ERFAN: Efficient Reconfigurable Fault-Tolerant Deflection Routing Algorithm for 3-D Network-on-Chip

Somayeh Maabi[1], Farshad Safaei[1], Amin Rezaei[2], Masoud Daneshtalab[3], Dan Zhao[4]
[1] Faculty of Computer Science and Engineering, Shahid Beheshti University (SBU), Tehran, Iran
(s_maabi@sbu.ac.ir, f_safaei@sbu.ac.ir)
[2] University of Louisiana at Lafayette (ULL), Lafayette, USA (me@aminrezaei.com)
[3] Mälardalen University (MDH) and Royal Institute of Technology (KTH), Sweden (masdan@kth.se)
[4] Old Dominion University (ODU), Norfolk, USA

***Abstract*— With degradation in transistors dimensions and complication of circuits, Three-Dimensional Network-on-Chip (3-D NoC) is presented as a promising solution in electronic industry. By increasing the number of system components on a chip, the probability of failure will increase. Therefore, proposing fault tolerance mechanisms is an important target in emerging technologies. In this paper, two efficient fault-tolerant routing algorithms for 3-D NoC are presented. The presented algorithms have significant improvement in performance parameters, in exchange for small area overhead. Simulation results show that even with the presence of faults, the network latency is decreased in comparison with state-of-the-art works. In addition, the network reliability is improved reasonably.**

Keywords—3-D NoC, Fault Tolerance, Deflection Routing Algorithm, TSV, Reliability

I. INTRODUCTION

In many-core era, there exists a need for easily scalable, high-performance and low-power intra-chip communication infrastructure specially for emerging systems. In Many-Core System-on-Chips (MCSoCs) with high processing rate, Network-on-Chips (NoCs) are considered to be the optimal way to manage the exchange of data between the Processing Elements (PEs) [1]. Despite the fact that NoC has a lot of advantages, because of the presence of long wires in two-sided metal interconnects, it suffers from high latency and high power consumption [2]. Therefore, alternative technologies such as, 3-D NoC were introduced. In 3-D integration technology multiple dies are stacked on a single chip by using Through Silicon Vias (TSVs). 3-D chips provide complicated applications with significant improvement in performance, cost, and time-to-market availability. The long horizontal wires in a 2-D design can be replaced by shorter and more efficient vertical wires, leading to lower interconnect delay and power consumption [3]. Moreover, 3-D NoCs overcome the limited scalability of 2-D NoCs over 2-D planes by using short and fast vertical interconnects of 3-D ICs [4]. Also 3-D NoCs greatly reduce the network diameter and overall communication distance [5].

On the other hand, dimension degradation and complexity increment, make 3-D NoCs more vulnerable to faults. Therefore, introducing fault-tolerant mechanisms is essential in 3-D NoCs. By designing fault-tolerant routing algorithms, the reliability of NoCs will increase. However, most of the fault-tolerant routing algorithms are offered for 2-D NoCs while those have been presented for 3-D routing are in fact an extension from 2-D to 3-D [6]. Recently, buffer-less routing, also called deflection routing algorithms, has been widely used for reducing the hardware overhead and power consumption in NoCs. In deflection routing, an incoming packet is always routed to a free output port even though it is far away from the destination. Because of its non-minimal routing characteristic, deflection routing can be easily modified to achieve fault tolerance [7].

In this paper two Efficient Reconfigurable Fault-tolerant routing Algorithms for 3-D NoCs, ERFAN, are proposed in order to tolerate both static and dynamic permanent faults while reduce network latency and power consumption. The characteristics of the algorithms are as follows:

- Instead of one global table for the whole network, each router is equipped with a table contains the hop-count level of its layer and two separate fault tables for faulty horizontal links and faulty TSV links;
- The channels are buffer-less since the combination of minimal and deflection routing makes the proposed algorithms deadlock free;
- Routing table does not require the z-dimension field, because packets are first routed in the third dimension.

The rest of the paper is organized as follows: Section II reviews the related work and briefly analyzes some well-known routing algorithms. The proposed 3-D NoC architecture and the fault-tolerant routing algorithms are explained in Section III and IV respectively. Section V evaluates the performance of the proposed algorithm and compares it with state-of-the-art algorithms. Finally, the conclusion and future works are given in Section VI.

II. RELATED WORKS

Fault-tolerant routing algorithms are divided into three different categories. The first approach sends redundant packets to different decisive or partial adaptive routes. If the main packet faces any faulty links, the redundant packet still has the possibility of reaching to the destination [7].

978-1-5090-1367-8/16/$31.00 ©2016 IEEE

TABLE I. Routing table in a layer of 3×3×3 3-D NoC

	0	1	2	3	4	5	6	7	8
	N E S W	N E S W	N E S W	N E S W	N E S W	N E S W	N E S W	N E S W	N E S W
0	-1 2 -1 2	-1 1 3 -1	-1 2 4 -1	-1 3 1 -1	-1 2 2 -1	-1 3 3 -1	-1 4 2 -1	-1 3 3 -1	-1 4 4 -1
1	-1 3 3 1	-1 2 2 2	-1 1 3 3	-1 4 2 2	-1 3 1 3	-1 2 2 4	-1 5 3 3	-1 4 2 4	-1 3 3 5
2	-1 -1 4 2	-1 -1 3 1	-1 -1 2 2	-1 -1 3 3	-1 -1 2 2	-1 -1 1 3	-1 -1 4 4	-1 -1 3 3	-1 -1 2 4
3	1 3 3 -1	2 2 4 -1	3 3 5 -1	2 2 2 -1	3 1 3 -1	4 2 4 -1	3 3 1 -1	4 2 2 -1	5 3 3 -1
4	2 4 4 2	1 3 3 3	2 2 4 4	3 2 3 1	2 2 2 2	3 1 3 3	4 4 2 2	2 3 1 3	4 2 2 4
5	3 -1 5 3	3 -1 4 2	1 -1 3 3	4 -1 4 2	3 -1 3 1	2 -1 2 2	5 -1 3 4	4 -1 2 2	3 -1 1 3
6	2 4 -1 -1	3 3 -1 -1	4 4 -1 -1	1 3 -1 -1	2 2 -1 -1	3 3 -1 -1	2 2 -1 -1	3 1 -1 -1	4 2 -1 -1
7	3 5 -1 3	3 5 -1 4	3 3 -1 5	2 4 -1 2	1 3 -1 3	2 2 -1 4	3 1 -1 3	2 2 -1 2	3 1 -1 3
8	4 -1 -1 4	3 -1 -1 3	2 -1 -1 4	3 -1 -1 3	2 -1 -1 2	1 -1 -1 3	4 -1 -1 2	3 -1 -1 1	2 -1 -1 2

Authors in [8] proposed a fault-tolerant routing scheme for 3-D NoCs using hybrid turn models. In addition to energy consuming for sending redundant packets, the degree of fault tolerance in this method is not high, because both main and redundant packets may face faulty links simultaneously.

The second approach is using routing tables and choosing safe routes for sending packets [6, 9, 10]. However, this approach is not suitable for on-chip networks with limited areas as scaling the size of the network, considerably increases the size of the tables. But on the other hand the existence of routing table decreases the time of calculating the optimum route during routing.

The third approach is making faults convex or rectangular districts to route packets [11, 12, 13]. However, for making convex districts some components of the network as well as some operating nodes would became useless.

According to the above discussions and the necessity of considering fault tolerance, as well as avoiding deadlock and live-lock in routing algorithms, two fault-tolerant deflection routing algorithms with deadlock and live-lock avoidance are proposed for 3-D NoCs. When facing faults, the proposed algorithms are capable of changing route to the destination without waiting in buffers. Moreover, existence of TSV fault tables helps to understand the faulty status of specific TSVs before sending the packet. Also in contrast to previous works that used global table consist of all the nodes, we use local routing table for every layer nodes; thus, the size of the tables is considerably reduced.

III. 3D-NoC Architecture

The architecture of 3-D NoC is presented in this section. In this architecture four ports connect four adjunct switches in a layer. Also every switch has two additional vertical ports for communication with switches in up/down layers.

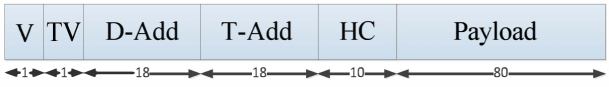

Fig. 1. Packet structure

A. Packet Structure

As shown in Fig. 1 each packet contains 48-bit header and 80-bit payload. The header includes five fields as follows:

Valid (V) is a single bit for packet validation

Destination Address (D-Add) is for destination address. Related addresses are updated during passing in a switch.

Temporary Address (T-Add) stores the intermediate address in (x, y, z). When a packet is about to route in up/down layer but the vertical link of the current switch is faulty, this field is used to send the packet to an intermediate switch with a safe (i.e. non-faulty) vertical link to up/down layer.

Temporary Validation (TV) bit validates the T-Add. When TV is set, the packet will be routed to the intermediate switch first.

Hop Count (HC) is a 10-bit field used as packet priority which records the number of hops the packet has been routed. The switch uses this field to make routing decisions for live-lock avoidance.

B. Routing Table

Each switch is equipped with a $n^2 \times 4$ table where n is the number of nodes in a layer. The table contains the number of hops to all destinations on the 2-D layer from four outputs ports: north (N), east (E), south (S) and west (W). Table I shows the routing table in a 3×3×3 3-D NoC. The table contains the minimum hop-counts from one node to the other nodes in a layer that is initiated by Equation (1).

$$H_t^s(p,y) = 1 + \min[H_{t-1}^s(p,y)] \qquad (1)$$

$H_t^s(p,y)$ represents the minimum hops from source node s to destination node p by neighbor node y. When s sends a packet to p using node y, the Equation (1) will return '1' plus the minimum number of hops from y to p in order to reconfigure the routing table with the entry of s. Existence of '1' in this equation is because of moving to a noteworthy

direction at first and then catching the destination distance to this node from the aforementioned direction.

Furthermore, Table I contains the information for every switch in addition to the number of hops from every node to the current switch in a layer. It contains the hop-counts in order to choose the shortest path during routing. Having routing information of other nodes is useful for bypassing the faulty link and reaching the destination from other nodes with the lowest distance to the destination. When there is no route from the current direction to the destination, '-1' is inserted in the routing table. This happens on border nodes of each layer. For example, in Table I the corresponding values of (N, E, S, W) for the element (5, 2) are respectively (1, -1, 3, 3) which '-1' represents that there is no route from the east direction from source node 5 to the destination node 2. Since the node 5 is on the right border of the layer, there is no route from this node to any other nodes using this direction.

C. Fault Table

For the packets routed through the layers, switch makes routing decisions according to the table of vertical link state. The size of this table is m×2 where m represents the number of nodes of the corresponding layer while there are two vertical up/down links in each node. This table records faulty up/down TSV links in current layer. A '1' in TSV fault table represents faulty TSV link. Also for simulating horizontal faulty links in a layer, one m×4 fault table for showing faulty state of four links in every switch is utilized A '1' in horizontal fault table represents faulty bidirectional link too.

D. Fault Model

In this work the faulty link is assumed as completely broken link (i.e. permanent faults). A faulty link for 3-D NoC is predicated as a vertical or horizontal link. It is assumed that a recognition fault mechanism [14] exists for fault detection.

IV. Proposed Routing Algorithms

In the proposed routing algorithms the number of remaining hops to the destination (i.e. HC field) is used as packet priority to adaptively route the packets. It is worth mentioning that every switch can handle six packets simultaneously. If more than one packet contend for one output port, the packet with higher priority will be selected while the others will be deflected through the next available output ports.

The algorithm is divided into two parts: routing within a layer and routing between the layers. A learning-based deflection routing algorithm is used to route packets within a layer. Each switch uses the routing table to route the packets from the ports with the lowest hop-count to the destination along the x or y axis. If there are two or more directions, a dynamic fault-tolerant routing algorithm called ERFAN-Dy will choose a port with the minimum traffic load which actually is the number of packets handled by neighboring switches. This balances the network traffic load. On in the other hand, in ERFAN-Sp the packet will be routed just along the last shortest path. If the packet faces a faulty horizontal link in the same layer, the current switch routes it to another remaining shortest path with the lowest traffic load using routing table. Otherwise the remaining free ports will be chosen. In fact, deflection routing is avoiding from blocked and busy routes.

On the other hand, for those packets routed between the layers, switch makes routing decisions according to TSV fault table. If the vertical link of the current switch is faulty, it will try to find an intermediate node with an operating vertical link in that layer which has the minimum distance to the current node. Then the packet will be routed to the intermediate node according to the routing table of that layer and will go to the destination layer from its vertical link. Checking the vertical link of the intermediate node, before routing, prevents the packet returning from the routed path while its TSV link is faulty. Note that, if the layer of the source and the destination node is different, first the packet will be routed to the destination layer. In fact the packet first unifies the z axis field, and then routes through the layer toward the destination In this case, the routing table does not require z-dimension field.

For example according to Fig. 2, source node S is sending packet to destination node P. At first, because the source and the destination are in the different layers, the packet will be sent to node A from the above link. In node A the above link is faulty, so it will find the nearest node with the healthy vertical link as an intermediate node. Because the nearest node with operating TSV is node C, then the packet will be routed to it. Now the above link is operating properly. Thus, it goes to D and then using layer routing table, will be routed to the destination node P.

ALGORITHM I. ERFAN-Dy Routing Algorithm

Input: Source, destination and current node
Output: Channel number as output port

```
1.  if TV==1 then
2.    if temp_adr = cur  then
3.      horizon_des = des;
4.      TV = 0;
5.    else
6.      horizon_des = temp_adr;
7.    end if
8.  else
9.    horizon_des = des;
10. end if
11. for each node channels
12.   if z_dest! != z_source then
13.     des_ver_c = get_prefer_ver_dir(des_adr, cur)
14.     if des_ver_c is faulty then
15.       temp_adr = get_inter_adr (cur, vertical link status)
16.       TV = 1
17.     end if
18.   else if x_source!= x_dest or y_source != y_dest then
19.     des_horizon_c = table_lookup(horizon_des, cur)
20.     if des_horizon_c is faulty then
21.       des_horizon_c = table_lookup(horizon_des, cur)
22.     end if
23.   end if
24.   else
25.     des_horizon_c = local port
26.   end if
27. end
```

Algorithm I presents the ERFAN-Dy routing algorithm. When the packet reaches to each switch, at first the validity of temporary address (i.e. TV bit) is checked. If the packet reaches to an intermediate switch, then a suitable vertical up/down direction is set to the final destination and TV bit is reset to '0'. If the packet doesn't reach to an intermediate node, the temporary address will be set and the current switch will look up the layer routing table to find suitable direction(s) to the intermediate switch. When the temporary address is not valid (i.e. the source and destination layers are different), first the switch will find all possible directions according to the vertical related fault table. If the vertical direction is faulty, an intermediate switch with a non-faulty vertical link according to routing and TSV fault tables is selected and its address is inserted into the temporary address field (i.e. T-Add) of the packet. Also the TV bit is set. In fact T-Add is the address of the nearest node to the current node with non-faulty up/down TSV. When the source and the destination layers are the same, the switch finds the suitable routes to destination according to the layer routing table. But if this horizontal path is faulty, another port with the smallest hop-count and the smallest traffic load (EFRAN-Dy) or just with the smallest hop-count (ERFAN-Sp) will be the output port. Indeed, in ERFAN-Sp, unlike EFRAN-Dy, the traffic load is not taking into consideration.

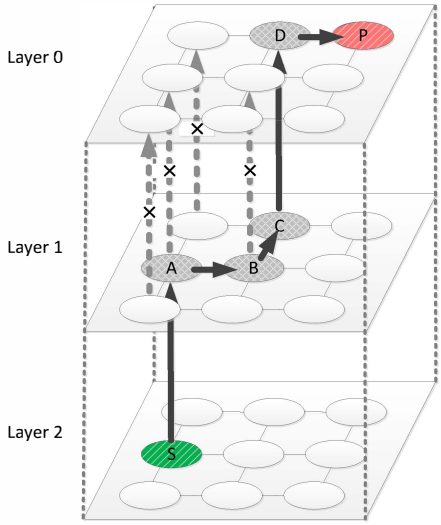

Fig. 2. Example of routing algorithm in a 3×3×3 3-D NoC

The proposed algorithms are live-lock free since vertical and horizontal faulty links do not separate the network. This is because at least one path between two nodes exists. Also the packet priority (i.e. HC field) helps to live-lock avoidance. A flit that passes lower hops has higher priority (i.e. less HC) for routing. The proposed deflection routing is also deadlock free because of using the shortest path in routing. The credit-based flow control is used to catch the traffic load information of a switch. Thus every router will understand the local traffic load (i.e. the number of occupied cells) of the entering buffers in adjacent routers. In general, flits never have to wait for a channel's buffer in the proposed algorithms. In the case of being blocked by one channel, ERFAN-Sp selects another channel with the lowest distance to the destination and ERFAN-Dy selects another channel with the lowest distance as well as lowest traffic load in the path.

V. EXPERIMENTAL RESULTS

The Booksim simulator [15] is used for evaluating our proposed routing algorithms. ORION 2.0 [16] is embedded in Booksim in order to evaluate power consumption. Two 3-D mesh topology architectures (i.e. 7×7×7 and 4×4×4) for both synthetic and real application workloads are used. Six trace-driven traffic patterns (i.e. FFT, LU, Ocean, Radix, Barnes, and Water-Spatial) are taken from Splash-2 [17]. The trace-driven traffic patterns are repeated to cover all the nodes of the network. The fault rates of 10%, 15%, and 20% are randomly injected to vertical, horizontal, or both links. For comparison, a low overhead fault-aware deflection routing algorithm (LOFAD) [10] is implemented and simulated.

A. Performance Evaluation

Table II to IV show the simulation results for ERFAN-Dy (with traffic control), ERFAN-Sp (without traffic control) and LOFAD for a 7×7×7 3-D NoC under real application workload. As we can see from these tables, the latency in ERFAN-Sp algorithm is less than the latency in the LOFAD and ERFAN-Dy in all the cases. For ERFAN-Dy with no-fault, the latency is almost equal to LOFAD but in the presence of TSV faults, it is less than LOFAD.

TABLE II. ERFAN-Dy - Real traffic workloads

7×7×7 3D-NoC	No Faults	10% Mix Faults	10% TSV Faults	10% Layer Faults
Latency (Injection rate=0.1)	56.99	61.64	59.60	64.21
Received Packets	760	737	741	726
Power	27.30	27.31	27. 29	27.31

TABLE III. ERFAN-Sp - Real traffic workloads

7×7×7 3D-NoC	No Faults	10% Mix Faults	10% TSV Faults	10% Layer Faults
Latency (Injection rate=0.1)	50.43	53.97	53.37	53.34
Received Packets	760	742	751	733
Power	27.13	27.19	27.16	27.16

TABLE IV. LOFAD - Real traffic workloads

7×7×7 3D-NoC	No Faults	10% Mix Faults	10% TSV Faults	10% Layer Faults
Latency (Injection rate=0.1)	56.98	66.70	67.96	63.01
Receive Packets	760	722	715	718
Power	27.30	27. 32	27. 31	27.31

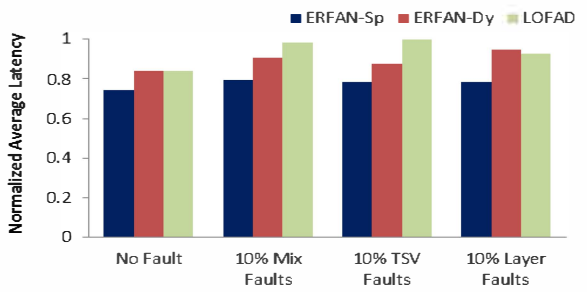

Fig. 3. Normalized average latency under real traffic workloads with 10% link faults

Also, as shown in Fig. 3, under real traffic workload with 10% fault links, the average improvement for ERFAN-Sp and ERFAN-Dy in comparison with LOFAD is 16.5% and 4% respectively. The improvement especially in the presence of TSV faults is because in presented routing algorithms, in case of fault in TSV link, the nearest intermediate node with a non-faulty TSV link according to TSV fault table will be chosen for packet routing. However, in LOFAD, first the packet is sent to the intermediate node and then the TSV link is checked. In this case if the TSV link is faulty, it has to re-route to another intermediate node. As an another observation from Fig. 3, it can be seen that load balancing (as it is used in ERFAN-Dy) is time consuming, rather than not applying the traffic load (as it is used in ERFAN-Sp).

In addition, Fig. 4 shows the normalized average latency of different algorithms with 10%, 15%, and 20% faults. As it can be seen, in both mixed and TSV faults both ERFAN-Sp and ERFAN-Dy outperforms LOFAD. Also, with increasing the link faults, the slope of latency increase for both ERFAN-Sp and ERFAN-Dy is less than LOFAD.

Furthermore, Fig. 5 shows the latency comparison between ERFAN-Sp and LOFAD for a 4×4×4 3-D NoC under uniform traffic workload with different injection rates. The average latency improvement for all injection rates in ERFAN-Sp is 15.5% and almost 20% in the presence of TSV faults. Fig. 6 shows the same comparison between ERFAN-Dy and LOFAD. Although LOFAD outperforms ERFAN-Dy in the presence of layer fault, the average latency improvement for all injection rates in ERFAN-Dy is 4.5%. As another observation from Fig. 5 and Fig. 6, it can be seen that both ERFAN-Sp and ERFAN-Dy work better than LOFAD in the presence of TSV faults.

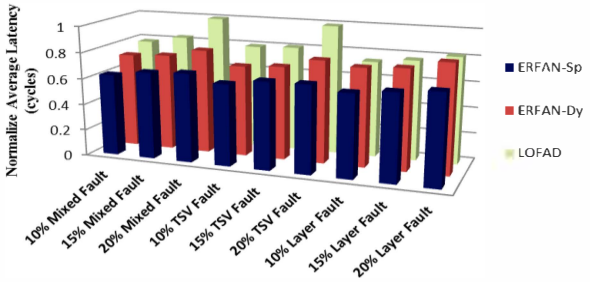

Fig. 4. Normalized average latency under real traffic workloads with different link faults

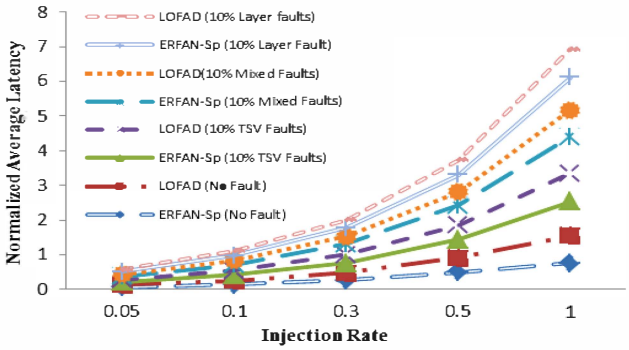

Fig. 5. Normalized average latency under synthetic traffic workloads (ERFAN-Sp vs. LOFAD)

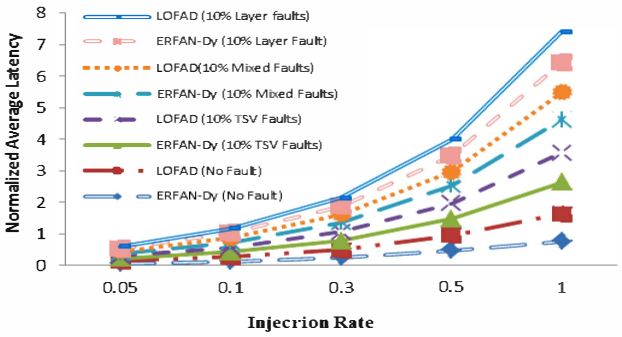

Fig. 6. Normalized average latency under synthetic traffic workloads (ERFAN-Dy vs. LOFAD)

B. Power Consumption Comparison

Power consumption is another parameter to be compared between different algorithms. As it can be seen from Fig. 7, all of the algorithms almost consume equal power. However, because of hop-count reduction of ERFAN-Sp, a slight improvement in power consumption is achieved.

C. Reliability Analysis

As it is shown in TABLE II to IV, the number of delivered packets in every network node in all faulty states in ERFAN-Sp and ERFAN-Dy algorithms is more than LOFAD. This shows the reliability of the proposed routing algorithms. In fact in average the reliability of these two algorithms are 97.6% and 96.6% respectively Fig. 8 depicts the reliability analysis of the routing algorithms. Both the proposed algorithm performs better in the presence of TSV faults.

D. Hardware Overhead

In order to capture hardware overhead, a credit-based wormhole switching, with XY routing were designed and implemented as our baseline. This router was created in the VHDL language and synthesized for the Altera Cyclone IV family of FPGAs, which use a 60 nm CMOS process. This router has five ports and each port has an input buffer with capacity of 4 flits per physical channel. Each flit has a number of bits equal to the physical data path width (i.e. 16 bits).

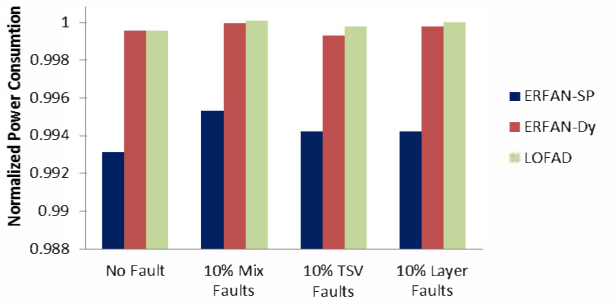

Fig. 7. Normalize power consumption under real traffic workloads

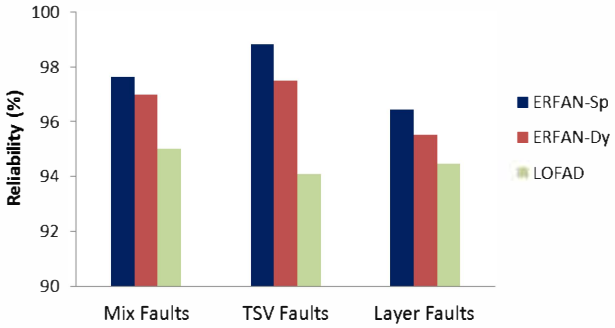

Fig. 8. Reliability Analysis

The values reported in Table V are the percentages of the area overhead that the different router designs impose to the system based on the baseline (i.e. a credit-based wormhole switching) The area overhead of ERFAN is due to its ability to efficiently tolerate faults.

TABLE V. Area overhead (LOFAD vs. ERFAN) percentages according to baseline

	Routing Table	Horizontal Fault Table	TSV Fault Table	Area Overhead
LOFAD	Yes/ Static	Yes/ dynamic	Yes/ dynamic	6.03%
ERFAN	Yes/ Static	Yes/dynamic	Yes/ dynamic	12.5%

VI. Conclusion and Future Works

In this paper two fault-tolerant routing algorithms (i.e. ERFAN-Sp and ERFAN-Dy) for 3-D NoCs are presented. Instead of one global table for the network, each router is equipped with a table contains the hop-count level of its layer and two separate layer fault tables for faulty horizontal links and faulty TSV links. In comparison with LOFAD [10], ERFAN-Sp and ERFAN-Dy have significant improvement in performance parameters while achieving high-reliability in exchange for small area overhead.

By focusing the main purpose of chip design on a power-driven one, for future work, proposing low-power routing algoirthms for both 2-D and 3-D NoCs targetting dark silicon architectures (e.g. Shift Sprinting [18]) seems to be beneficial.

References

[1] R. Akbar, A. A. Etedalpour, and F. Safaei, "An efficient fault-tolerant routing algorithm in NoCs to tolerate permanent faults," In *Journal of Supercomputing*,, Vol 72, pp 1-22, 2016.

[2] A. Rezaei, M. Daneshtalab, F. Safaei, and D. Zhao, "Hierarchical approach for hybrid wireless network-on-chip in many-core era," In *International Journal of Computers and Electrical Engineering*, Vol. 51, Issue C, pp. 225–234, 2016.

[3] V. F. Pavlidis and E. G. Friedman, "3-D topologies for networks-on-chip," In *IEEE Transactions on Very Large Scale (VLSI) Integration*, Vol. 15, Issue 10, pp. 1081-1090, 2007.

[4] C. Seiculescu, S. Murali, L. Benini, and G. De Micheli, "SunFloor 3D: A tool for networks on chip topology synthesis for 3D systems on chips", In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 9-14, 2009.

[5] Y. Qian, Z. Lu, and W. Dou, "From 2D to 3D NoCs: A case study on worst-case communication performance," In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 2-5, 2009.

[6] R. S. Ramanujam and B. Lin, "Randomized partially-minimal routing on three-dimensional mesh networks," In *IEEE Computer Architecture Letters*, Vol. 7, Issue 2, pp. 37-40, 2008.

[7] J. Duato, S. Yalamanchili, and L. Ni, *Interconnection Networks: An Engineering Approach*, Morgan Kaufmann Publishers Inc., 2000.

[8] C. Feng, Z. Lu, A. Jantsch, J. Li, and M. Zhang, "A reconfigurable fault-tolerant deflection routing algorithm based on reinforcement learning for network on-chip," In *International Workshop on Network on Chip Architectures (NoCArc)*, pp. 11-16, 2010.

[9] S. Paricha and Y. Zou, "A low overhead fault tolerant routing scheme for 3D networks-on-chip," In *International Symposium on Quality Electronic Design (ISQED)*, pp. 1-8, 2011.

[10] C. Feng, M. Jang, Z. Lu, and A. Jantch, "A low-overhead fault-aware deflection routing algorithm for 3D network-on-chip," In *IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pp. 19-24 2011.

[11] B. V. Dao, J. Duato, S. Yalamanchili and Y. Suh, "Software-based rerouting for fault-tolerant pipelined communication," In *IEEE Transactions on Parallel and Distributed Systems*, Vol. 11, Issue 3, pp. 193-211, 2000.

[12] T. Schonwald, J. Zimmermann, O. Bringmann, and W. Rosenstiel, "Fully adaptive fault tolerant routing algorithm for network-on-chip architectures," In *Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD)*, pp. 527-534, 2007.

[13] J. Wu, "A fault-tolerant and deadlock-free routing protocol in 2D meshes based on odd-even turn model," In *IEEE Transactions on Computers*, Vol. 52, Issue 9, pp. 1154-1169, 2003.

[14] A. Prodromou, A. Panteli, C. Nicopoulos, and Y. Sazeides, "NoC alert: An on-line and real-time fault detection mechanism for network-on-chip architectures", In *Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pp. 60-71, 2012.

[15] N. Jiang; D.U. Becker, G. Michelogiannakis, J. Balfour, B. Towles, D. E. Shaw, J. Kim, J and W. J. Dally, "A detailed and flexible cycle-accurate network-on-chip simulator," In *IEEE International Symposium on Performance Analysis of Systems and Software (ISPA*SS), pp.86-96, 2013.

[16] A. Kahng, B. Li, L. Peh, and K. Samadi, "ORION 2.0: a fast and accurate NoC power and area model for early-stage design space exploration," In *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 423-428, 2009.

[17] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The SPLASH-2 programs: characterization and methodological considerations," In *International Symposium on Computer Architecture (ISCA)*, pp. 24-36, 1995.

[18] A. Rezaei, D. Zhao, M. Daneshtalab, and H. Wu, "Shift sprinting: fine-grained temperature-aware NoC-based MCSoC architecture in dark silicon age," In *Proceedings of Design Automation Conference (DAC)*, Article 155, 2016.