A Routing-Level Solution for Fault Detection, Masking, and Tolerance in NoCs

Xiaofan Zhang[1], Masoumeh Ebrahimi[2,3], Letian Huang[1], Guangjun Li[1], Axel Jantsch[4]
[1] University of Electronic Science and Technology of China, China
[2]KTH Royal Institute of Technology, Sweden
[3]University of Turku, Finland
[4]Vienna University of Technology, Austria

***Abstract*— Faults may occur in numerous locations of a router in a NoC platform. Compared with the faults in the data path, faults in the control path may cause more severe effects which may result in crashing the entire system. Most of the current efforts in literature focus on disabling a router when a fault is detected. Considering this level of coarse-granularity, the functioning parts of a router have to be unnecessarily disabled which may severely affect the performance or functionality of the on-chip network. To cope with this problem, in this paper we propose a mechanism to tolerate faults in the control path which largely avoid disabling a router as long as the fault is not severe. This mechanism is called DMT, standing for three distinguishing characteristics of the proposed method as fault Detection, fault Masking and fault Tolerance. The proposed mechanism can efficiently detect the faults expressed as illegal turns while it has the capability to tolerate faults without a prior knowledge on where and why a fault has happened.**

Keywords-Network-On-Chip; fault-tolerance;

I. INTRODUCTION

As a competitive solution in communication structure, Network-on-chip (NoC) is a preferable choice of infrastructure to connect hundreds of processing elements to each other in a complex Systems-on-Chip [1][2]. In NoC, resources are largely distributed and shared, meaning that packets may use several links and routers in different parts of the network. In spite of these advantages, the drawback is that a single fault in NoC may result in different types of router malfunctions such as miscalculation in the routing computation, conflict in the arbitration, mismatches in the crossbar, and reading from an empty buffer [3]. NoC routers generally consist of five stages as routing computation (RC), virtual channel allocation (VA), switch allocation (SA), crossbar (Xbar), and link traversal (LT). 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 faults occur in the data path, the most common and obvious phenomenon is that faults affect the data carried by packets such as a bit flip caused by the low noise margin [4], the electromagnetic coupling effects [5], or the crosstalk [6]. Fortunately these faults can be easily detected by the receiving nodes using Error Correcting Code (ECC) and can be solved by data redundancy mechanism like retransmission. 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, 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 leading to illegal turns. This solution provides real-time and on-line fault detection, called DMT, covering the detection, masking and tolerating of illegal-turn faults. To achieve these goals, a non-minimal routing algorithm is designed and faults are classified into severe and ignorable. More importantly, the additional hardware for fault detection is very lightweight. Compared with the complicated components of the control path, the detection parts achieve much higher reliability to ensure the reasonable detection results.

The reminder of this paper is organized as follows. In Section II, related work is given. In Section III, the simple idea of the proposed mechanism is presented. In Section IV, the components of the DMT are introduced. The analytical and experimental results are reported in Section V while the summary and conclusion are given in the last section.

II. RELATED WORK

Fault-tolerance in NoC is more than a feature but a necessity for achieving higher reliability because of the increasing design complexity and the continued shrinkage of semiconductor process feature sizes [7]. The functionality of on-chip network can be highly affected by power supply noise, ground bounce, energetic particles and interconnection noise such as crosstalk and electromagnetic interference. These faults, considered as transient faults, occur randomly and do not cause physical damages on circuits.

Most of the current efforts in literature focus on disabling a router when a fault is detected [8][9]. In other efforts, non-minimal routing algorithms have been investigated aiming to

achieve fault-tolerant methods by enabling packets to turn around the faulty areas [10][11]. In [8], an algorithm is proposed 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 status of eight direct and indirect neighbouring routers. The DBP approach [9] 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 contrast, HiPFaR [12] targets tolerating faulty routers by avoiding non-minimal paths as along as possible. These proposed methods are based on a common assumption that the entire router is disabled in the case of a single fault. They try to keep the network working by disconnecting both the faulty router and the core from the network. 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).

Some other literatures are susceptible to failures during an initial phase of transmission where every intermediate node drops packets thus thwarting the transmission altogether [13]. These methods targets special types of faults or specific router components. In addition, the capability of tolerating faults is based on the prior knowledge on where and why a fault has occurred.

In this paper, we introduce a mechanism with the capability of fault detection, the non-minimal routing support, the fault classification and the fault tolerance which are able to improve the reliability of on-chip network. Routers are no longer necessarily disabled when faults occur. By applying the proposed mechanism, latency of the network are close to those of using DyXY routing algorithm [14] (which is considered 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.

III. The Simple Idea of DMT

Using the example of Fig. 1, we explain the simple idea of the proposed mechanism, DMT. Let us assume that a packet is sent from the source 0 to the destination 8. The packet is currently at the router 4 arrived from its south input port and the virtual channel 1 (S1). Since the destination is to the NE position of the router 4, the possible minimal directions are as:

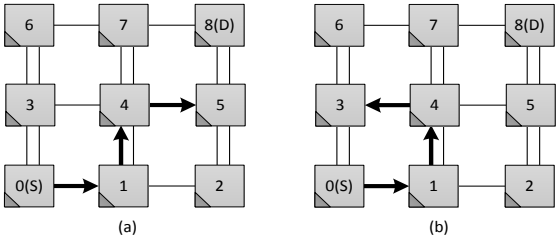

Fig. 1 Example of the proposed mechanism

N1, N2, and E where among them the E port is selected in the example of Fig. 1 (a). However, due to faults at the control path of the router 4, the packet may be delivered to the router 3 instead of the router 5 as shown in Fig. 1 (b). Since the algorithm does not support non-minimal paths, the packet is blocked at the router 3 which results in blocking other packets and finally crashing the whole system. DMT enables the router 3 to detect the fault at the router 4 by checking the possibility of receiving a packet taking a North-West turn while having the destination at the NE position. It is worth mentioning that non-minimal routing algorithms are commonly used to turn around the disabled faulty router. However, in DMT, non-minimal routing algorithms are used for a different purpose, i.e. avoid disabling a router for as long as possible.

If due to faults, the turn taken by the packet leads to a minimal direction (e.g. the packet is delivered from the router 4 to the router 7 instead of the router 5), the fault is masked and no further attention is needed as the fault is prevented from introducing errors. On the other hand, we design a non-minimal routing algorithm to tolerate the faults that lead to non-minimal directions which cannot be masked (Fig. 1 (b)). Based on the capability of the algorithm in delivering packets over different output channels, the faults are categorized as severe or ignorable.

IV. The Proposed Mechanism

A. Fault Detection

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 whether the upstream router has performed the routing correctly or not. However, no routing recalculation is done in the receiving router. Taking the example of Fig. 1 (b) where a minimal routing algorithm is running, fault at the router 4 will be detected at the router 3. To detect the correctness of the routing decision made in the router 4, two parameters are needed at the router 3 as: the input port in which the packet is received at the router 4 and the destination position of the packet regarding the router 4. As shown in Fig. 2, the input port and the virtual channel number are transferred using 3 extra bits along with the packet. The destination position of the packet can be easily obtained using the same logic as in the routing unit without any extra hardware and effect on the critical path of the routing computation unit. This step can be done in parallel with the output selection part in the routing computation unit, shown in Fig. 2 which implies that the fault detection circuit will not affect the critical path. In sum, the algorithm follows a minimal and fully adaptive algorithm, similar to DyXY and for detecting faults, the algorithm shown in Fig. 3 is used.

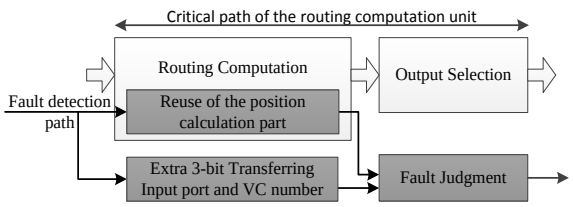

Fig. 2 Block diagram of the fault detection

```
(Xc,Yc): X and Y coordinate of the current router;
IF InCh (Xc,Yc) = {E}
   IF InCh (Xc+1,Yc) = {L,N1,S1,E} THEN
     "Fault is Ignorable"  ELSE  "Severe";
IF InCh (Xc,Yc) = {W}
   IF InCh (Xc-1,Yc) /= {E} AND Pos((Xc-1,Yc) and (Xd,Yd))=
     {E,NE,SE} THEN
     "Fault is Ignorable"  ELSE  "Severe";
IF InCh (Xc,Yc) = {N1}
   IF InCh (Xc,Yc+1) = {L,S1,E} THEN
     "Fault is Ignorable"  ELSE  "Severe";
IF InCh (Xc,Yc) = {N2}
   IF InCh (Xc,Yc+1) /= {S2} AND Pos((Xc,Yc+1) and (Xd,Yd))=
     {N,E,NE,SE} THEN
     "Fault is Ignorable"  ELSE  "Severe";
IF InCh (Xc,Yc) = {S1}
   IF InCh (Xc,Yc-1) = {L,N1,E,S1} THEN
     "Fault is Ignorable"  ELSE  "Severe";
IF InCh (Xc,Yc) = {S2}
   IF InCh (Xc,Yc-1) /= {N2} AND Pos((Xc,Yc-1) and (Xd,Yd))=
     {S,E,NE,SE} THEN
     "Fault is Ignorable"  ELSE  "Severe";
```

Fig. 3 The fault detection algorithm

B. Classifying Faults into Ignorable and Severe

We classify faults into two groups as ignorable faults and severe faults depending on whether the turn taken by a packet is among the allowable turns or not.

The ignorable fault is the fault that can be tolerated by our proposed mechanism. When faults occur in the control path, one of the most obvious cases is that a packet is sent to a wrong port which eventually may lead to deadlock and livelock in NoC platforms. Fortunately, by applying the proposed mechanism, some illegal turns in the network are still supported. It means that the packet experiences a wrong turn(s) but eventually arrives at the right destination node by taking advantage of the flexibility of the non-minimal routing algorithm.

The severe fault is the fault that cannot be supported by our mechanism because of the rules of the non-minimal routing in which some turns are illegal. Under this circumstance, faults cannot be tolerated and the affected packet must be dropped in order to avoid deadlock or livelock.

In general, DMT introduces a new perspective to non-minimal routing algorithms. In traditional approaches, non-minimal routing algorithms are used to reroute packets around the faulty router or region. Thereby, the basic assumption is that the whole router is disabled due to faults. However, in DMT, the flexibility of non-minimal routing is used to avoid disabling the faulty router. The non-minimal algorithm in this paper is used as a case study while more flexible algorithms can be designed such that to eliminate more types of faults even those of classified as severe in the proposed solution.

V. RESULT AND DISCUSSION

We evaluate the proposed mechanism in terms of average latency and reliability. The average latency is defined as the average time takes for packets to reach from a source node to a destination node. When faults occur in the control path, turns may be illegal and some packets may never reach the destination. Therefore, the survival rate is a vital factor to measure the fault-tolerant capability, which defined as the ratio of the number of packets successfully reach the destinations over the total number of packets. These experiments are performed on a 2D 8×8 mesh network using wormhole switching with a constant packet size of 4 flits and different packet injection rates. The simulator is cycle-accurate implemented with VHDL. It is able to generate different traffic patterns such as Uniform, Transpose1, Transpose2, and Bit-Reversal traffic. The difference between Transpose1 and Transpose2 is in the orientation of choosing source and destination nodes. DMT 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 DMT faults are injected in the form of taking a wrong turn inside a router to simulate faults expressed as illegal turns. In DMT, for example fault injection rate of 5% means that at every router in the network 5% of all turns are chosen differently as decided by the routing unit. This assumption is pessimistic and faults may happen with a lower probability. Since the nature of the DMT approach is unique and the form of injecting faults is specific, there is no equivalent fault-tolerant method to be compared with. In simulations, DMT is compared with DyXY which is a fully adaptive routing algorithm using the same number of virtual channels but supporting only the minimal directions [14].

A. Performance Evaluation

The performance of DMT and DyXY methods are compared with each other from Fig. 4 to Fig. 6 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. 4 when there is no fault in the whole network. As can be seen in this figure, the average latency of the proposed mechanism is nearly the same as those using DyXY. The reason is that, in fault-free conditions, both methods follow the minimal paths.

Fig. 5 shows the performance results when 3% of turns are faulty in all the routers of the network. By comparing the equivalent curves of Fig. 4 and Fig. 5, it can be observed that the saturation point of the DMT method remains almost the same in both figures. However, the saturation point of DyXY improves in Fig. 5. This can be explained that DyXY cannot tolerate faults and thus packets are dropped from the network. It results in less congestion and thus a better performance. Even though the DyXY apparently seems to work better than DMT in Fig. 5. The latency of DMT and DyXY methods with the fault but in reality it is because of its inability to tolerate faults.

In order to observe the impact of a higher injection rate, we increase the injection rate to 5% in Fig. 6. The results show that the performance of the network is still at the same level in DMT even though many turns are taken wrongly due to faults. On the other hand, the performance of DyXY keeps growing, implying that more and more packets are dropped in the network.

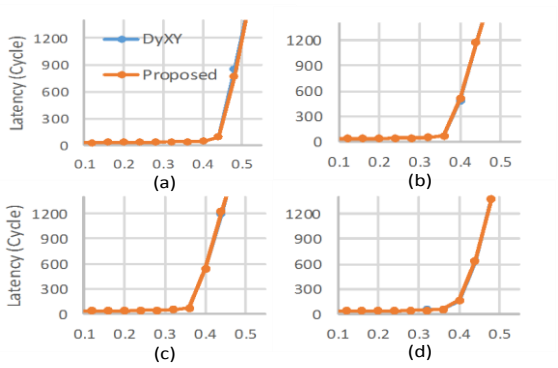

Fig. 4. The latency of DMT and DyXY methods in fault-free cases and under four traffic patterns as (a) Uniform, (b) Transpose1, (c) Transpose2 and (d) Bit-Reversal

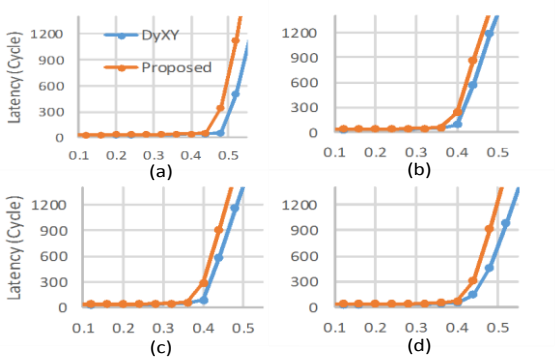

Fig. 5 The latency of DMT and DyXY methods with the fault injection of 3% and under four traffic patterns as (a) Uniform Traffic, (b) Transpose1 Traffic, (c) Transpose2 Traffic (c) and Bit-Reversal Traffic (d))

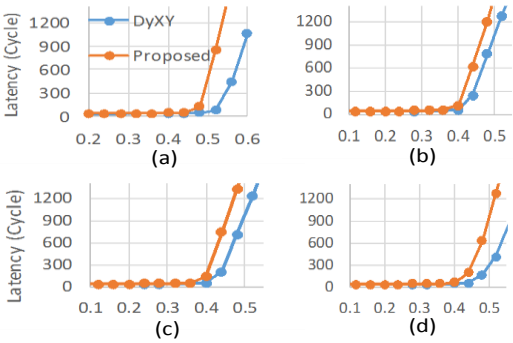

Fig. 6 The latency of DMT and DyXY methods with the fault injection of 5% and under four traffic patterns as (a) Uniform Traffic, (b) Transpose1 Traffic, (c) Transpose2 Traffic (c) and Bit-Reversal Traffic (d))

Table 1. Area overhead and power consumption

	Proposed Mechanism	Router with DyXY	Additional Overhead
cell area	31255	29502	5.9%
cell power	444.967uW	416.525uW	6.8%

B. Fault-Tolerant Capability

In order to evaluate the reliability of DMT, we simulate the faults which occur in the control path of a router. Thereby, we force packets to take a random turns despite the decision of the routing computation unit. This operation is similar to the behaviours when a fault occurs in the circuit level of routers such as stuck-at-1 and stuck-at-0 and eventually faults lead to illegal turns.

In this set of experiment, we evaluate the reliability of the network for different fault injection percentage and for different traffic patterns. The fault injection percentages are selected to be 0%, 0.5%, 1%, 1.5% and 2%. Since the number of dropped packets exceeds 10% of the total number of packets, we stop the fault injection percentage at 2%. The results are shown from Fig. 7 to Fig. 10. The statistical information in these experiments includes the total number of packets which can be successfully received by the destination (shown along y-axis) and the fault injection percentages (from 0% to 2% along x-axis). DMT has the 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 Fig. 7 to Fig. 10, DMT is able to survive about 2839, 2291, 3276, and 2987 packets, respectively, more than the DyXY algorithm with 2% fault injections. Although in this paper we introduce a new way of utilizing non-minimal routing, there are several ways for more improvement as 1- designing a more flexible non-minimal algorithm 2- combining this method with ordinary way of using the non-minimal algorithm so that if a router is detected as creating severe faults repeatedly, it can be disabled.

C. Hardware Overhead

To assess the area overhead and power consumption, an on-chip network router with the proposed mechanism and a general one using DyXY routing algorithm are synthesized using Synopsys Design Compiler. We compared the area overhead and power consumption of a router in both models. For synthesizing, we use the TSMC45nm technology at the operating frequency of 1GHz and supply voltage of 0.9V. As indicated in Table 1, the power consumption and area overhead of the router with the proposed mechanism and the general one are comparable. With a negligible hardware overhead, the proposed algorithm can offer a more reliable no-chip network.

VI. CONCLUSION

We proposed a mechanism to tolerate faults which can be expressed as illegal turns in the control path, called DMT. It is an efficient process with low latency and hardware overhead including fault detection, non-minimal routing, fault classification and fault tolerance. DMT introduces a new way of employing non-minimal algorithms in NoCs and taking a better advantage of them. By applying the proposed mechanism, a router is not necessary disabled when faults occur. The extra hardware resources to implement the algorithm do not affect the critical path of the router. 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 algorithm leads to less packet drops as illegal turn faults will be supported by the non-minimal algorithm.

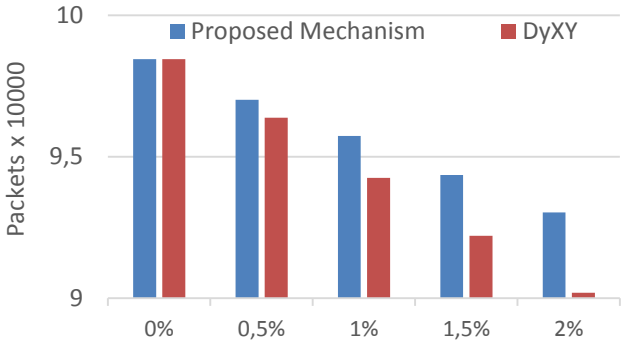

Fig. 7 Fault-tolerant capability under uniform traffic

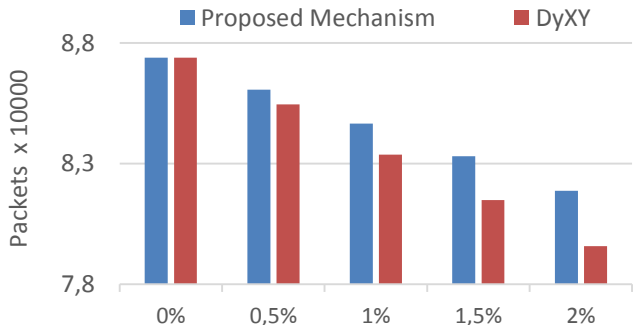

Fig. 8 Fault-tolerant capability under Transpose1 Traffic

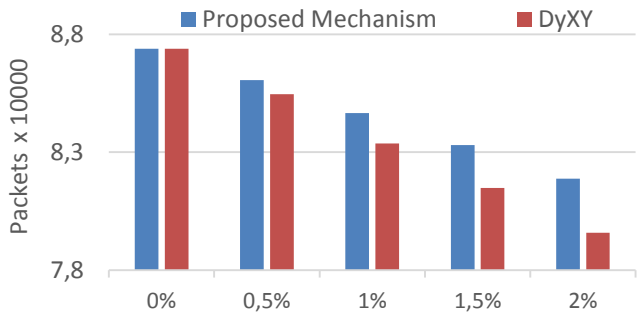

Fig. 9 Fault-tolerant capability under Transpose2 Traffic

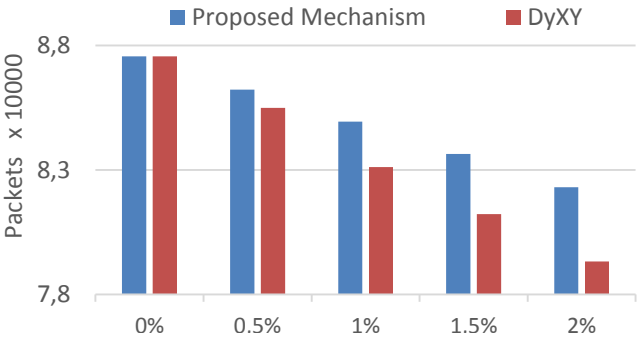

Fig. 10 Fault-tolerant capability under Bit-Reversal Traffic

ACKNOWLEDGMENT

This work was supported by the NSFC under grant No.61176025 and No.61006027.

REFERENCES

[1] Jantsch, Axel, and Hannu Tenhunen, eds. "Networks on chip". Vol. 396. Dordrecht: Kluwer Academic Publishers, 2003.

[2] M. Daneshtalab et al., "A Low-Latency and Memory-Efficient On-hip Network," in Proceedings of 4th International Symposium on Network-on-Chip (NOCS), pp. 99-106, May 2010, France.

[3] A. Prodromou, A. Panteli, C. Nicopoulos, and Y. Sazeides, "NoCAlert: An On-Line and Real-Time Fault Detection Mechanism for Network-on-Chip Architectures," in Proc. of Micro, pp. 60-71, 2012.

[4] Suwen Yang, M. Greenstreet, "Noise margin analysis for dynamic logic circuits," iccad, pp.406-412, 2005 International Conference on Computer Aided Design (ICCAD'05), 2005

[5] Erdin, Ihsan, Michel S. Nakhla, and Ramachandra Achar. "Circuit analysis of electromagnetic radiation and field coupling effects for networks with embedded full-wave modules." Electromagnetic Compatibility, IEEE Transactions on 42.4 (2000): 449-460.

[6] Aniket, Ravishankar Arunachalam, "Novel Algorithm for Testing Crosstalk Induced Delay Faults in VLSI Circuits," vlsid, pp.479-484, 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design (VLSID'05), 2005

[7] L. Benini and G. De Micheli, "Networks on Chips: Technology and Tools," Morgan Kauffmann, 2006.

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

[9] 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.

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

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

[12] M. Ebrahimi, M. Daneshtalab, J. Plosila, "High Performance Fault-Tolerant Routing Algorithm for NoC-based Many-Core Systems", in Proc. PDP, pp. 462-469, 2013.

[13] M. Pirretti, G. M. Link, et al., "Fault tolerant algorithm for network-on-chip interconnect," Symposium on VLSI, 2004

[14] M. Li, Q.A. Zeng, et al., "DyXY – a proximity congestion-aware deadlockfree dynamic routing method for network on chip," Proc. DAC, pp. 849-852, 2006.

[15] M. Ebrahimi, M. Daneshtalab, P. Liljeberg, J. Plosila, and H. Tenhunen, "LEAR – A Low-weight and Highly Adaptive Routing Method for Distributing Congestions in On-Chip Networks," in Proc. of PDP, pp. 520-524, 2012