MD: Minimal path-based Fault-Tolerant Routing in On-Chip Networks

Masoumeh Ebrahimi[1], Masoud Daneshtalab[1], Juha Plosila[1], Farhad Mehdipour[2]
[1]Department of Information Technology, University of Turku, Finland
[2]E-JUST Center, Kyushu University, Japan
{masebr, masdan, juplos}@utu.fi, farhad@ejust.kyushu-u.ac.jp

Abstract— **The communication requirements of many-core embedded systems are convened by the emerging Network-on-Chip (NoC) paradigm. As on-chip communication reliability is a crucial factor in many-core systems, the NoC paradigm should address the reliability issues. Using fault-tolerant routing algorithms to reroute packets around faulty regions will increase the packet latency and create congestion around the faulty region. On the other hand, the performance of NoC is highly affected by the network congestion. Congestion in the network can increase the delay of packets to route from a source to a destination, so it should be avoided. In this paper, a minimal and defect-resilient (MD) routing algorithm is proposed in order to route packets adaptively through the shortest paths in the presence of a faulty link, as long as a path exists. To avoid congestion, output channels can be adaptively chosen whenever the distance from the current to destination node is greater than one hop along both directions. In addition, an analytical model is presented to evaluate MD for two-faulty cases.**

I. INTRODUCTION

As is predicted by Moore's law, over a billion transistors could be integrated on a single chip in the near future [1]. In these chips, hundreds of functional intellectual property (IP) blocks and embedded memory modules could be placed together to form a Multi-Processor Systems-on-Chip (MPSoCs) [1]. By increasing the number of processing elements in a single chip, traditional bus-based architectures in MPSoCs are not useful any longer and a new communication infrastructure is needed. Network-on-Chip (NoC) has become a promising solution for on-chip interconnection in many-core Systems-on-Chip (SoC) due to its reusability and scalability [2][3][4].

On-chip interconnects implemented with deep submicron semiconductor technology, running at GHz clock frequencies are prone to failures [2][5][6]. Due to this extreme device scaling, the likelihood of failures increases [17]. Two different types of faults that can occur in NoCs are transient and permanent. Transient faults have unpredictable causes (e.g. power grid fluctuations, particle hits) and are often difficult to be detected and corrected. Permanent faults are caused by physical damages such as manufacturing defects, device wear-out. In this paper, our focus is on permanent faults. Routing techniques provide some degrees of fault tolerance in NoCs. Routing algorithms are mainly categorized into deterministic and adaptive approaches [7][8][9]. A deterministic routing algorithm uses a fixed path for each pair of nodes, resulting in increased packet latency especially in congested networks. Implementations of deterministic routing algorithms are simple but they are unable to balance the load across the links in non-uniform traffic. The simplest deterministic routing method is dimension-order routing which is known as XY or YX algorithm. The dimension-order routing algorithms route packets by crossing dimensions in strictly increasing order, reducing to zero the offset in one direction before routing in the next one. In contrast, in adaptive routing algorithms, a packet is not restricted to a single path when traveling toward a destination node. So they can decrease the probability of routing packets through congested or faulty regions. In sum, unlike deterministic routing algorithms, adaptive routing algorithms could avoid congestion in the network and provide better fault-tolerant characteristics by utilizing alternative routing paths.

In wormhole routings, messages are divided into small flits traveling through the network in a pipelined fashion. This approach eliminates the need to allocate large buffers in intermediate switches along the path [10]. Moreover, in wormhole routing, message latency is less sensible to distance. However, it should be used with special care to avoid deadlock in the network. Deadlock is a situation when the network resources continuously wait for each other to be released. Routing algorithms are required to be deadlock-free and break all cyclic dependencies among channels. Virtual channels are mainly used in the network for avoiding deadlock, increasing performance and tolerating faults, but it is an expensive solution.

Conventional fault-tolerant routing algorithms reroute packets around faulty regions, either convex or concave, so that the selected paths are not always the shortest ones. However, rerouting is an expensive solution and considerably increases packet's latency and router's complexity. In addition the information about faulty components is insufficient or the way of utilizing them is inefficient. On the other hand, most of the presented fault tolerant algorithms are limited to deterministic routing algorithms, resulting in considerable performance loss. In this paper, we present a minimal and defect-resilient (MD) routing algorithm where the key ideas are threefold. First, it can tolerate all one-faulty links using a minimal path between each pair of source and destination nodes, if a minimal path exists. Second, to avoid congestion, output channels can be adaptively chosen whenever the distance from the current to destination node is greater than one hop along both directions. Third, to evaluate the presented routing for two faults condition, an analytical model is introduced and analyzed.

The rest of this paper is organized as follows: Section II reviews the related work, while the underlying fully adaptive routing algorithm is also discussed in this section. Fault distribution mechanism, bypassing faulty links and the proposed fault-tolerant algorithm are explained in Section III. The results are given in Section IV while we summarize and conclude in the last section.

II. RELATED WORK

Fault-tolerant routing algorithms can be classified into two main groups: one can handle convex or concave regions [11][12][13][14] and the other utilizes the contour strategy for addressing faults [15][16]. The basic assumptions in all of these methods are the permanent faulty cases. The methods in the first group are based on defining fault ring (f-ring) or fault chain (f-chain) around faulty regions where healthy nodes are disabled in order to form a specific shape. A reconfigurable routing algorithm using the contour strategy provides the possibility of routing packets through a cycle free surrounding a faulty component. The presented algorithm in [15] is able to tolerate all one-faulty routers in 2D mesh network without using virtual channels and disabling healthy nodes. However, to support more number of faulty routers, the contours must not be overlapped and thus faulty routers should be located far away from each other. This algorithm is deterministic and does not make any effort toward alleviating congestion in the network.

Fault-tolerant routing algorithms could be also divided into two classes: the methods using virtual channels [16][17][18] and those without using virtual channels [19][20]. In general, different methods define a new tradeoff between the number of virtual channels, the ability to handle different fault models, and the degree of adaptiveness. The virtual channel-based fault-tolerant routing algorithms provide better fault-tolerant characteristics than those without virtual channels. The methods that do not use any virtual channel are mainly based on the turn models [21][22]. In turn models, some turns are eliminated in order to guarantee the deadlock freeness in the network and then the remaining turns are used both for routing packets and tolerating faults.

In this paper, the fault information is distributed and utilized in such a way that packets can be routed through the shortest paths in the presence of faults. This method can be used with any number of virtual channels in the network. However, in order to keep the area overhead low, we use two virtual channels in each direction. The proposed method not only is able to tolerate all one-faulty links but also all packets can be routed through the shortest paths as long as any path exists. Moreover, the algorithm is adaptive and in most cases, packets have alternative choices to reach the destination node.

A. Dynamic XY Routing Algorithm

Our proposed method is based on a fully adaptive routing algorithm. To provide this requirement, we take advantage of the Dynamic XY (DyXY) method. DyXY is an adaptive and deadlock free routing algorithm proposed in [23]. In this algorithm, which is based on the static XY algorithm, a packet is sent either to the X or Y direction depending on the congestion condition. It uses local information (i.e. the current queue length of the corresponding input buffer in the neighboring routers) to decide on the next hop. It is shown that the usage of this local information leads to a lower latency path from the source to the destination node. Fig. 1 shows an example of the DyXY method where the nodes S and D are the source and destination of the packet. In the DyXY method, at the source node S, the number of free buffer slots at the west input buffer of node 1 is compared with that of the south input buffer of node 4. The packet is sent in a less congested direction that is the east direction in this example. When the packet arrives at node 1, it has to be delivered toward the destination node through either node 2 or node 5. According to the congestion condition shown in Fig. 1, the node 5 is less congested and thus the packet is delivered to it. In this way, the packet chooses a less congested direction at each routing step until it reaches the destination node.

Due to the fact that packets can be routed in both X and Y directions, there is possibility of deadlock. DyXY uses two virtual channels along the Y direction and one virtual channel along the X direction. This algorithm avoids deadlock as follows: The network is partitioned into two subnetworks called +X and –X, each having half of the channels in the Y dimension. If the destination node is on the right of the source, the packet will be routed through the +X subnetwork. If the destination node is on the left of the source, the packet will be routed through the -X subnetwork. When the destination is in the north or south of the source node, either subnetwork can be used. Without the loss of generality, in this work we take advantage of two virtual channels in each direction. The first virtual channel is used for the eastward packets and the second virtual channel is utilized for the westward packets.

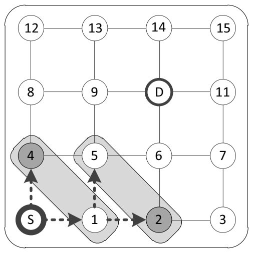


Fig. 1. Dynamic XY (DyXY) routing algorithm

III. MD: THE PROPOSED APPROACH

A. Fault Distribution Mechanism

We present a new fault distribution mechanism and the method of utilization which avoids taking unnecessary non-minimal paths. As shown in Fig. 2(d), the fault information is distributed in a way that each router is informed about the fault condition in two-hop links. For this purpose, each router transfers the fault information on its direct links to the neighboring nodes. Note that in this figure, E, W, N, and S stand for the East, West, North, and South directions. In Fig. 2(a), the neighboring node in the north of the current node (C) transfers the fault information on its links in N, E, and W directions to the current node. Accordingly, the current node would be informed about the fault information in its N, NN, NE, and NW paths. In Fig. 2(b), by receiving the fault information from the neighboring node in the east direction, the current node knows about the fault information of E, EE, EN, and ES paths as well. Similarly, in Fig. 2(c), this knowledge is extended to know about the fault information on the links in S, SS, SE, and SW paths. Finally, as shown in Fig. 2(d) by receiving the information from the neighboring node in the west direction, the current node has the information about the links in E, EE, EN, ES, W, WW, WN, WS, N, NN, NE, NW, S, SS, SE, and SW paths in total. For routing a packet in the northeast direction, for instance, a router uses the fault information on the corresponding minimal paths (e.g. EE, EN, NN, and NE). Similarly, for a southwest packet, the fault's information on some paths (e.g. WW, WS, SS, and SW) is beneficial for making a reliable routing decision. Using this information, packets are routed through minimal and non-faulty paths which avoids making unnecessary routing around faulty components.

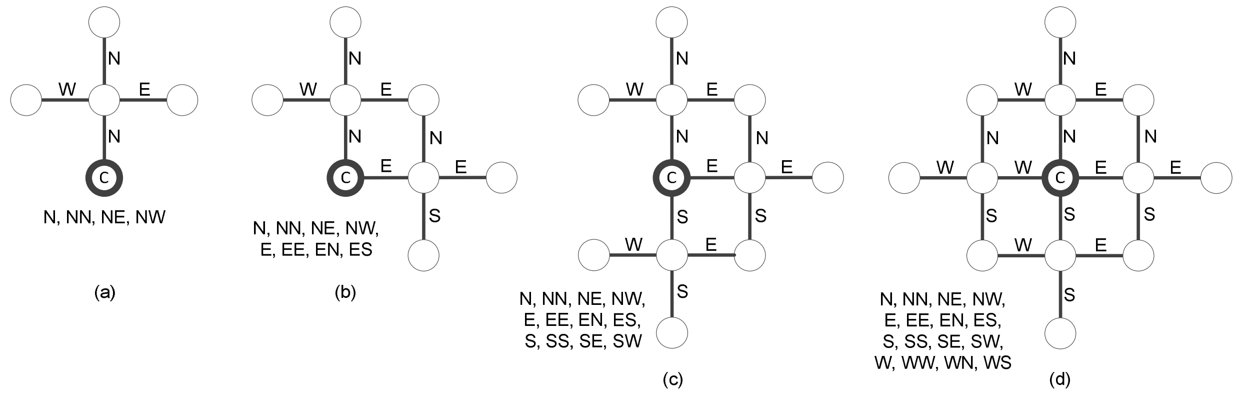


Fig. 2. The fault distribution mechanism

B. Bypassing Faulty Links

Regarding the relative position of the source and destination nodes, a packet can be sent in eight directions: north, south, east, west, northeast, northwest, southeast, and southwest. By using the DyXY method and the explained distribution mechanism (Fig. 2), we show that the packets destined for northeast, northwest, southeast, and southwest directions, take only the shortest paths in the presence of a faulty link in the network. Therefore, no rerouting takes place in these cases and the algorithm remains deadlock free. However, for eastward, westward, northward, and southward packets, non-minimal paths must be taken if a faulty link exists in the path.

1. Northeast, Southeast, Northwest, and Southwest Directions

Using the DyXY method, all shortest paths in the east direction are valid for eastward packets. Similarly, westward packets can utilize all shortest paths in the west direction. When the destination is in the northeast position of the current node, the packet can be delivered in either the north or east direction. As illustrated in Fig. 3(a), the distances along both east and north directions are one. On the other hand, according to the distribution mechanism, the current node is informed about the faulty condition of the links in EN and NE paths. Using this information, if a link is faulty in either the NE or EN path, the other shortest path is selected by the routing unit. As a result, the packet is always routed through a minimal path to the destination.

The example in Fig. 3(b) shows the case where the distance along the X dimension reaches one. The current node knows the information about EN, NE, and NN paths which are located in the minimal path. The packet can be delivered in the east direction if the EN path is non-faulty. However, by this choice, the distance along the X dimension reaches zero and the packet has to take the Y direction in the remaining path toward the destination node. Thereby, if there is a faulty link in the Y dimension, the packet must take a non-minimal path to bypass the fault. This is not an optimal solution which is addressed by the presented algorithm. MD avoids reducing the distance into zero in one direction when the distance along the other direction is greater than one. In other words, when the distance between the current and destination nodes reaches one in at least one dimension, at first all the possible shortest paths on the greater-distance dimension are checked. The packet is sent along the greater-distance dimension if any minimal and non-faulty paths exist; otherwise the links on the smaller-distance dimension are examined. Thereby, in Fig. 3(b) the availability of NN and NE paths is checked before that of EN. If either the NN or NE path is healthy, the packet is sent through it. In the next hop, the packet faces the similar situation as in Fig. 3(a), and thus only the shortest paths are selected by MD so far. In another case of Fig. 3(b), when both NN and NE paths are faulty (i.e. the north link is faulty), the packet is routed through the east direction and is sent to the destination using the shortest path (i.e. we assume there is one faulty link in the network). Similarly, in Fig. 3(c) the conditions of EE and EN paths are examined earlier than the NE path. Finally, in Fig. 3(d), the packet can be delivered through the east direction if the EE or EN path is non-faulty or it can be sent through the north direction when either the NN or NE path is healthy. By these choices, the packet faces the similar situation as in Fig. 3(b) or Fig. 3(c) and thereby only the shortest paths are taken by MD in all cases.

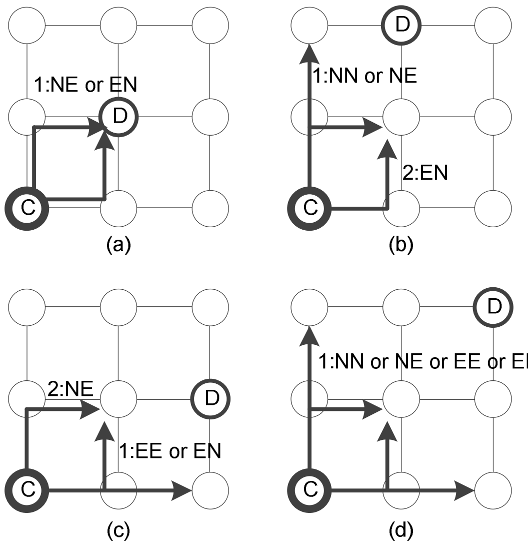


Fig. 3. Bypassing faulty links when the destination is located in the northeast position of the source node (Note that numbers determine the priority of different paths)

2. East, West, North, and South Directions

As is already mentioned, when a packet is eastward, westward, northward, or southward and there is a faulty link in the path, the packet must be routed through a non-minimal path and turned around the faulty link. As illustrated in Fig. 4(a) for the eastward packet, at first the east link is checked and if it is healthy, the packet is sent through this direction. However, if the link is faulty, the packet is delivered to the north or south direction with the same priority. The situation is similar for the westward packet

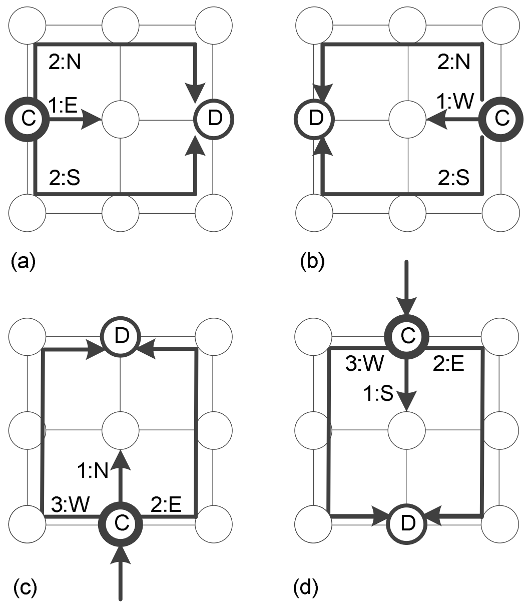


Fig. 4. Bypassing faulty links when the destination is located in the (a) east (b) west (c) north, and (d) south positions of the source node

(Fig. 4(b)). In this case, the fault information in the west direction is checked before those of north and south directions. When the packet is northward (Fig. 4(c)) and the north link is faulty, the east direction is checked earlier than the west direction. It means that the west direction is used only when the faulty link is located in the rightmost border. A similar perspective is applied to southward packets (Fig. 4(d)). It is worth mentioning that for the northward and southward packets, only the first virtual channel is used unless the packet is generated at one of the nodes in the rightmost border (i.e. the second virtual channel is utilized).

C. *Adaptive Fault-Tolerant Algorithm*

A deterministic routing algorithm is a common method used in traditional fault tolerant methods since the path of a packet is predictable. However, in our algorithm we use the deterministic routing only when a packet gets close to the area of faulty link. Based on MD, if the distance from the current to destination node is greater than one hop along both directions, packets can adaptively choose among the non-faulty links without any restriction. According to DyXY, a packet is sent in a direction which has a more number of free slots in the corresponding input buffer. When both directions have the same number of free buffer slots, a direction is chosen by random. On the other hand, MD tries not to reduce the distance in one dimension to zero while the distance along the other dimension is greater than one. To step toward this goal, we try to keep the distances along both directions as equally as possible. However, forcing a packet to choose a specific direction is against the adaptive characteristic. Our solution to this issue relies on sending a packet in a desired direction (greatest-distance dimension) whenever the number of free buffer slots are nearly equal in both directions. The equality situation might happen in a few cases, but as it is obvious the number of free buffer slots does not need to be exactly the same. For instance, four or five free buffer slots out of eight available slots can be seen similar in terms of affecting the performance. Therefore, when the difference between the number of free slots in two input buffers is less than or equal to two flits, the packet is sent in the desirable direction.

An example is shown in Fig. 5(a), when the source and destination are located at nodes S and D while the link (11,D) is faulty. As can be seen by the figure, there are several paths that can be taken by MD. The simple rule is that the distance along each direction should remain greater than or equal to one. The number of free buffer slots is used to select among the output channels at each router. As illustrated in Fig. 5(b), the adaptivity options of the non-faulty case are nearly similar to the faulty case (Fig. 5(a)).

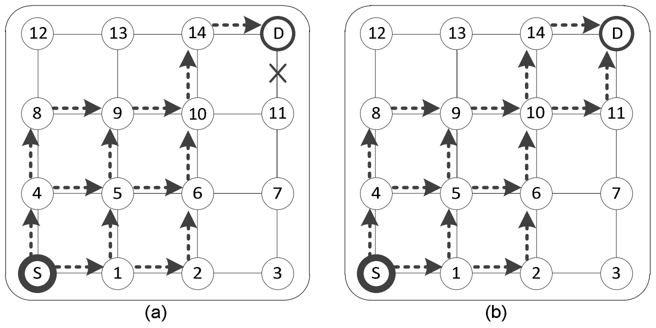


Fig. 5. Alternative paths from source node S to destination D

Fig. 6 shows an example of comparing MD with traditional methods which are based on contour strategy. Both methods select the output channels based on the congestion condition and fault information. Let us assume that the detour-based method knows only about the fault information in its direct links. In this figure, a packet is sent from the source node S to the destination node D when the links (S,1), (7,11), (8,9), (8,12), and (14,D) are faulty. This example shows that MD can support some multiple faulty links still using the shortest paths. At the source node, the north direction is selected by both methods since the east link is faulty. At the node 4, the detour-based method (Fig. 6(a)) may deliver the packet in the north direction (unaware of the faults in two-hop links) where the packet faces the faulty links and has to be returned to the node 4. If the packet arrives at the node 6, it might select the intermediate node 7 as the next hop. However, the node 7 has a faulty link in the north direction and returns the packet to the node 6. The packet might reach the node 14 where the east link is faulty and thus it has to be rerouted. In sum, the packet takes several unnecessary non-minimal paths which increase the latency. In contrast, at the node 4, MD (Fig. 6(b)) is aware of the faulty links at the NN and NE paths, so it delivers the packet in the east direction instead of the north direction. By arriving the packet to the node 6, MD avoids reducing the distance to zero along the X dimension (also it notices that the EN path is faulty), so it delivers the packet to the node 10. At this node, the packet is sent through the east direction since it knows that the NE path is faulty.

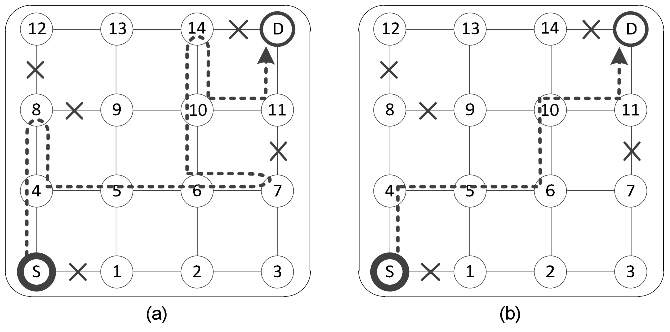


Fig. 6. Comparison of (a) a detour-based method with (b) MD

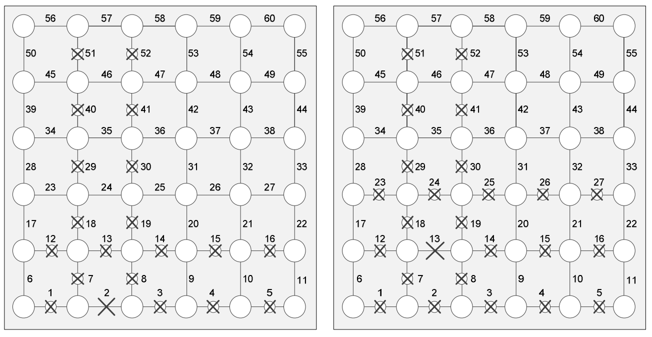


Fig. 7. Robustness analysis, specified links must be healthy

D. Analytical Analysis of Two-Faulty Situations

Since MD could not support all two-faulty cases, we take advantage of analytical models to evaluate the reliability of MD when two faults occur in the network. Let us assume that the location of faults is chosen randomly. We divide the problem into two cases: the first fault occurs on the border links or it occurs in central links. If the first fault occurs on one of the border links, the second fault must not happen on some specific locations. For example, in Fig.7 (a), if the link 2 is faulty, the specified locations in the figure must be healthy. A similar situation exists for all borderline links (i.e. 4(n-1) in n×n network). The probability that the first faulty link occurs on the border links and the second one does not locate in specific locations is calculated as follow:

$$\frac{4(n-1)}{2n(n-1)} * \left(1 - \frac{3*(n-1)+(n-2)}{2n(n-1)}\right) = \frac{2}{n} * (1 - \frac{4n-5}{2n(n-1)})$$

Similarly, for the case of Fig.7(b), the probability is obtained by the following formula:

$$\frac{2(n-1)(n-2)}{2n(n-1)} * \left(1 - \frac{4*(n-1)+(n-2)}{2n(n-1)}\right) = \frac{(n-2)}{n} * (1 - \frac{5n-6}{2n(n-1)})$$

According to these formulas, Table I shows the robustness when two faults occur in different network sizes. It is worth mentioning that, all packets are still routed adaptively in the network.

Table I. Robustness for 2-faulty links under different network sizes

Network size	Robustness for 2-faulty links
4×4	48%
6×6	63%
8×8	71%
16×16	85%

IV. Experimental Results

To evaluate the efficiency of the proposed routing scheme, 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 on 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 request rate is defined as the ratio of the successful message injections into the network over the total number of injection attempts. The simulator is warmed up for 12,000 cycles and then the average performance is measured over another 200,000 cycles. To have a fair comparison, we defined our baseline as a detour strategy similar to [16]. Like MD, the baseline method has two virtual channels along X and Y directions, respectively. Unlike MD, the baseline method may take unnecessary longer paths as discussed but it is able to support all one-faulty links.

A. Performance Analysis under Uniform Traffic Profile

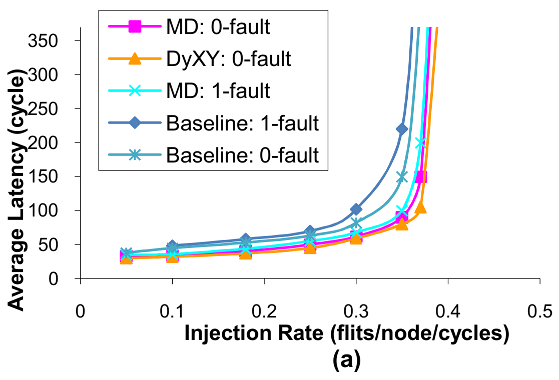
In the uniform traffic profile, each processing element (PE) generates data packets and sends them to another PE using a uniform distribution [21]. The mesh size is considered 4×4. In Fig. 8(a), the average communication latencies of MD, the baseline method and DyXY are measured for a fault-free case. In addition, the latencies of the MD and baseline methods are compared in a one-faulty link case. As observed from the results, in a fault-free case, DyXY performs better than MD. The reason is that DyXY method is a fully adaptive routing algorithm while MD limits packet adaptivity when packets get close to the destination node. The baseline method shows larger latency since it is a deterministic method. In a one-faulty case, MD performs better than the baseline method. This is due to the fact that MD can route packets through the shortest paths while in the baseline method, packets may take longer paths when facing a faulty link.

B. Performance Analysis under Hotspot Traffic Profile

Under the hotspot traffic pattern, one or more nodes 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 node with an additional *H* percent probability. We simulate the hotspot traffic with a single hotspot node at (2, 2) in 4×4 2D mesh. The performance of the MD, the baseline method, and DyXY is also measured for fault-free and one-faulty (except DyXY) link cases. The performance of each network with *H = 10%* is illustrated in Fig. 8(b). As observed from the figure, in the hotspot traffic and in both faulty and non-faulty cases, the performance improvement of MD is better than the detour-based scheme.

C. 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 MD with the baseline and DyXY methods [23]. The power consumption of DyXY is measured only in the fault-free case while the other two methods tolerate one faulty link. Each scheme includes network interfaces, routers, and communication channels. 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 2D mesh. The layout area and power consumption of each platform are shown in Table II. As indicated on the table, MD has a larger area overhead than DyXY and a lower one than the baseline method. It is because of using a simpler routing unit at the DyXY method and a more complex one in the baseline method. As indicated on the table even if the MD has to support a one-faulty link (while DyXY is fault-free), the power consumption of MD remains relatively small. This is due to the fact that MD could route packets through the shortest paths and thus consuming less power.



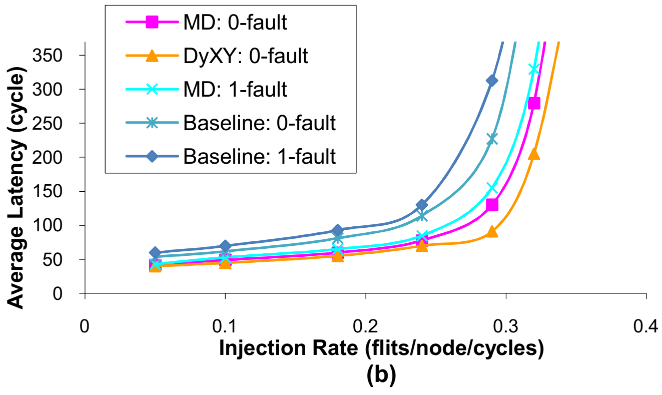


Fig. 8. Performance analysis of MD and the baseline method in 4×4 mesh network (a) under uniform traffic profile (b) hotspot traffic profile in fault-free, 1-faulty link cases.

Table II. Details of hardware implementation

Network platforms	Area (mm^2)	Power (W) dynamic & static
DyXY	6.710	2.32
MD	6.753	2.39
baseline	6.913	2.78

V. CONCLUSION

In this paper, a fault-tolerant routing algorithm named MD was presented using two virtual channels along both directions. The presented algorithm avoids taking unnecessary non-minimal paths when any minimal paths are available, resulting in significant performance hits. This improvement achieves by a new fault's information propagation mechanism and utilizing the information to deliver packets through the shortest paths. Moreover, MD is able to deliver packets through alternative paths to the destination, thereby alleviating congestion in the network both in fault-free and faulty conditions. Finally, we perform robustness analysis based on the presented analytical model for two faults in the network.

REFERENCES

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

[2] W. Tsai et al., "A fault-tolerant NoC scheme using bidirectional channel", in Proc. DAC, pp.918-923, 2011.

[3] M. Daneshtalab et al., "Adaptive input-output selection based on-chip router architecture," Journal of Low Power Electronics (JOLPE), Vol. 8, No. 1, pp. 11-29, 2012.

[4] 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. DATE'03, pp. 350-355, 2003.

[5] M. Cuviello et al., "Fault modeling and simulation for crosstalk in system-on-chip interconnects", in Proc. ICCAD, pp. 297-303, 1999.

[6] M.H Neishaburi et al., "HW/SW architecture for soft-eError cancellation in real-time operating system," Journal of the Institute of Electronics, Information and Communication Engineers, Vol. 4, No. 23, pp. 755-761, 2007.

[7] J. Duato et al., "Interconnection networks: an engineering approach", Morgan Kaufmann Publishers, 2003.

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

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

[10] L.M. Ni et al., "A survey of wormhole routing techniques in direct networks", in Proc. IEEE Computer, v.26, I.2, pp.62-76, 1993.

[11] D. Fick et al., "Vicis: a reliable network for unreliable silicon", in Proc. of Design Automation Conference, pp. 812-816, 2009.

[12] S. Chalasani et al., "Fault-tolerant wormhole routing algorithms for mesh networks", IEEE Trans on Computers, 44(7):848–64, 1995.

[13] PH. Sui et al.," An improved algorithm for fault-tolerant wormhole routing in meshes", IEEE Trans on Computers x;46(9):1040–2, 2011.

[14] S. Park et al., "Fault-tolerant wormhole routing algorithms in meshes in the presence of concave faults", in Proc. of International Parallel and Distributed Processing Symposium (IPDPS), pp. 633–8, 2000.

[15] Z. Zhang et al., "A reconfigurable routing algorithm for a fault-tolerant 2D-mesh Network-on-Chip", in Proc. DAC, pp. 441-446, 2008.

[16] M. Valinataja et al.,"A reconfigurable and adaptive routing method for fault-tolerant mesh-based networks-on-chip", in Proc. International Journal of Electronics and Communications (AEU), v. 65, I.7, pp. 630-640, 2011.

[17] M. Koibuchi et al., "A lightweight fault-tolerant mechanism for network-on-chip", in Proc. NOCS, pp.13-22, 2008.

[18] M. Ebrahimi et al., "MAFA: adaptive fault-tolerant routing algorithm for networks-on-chip," in Proc. DSD, pp. 201-206, 2012..

[19] 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.

[20] D. Fick et al., "A highly resilient routing algorithm for fault-tolerant NoCs", in Proc. DATE, pp. 21-26, 2009.

[21] C.J. Glass et al., "The turn model for adaptive routing", in Proc. 19th Int'l Symp. Computer Architecture, pp. 278-287, 1992.

[22] M. Ebrahimi et al., "An efficent dynamic multicast routing protocol for distributing traffic in nocs," in Proc. of 12th ACM/IEEE DATE, pp. 1064 - 1069 , April 2009.

[23] M. Li et al., "DyXY - A proximity congestionaware deadlock-free dynamic routing method for Network on Chip", in Proc. DAC, pp. 849-852, 2006.