easily supported according to Fig. 4 and Fig. 6. However, when there are close to each other, the fault situations shown in Fig. 8 are obtained. If the locations of three faults are similar to the patterns 1 or 2 of Fig. 8(a) and Fig. 8(b), then it is supported by the MiCoF routing algorithm (Fig. 5). In patterns 3 and 4, a few percentages of packets cannot reach the destination. Even under these severe fault conditions, the rest of the packets can be routed to their destinations through the shortest paths.

The focus of this work is to tolerate faults by only using the shortest paths without any performance loss. However, non-minimal paths or virtual channels can be used to support the remaining cases.

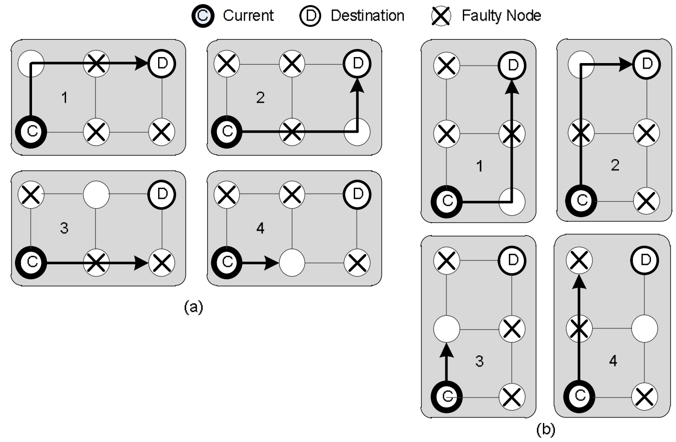

Fig. 8. Three faulty routers in the network which are close to each other

VI. EXPERIMENTAL RESULTS

To evaluate the efficiency of the proposed approach, a NoC simulator is developed with VHDL to model all major components of the on-chip network. For all the routers, the data width is set to 32 bits. Each input buffer can accommodate 8 flits in each virtual channel. Moreover, the packet length is uniformly distributed between 5 and 10 flits. As a performance metric, we use latency defined as the number of cycles between the initiation of a message issued by a Processing Element (PE) and the time when the message is completely delivered to the destination PE. The simulator is warmed up for 12,000 cycles and then the average performance is measured over another 200,000 cycles.

MiCoF is designed based on using one and two virtual channels along the X and Y dimensions. We implemented two other methods, called ReRS [11] (Reconfigurable Routing for Tolerating Faulty Switches) and RAFT [20] (Reconfigurable Routing for Tolerating Faulty Links). Unlike MiCoF, both methods are based on a detour strategy and thereby packets may take unnecessary longer paths to reach destinations. ReRS requires one virtual channel along each dimension and is able to tolerate all single faulty routers. RAFT utilizes two virtual channels to support all two faulty routers.

A. Reliability Evaluation under Uniform Traffic Profile

In the uniform traffic profile, each processing element (PE) generates data packets and sends them to another PE using a uniform distribution [22]. The mesh size is considered to be 8×8. To evaluate the reliability of MiCoF, the number of faulty routers increases from one to six. All faulty routers are selected using a random function. The results are obtained using 10,000 iterations when the traffic is uniform random.

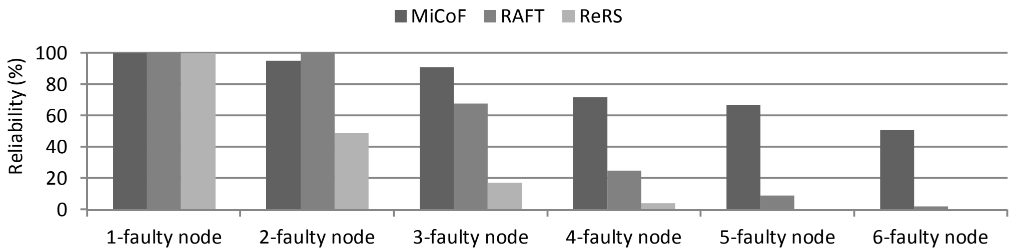

Fig. 9. Reliability measurment based on the first definition

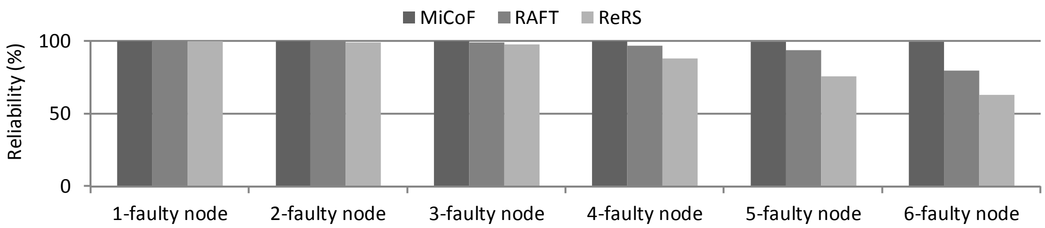

Fig. 10. Reliability measurment based on the second definition

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. The reliability values based on the first metric is shown in Fig. 9. All three approaches are reliable by 100% when there is a single fault in the network. RAFT is the only approach that can guarantee the reliability by 100% under the cases of two faulty routers. However, this method utilizes one more virtual channel than MiCoF. As illustrated in this figure, the reliability of ReRS and RAFT abruptly decreases with more faulty routers. MiCoF is highly reliable, for an instance, in 50% of all combinations of six faulty routers, the network is functioning normally without any packet loss. Fig. 10 shows the reliability measurement based on the second metric. The important point is that the reliability of MiCoF is more than 99.5% under six faulty routers in the network.

B. *Performance Analysis under Uniform Traffic Profile*

The performance analysis under uniform random traffic is shown in Fig. 11. The average communication latency of ReRS and RAFT are obtained under the cases of single and two faulty routers as more faulty routers are not well supported. The average communication latency of MiCoF is measured under one to six faulty routers in the network. To have a fair performance comparison we use two virtual channels in all three methods. The extra virtual channels are used for the performance purposes. In a fault-free network, the performance of all methods is comparable while RAFT outperforms others because of its better adaptiveness. As the number of faults increases, the performance of ReRS and RAFT significantly decreases. We increase the number of faulty routers to six faults and measure the performance of MiCoF. Surprisingly, the performance gradually starts growing under the same traffic load.

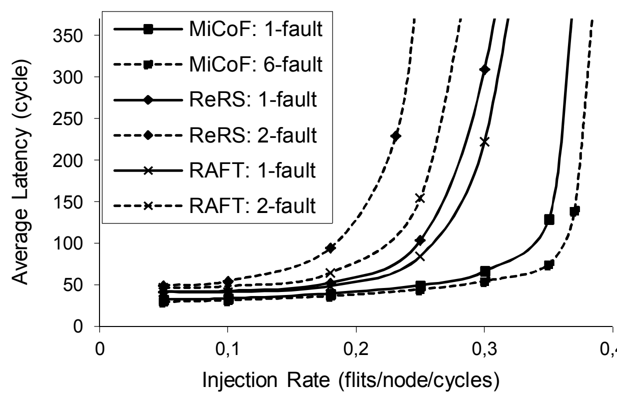

Fig. 11. Performance under uniform random traffic

This improvement is from the communication point of view while the whole system performance will be obviously decreased by occurring faults in the network. This is due to the fact that the routing does not take place in faulty routers and the total number of hops is decreased. For clarity, the performance curves of two- to five- faulty routers are omitted, but they are distributed between the curves of one- and six-faulty routers.

C. *Performance Analysis under Hotspot Traffic Profile*

Under the hotspot traffic pattern, one or more routers are chosen as hotspots receiving an extra portion of the traffic in addition to the regular uniform traffic. In simulations, given a hotspot percentage of H, a newly generated message is directed to each hotspot router with an additional H percent probability. We simulate the hotspot traffic with a single hotspot router at *(4,4)* in an 8×8 mesh network. The performance of each network under different numbers of faulty routers and $H=10\%$ is illustrated in Fig. 12.

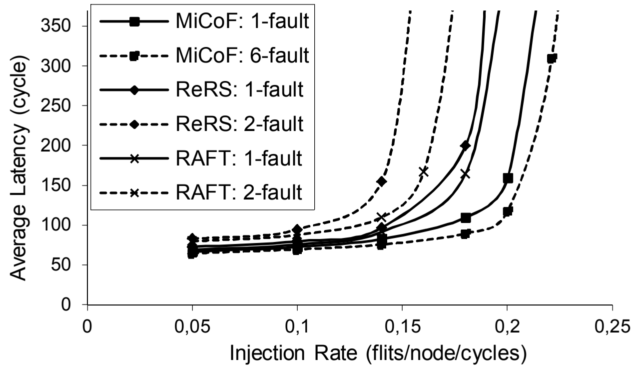

Fig. 12. Performance under hotspot traffic

D. Hardware Analysis

To assess the area overhead and power consumption, the whole platform of each method is synthesized by Synopsys Design Compiler. We compared the area overhead and power consumption of MiCoF with ReRS and RAFT. In this set of analysis, ReRS has no virtual channel, MiCoF uses one virtual channel in the Y dimension, and RAFT utilizes two virtual channels along both directions. The power consumption of all methods is measured in a non-faulty network. For each scheme, we include network interfaces, routers, and communication channels (MiCoF uses additional resources for connection retaining purposes (Fig. 2)). For synthesizing, we use the UMC 90nm technology at the operating frequency of 1GHz and supply voltage of 1V. We perform place-and-route, using Cadence Encounter, to have precise power and area estimations. The power dissipation is calculated using Synopsys PrimePower in a 6×6 mesh network. The layout area and power consumption of each platform are shown in Table 1. As indicated in the table, MiCoF has a lower area overhead than the RAFT and higher one than ReRS. This is mostly because of using different numbers of virtual channels.

Table 1. Details of hardware implementation

Network platforms	Area (mm^2)	Power (W) dynamic & static
MiCoF	6.886	2.40
ReRS	6.513	2.10
RAFT	7.295	2.85

VII. CONCLUSION

In this paper, we proposed a fault-tolerant approach named MiCoF. In the presented approach, all packets are routed through the shortest paths, maintaining the performance of NoC in the presence of faults. To be able to route packets through the shortest paths, the router architecture is slightly modified. The purpose of this modification is to maintain the connectivity among the surviving routers. For this to happen, when a router becomes faulty, the links are simply connected to each other along the horizontal and orthogonal directions. MiCoF is a very simple, lightweight, and adaptive approach which takes advantage of only one and two virtual channels along the X and Y dimensions. The high reliability provided by this simple approach is a final conclusion of this work.

REFERENCES

[1] Xu, Jiang, W. Wolf, J. Hankel, S. Charkdhar, "A Methodology for design, modeling and analysis for networks-on-Chip," in Proc. of IEEE International Symposium on Circuits and Systems, pp. 1778-1781, 2005.

[2] W. Tsai, D. Zheng, S. Chen, and Y.H. Hu, "A fault-tolerant NoC scheme using bidirectional channel", in Proc. DAC, pp.918-923, 2011.

[3] E. Rijpkema et al., "Trade offs in the design of a router with both guaranteed and best-effort services for networks on chip," in Proc. of DATE'03, pp. 350-355, 2003.

[4] M. Ali, M. Welzl, S.Hessler, "A Fault tolerant mechanism for handling Permanent and Transient Failures in a Network on Chip," in Proc. of international conference on Information Technology, pp.1027-1032, 2007.

[5] J. Duato, S. Yalamanchili, L. Ni, "Interconnection networks: an engineering approach", Morgan Kaufmann Publishers, 2003.

[6] M. Daneshtalab, M. Kamali, M. Ebrahimi, S. Mohammadi, A. Afzali-Kusha, and J. Plosila, "Adaptive Input-output Selection Based On-Chip Router Architecture," Journal of Low Power Electronics (JOLPE), Vol. 8, No. 1, pp. 11-29, 2012.

[7] J. Wu, "A Fault-Tolerant and Deadlock-Free Routing Protocol in 2D Meshes Based on Odd-Even Turn Model", IEEE transaction on computers, v. 52, pp.1154-1169 ,2003.

[8] F. Chaix, et al., "A fault-tolerant deadlock-free adaptive routing for On Chip interconnects," in Proc. of DATE, pp. 1-4, 2011.

[9] S. Jovanovic, C. Tanougast, et al., "A new deadlock-free fault-tolerant routing algorithm for NoC interconnections", in Proc. of FPL, pp.326-331, 2009.

[10] J. Wu and D. Wang, "Fault-tolerant and deadlock-free routing in 2-D meshes using rectilinear-monotone polygonal fault blocks", in Proc. of Parallel Algorithms., pp.99-111, 2005.

[11] Z. Zhen, A. Greiner, S. Taktak, "A reconfigurable routing algorithm for a fault-tolerant 2D-Mesh Network-on-Chip", DAC, pp. 441-446, 2008.

[12] M. Ebrahimi et al., "MD: Minimal path-based Fault-Tolerant Routing in On-Chip Networks," in Proc. of ASP-DAC, pp. 35-40, 2013.

[13] M. Ebrahimi et al., "MAFA: Adaptive Fault-Tolerant Routing Algorithm for Networks-on-Chip," in Proc. of DSD, pp. 201-206, 2012.

[14] M. Ebrahimi et al., "High Performance Fault-Tolerant Routing Algorithm for NoC-based Many-Core Systems," in Proc. of PDP, 2013.

[15] D. Fick. A. DeOrio, G. Chen, v. Bertacco, D. Sylverster, D.Blaauw, "A highly resilient routing algorithm for fault-tolerant NoCs", in Proc. of DATE, pp. 21-26, 2009.

[16] Ch. Feng, Zh. L, A. Jantsch, J. Li, M. Zhang, "A Reconfigurable Fault-tolerant Deflection Routing Algorithm Based on Reinforcement Learning for Network-on-Chip", in Proc. of NoCArc, 2010.

[17] M. Ebrahimi, M. Daneshtalab, F. Farahnakian, P. Liljeberg, J. Plosila, M. Palesi, and H. Tenhunen, "HARAQ: Congestion-Aware Learning Model for Highly Adaptive Routing Algorithm in On-Chip Networks," in Proc of NOCS, pp. 19-26, 2012.

[18] F. Chaix, D. Avresky, N. Zergainoh, M. Nicolaidis, "Fault-Tolerant Deadlock-Free Adaptive Routing for Any Set of Link and Node Failures in Multi-cores Systems", in Proc. of NCA, pp.52-59, 2010.

[19] W. Tsai, D. Zheng, S. Chen, and Y.H. Hu, "A fault-tolerant NoC scheme using bidirectional channel", in Proc. of DAC, pp.918-923, 2011.

[20] M. Valinataja, S. Mohammadi, J. Plosila, P. Liljeberg, H. Tenhunen, "A reconfigurable and adaptive routing method for fault-tolerant mesh-based networks-on-chip," International Journal of Electronics and Communications (AEU), v. 65, I.7, pp. 630-640, 2011.

[21] M. Koibuchi, H. Matsutani, H. Amano, T.M. Pinkston, "A Lightweight Fault-Tolerant Mechanism for Network-on-Chip", in Proc. of NoCS, pp. 13-22, 2008.

[22] C.J. Glass et al., "The Turn Model for Adaptive Routing", in Proc. of 19th Int'l Symp. Computer Architecture, pp. 278-287, 1992.