Fault Tolerant and Highly Adaptive Routing for 2D NoCs

Manoj Kumar[1], Vijay Laxmi[1], Manoj Singh Gaur[1], Masoud Daneshtalab[2], Masoumeh Ebrahimi[2] and Mark Zwolinski[3]
[1]Malaviya National Institute of Technology, Jaipur, India
bohra.manoj1980@gmail.com, vlaxmi@mnit.ac.in, gaurms@mnit.ac.in
[2]University of Turku, Turku, Finland
masdan@utu.fi,masebr@utu.fi
[3]University of Southampton, Southampton, United Kingdom
mz@ecs.soton.ac.uk

***Abstract*—Networks-on-Chip (NoCs) are emerging as a promising communication paradigm to overcome bottleneck of traditional bus-based interconnects for current micro-architectures (MCSoC and CMP). One of the known current problems in NoC routing is the use of acyclic Channel Dependency Graph (CDG) for deadlock freedom. This requirement forces certain routing turns to be prohibited, thus, reducing the degree of adaptiveness. In this paper, we propose a novel non-minimal turn model which allows cycles in CDG provided that Extended Channel Dependency Graph (ECDG) remains acyclic. The proposed turn model reduces number of restrictions on routing turns, hence able to provide path diversity through additional minimal and non-minimal routes between source and destination. We also develop a fault tolerant and congestion-aware routing algorithm based on the proposed turn model to demonstrate the effectiveness. In this algorithm, a non-minimal route is used only when links in minimal routes are congested or faulty. Average performance gain of the proposed method is up to 26% across all selected benchmarks when compared with DRFT and 12% when compared with LEAR for 7×7 mesh.**

Keywords*—Networks-on-Chip, fault tolerance, congestion, deadlock freedom, routing, non-minimal paths, degree of adaptiveness.

I. Introduction

Networks-on-Chip (NoCs) are aggressively discussed and researched in past few years as a prominent and promising communication infrastructure for Chip Multi Processors (CMPs) and Multi Core Systems on Chips (MCSoCs) architectures because of their energy-efficiency, parallelism, increased predictability, scalability, reliability and reusability [1]. In a CMP or MCSoC micro-architecture, reliability of the on-chip network is greatly affected by permanent and/or transient faults. Faults can degrade the performance of a network and affect the functionality as well. Thus, it has become essential to design fault tolerant NoCs.

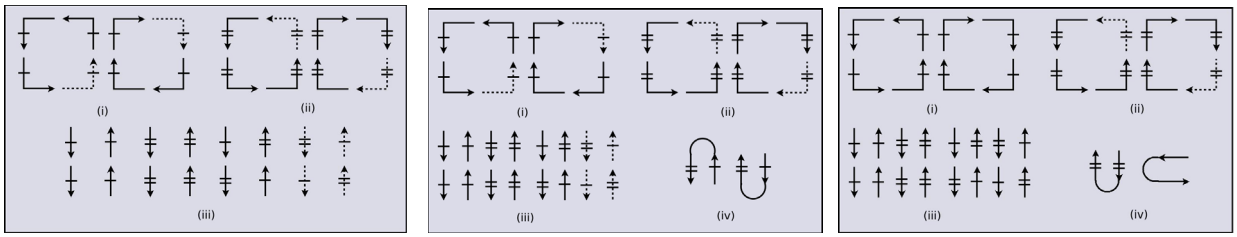
On the basis of inclusion of current network information in routing decision, routing algorithms are classified as either deterministic or adaptive. Deterministic routing algorithms always choose the same path between a given pair of source and destination nodes without considering current network status, even if there exists multiple possible paths. On the other hand, adaptive routing algorithms use current state (traffic and/or link status) of network in making routing decision to avoid congested or faulty links/regions of the network. It offers better throughput and latency than deterministic routing, especially under bursty and hot-spot traffic patterns.

According to their minimality, adaptive routing algorithms can be classified as minimal or non-minimal (mis-routing). Minimal adaptive routing algorithm only provides shortest routes from source node to destination node. At each node, it generates a productive output channel vector that suggests which output channels of the current node will make the current packet more closer to its destination. Non-minimal adaptive routing algorithms no longer restrict packets to move along a minimal path to the destination. Packets may be misrouted over output channels, which move them temporary away from the destination to avoid a faulty or congested channel. Minimal routing methods guarantee the shortest route between source and destination nodes. But, it is imprudent to neglect the promising performance improvements provided by non-minimal routing methods. For example, if all output channels corresponding to minimal routing paths are faulty; routing the packets along a non-minimal route may be the only alternative. Higher the degree of adaptiveness (path diversity) of the routing algorithm, lower is the probability for a packet to enter faulty or congested area. Thus, the focus of this research is to enhance the performance of routing algorithms by increasing degree of adaptiveness (path diversity) to achieve fault tolerance and congestion mitigation.

The remainder of this paper is organized as follow. Section II reviews the related work. Section III illustrates the proposed turn model, its deadlock freedom and fault tolerant routing algorithm. Section IV evaluates the effectiveness of the proposed method by comparing it with other routing methods. Finally, Section V concludes this paper.

II. Related Work

The overall network performance depends on many properties such as topology, switching mechanism, flow-control technique and routing algorithm. In this study, we mainly consider routing algorithms. The degree of adaptiveness and non-minimality of routing algorithms are used to balance traffic load and tolerate faults while maintaining deadlock freedom. Most of the routing schemes use turn models to



(a) Mad-y (b) LEAR (c) FTCAR

Fig. 1: Turn models (solid lines for permitted turns and dash lines for prohibited turns)

achieve deadlock freedom. Turn models [2]–[6] produce partially adaptive routing schemes resulting in lower degree of adaptiveness.

Virtual channels can be used to increase adaptivity and avoid deadlocks. The routing schemes introduced in [7], [8] produce an equivalent fully adaptive and minimal routing algorithm called double-y for 2D mesh. It requires one and two virtual channels in X and Y dimensions, respectively. In [9], authors introduced maximally adaptive double-y (Mad-y) routing algorithm which is an improvement over double-y network based schemes [7], [8]. It allows more routing turns by making better utilization of virtual channels to increase adaptivity. However, it does not tolerate link or node faults.

Generally, non-minimal routing is used to tolerate faults and alleviate congestion. In [10], authors proposed non-minimal fault tolerant routing algorithm to handle multiple nodes and links failures without using routing tables. Authors in [11], proposed distributed and reconfigurable fault tolerant (DRFT) routing scheme to tolerate single link faults. In the absence of faults, proposed scheme works as dimension order routing. Fick *et al*. [12] introduced a table-based implementation of routing scheme to tolerate faulty components. In [13], authors proposed a fault tolerant routing scheme. It can handle one faulty link or node. But, it provides lower degree of adaptiveness. In [14], [15], Ebrahimi *et al*. proposed non-minimal routing schemes for 2D mesh. These provide better adaptiveness than Mad-y [9] with the same number of virtual channels. However, these impose some unnecessary restrictions on routing turns, which can be removed to achieve fault tolerance and increased path diversity.

In this paper, we propose a novel turn model (FTCAR), which is a *major* improvement over existing 2D turn models used by several routing algorithms [13]–[18] including Mad-y [9]. The Mad-y algorithm is the base of all mentioned algorithms [13]–[19]. We have also extended the proposed model to tolerate faults in the network. It can tolerate all single-link failures. It can also handle multiple-links failures depending upon the distance of faults.

III. Proposed Work

A. Preliminaries

A packet moving towards direction A makes a routing turn A-B, if the packet turns towards direction B. For example, a turn E-$N1$ taken by a packet means that the packet is moving from east to north-1 virtual channel. In mesh networks, routing turns can be of three types: 0-degree (packet switches from one virtual channel to another virtual channel in same direction), 90-degree (packet switches from one virtual channel to another virtual channel in different dimension) and 180-degree (also known as U-turn in which packet switches from one virtual channel to a virtual channel in opposite direction of same dimension). For minimal routing, 180-degree turns cannot be used. The Mad-y, LEAR and the proposed turn model deploy double-y network that uses one and two virtual channels along X and Y dimensions, respectively.

For a given routing function R and a given topology, *channel dependency graph* (CDG) [20] is a directed graph whose nodes are network channels (C) and edges represent dependencies between these channels. For a given routing subfunction R_1 of the routing function R, an *extended channel dependency graph* (ECDG) [21] is a directed graph whose nodes are network channels $C_1 \subset C$ and edges represent dependencies between these channels.

Figures 1a and 1b illustrate the routing turns for Mad-y and LEAR routing algorithms, respectively. In order to achieve deadlock freedom, Mad-y imposes following constraints on routing turns:

1) It prohibits four 90-degree turns ($N2$-W, $S2$-W, E-$N1$ and E-$S1$) as shown in Fig. 1a(i) and 1a(ii).
2) It also prohibits two 0-degree turns ($S2$-$S1$ and $N2$-$N1$) as shown in Fig. 1a(iii).
3) Since it is a minimal routing algorithm, it does not permit any 180-degree turn.

The LEAR turn model has same turn constraints as in the Mad-y (Fig. 1b(i), 1b(ii), and 1b(iii)) except 180-degree turns. The LEAR model allows some 180-degree turns as shown in Fig. 1b(iv) while the Mad-y model prohibits all 180-degree turns. The deadlock freedom of both Mad-y and LEAR routing algorithms is proved using Dally and Seitz work [20] .

B. FTCAR: Turn Model

In a routing algorithm, an acyclic CDG requirement to avoid deadlocks imposes unnecessary restrictions on the routing turns. The deadlock freedom of both Mad-y and LEAR routing algorithms is proved using acyclic CDG [20]. Thus, they cannot use all qualified turns to route packets through less congested/faulty areas. The proposed model FTCAR imposes substantially fewer restrictions on routing turns (specially on 90-degree) using [21], thus it provides additional minimal and

non-minimal paths between source and destination nodes than Mad-y and LEAR.

Figure 1c shows turn model representation of FTCAR. A packet is permitted to use first virtual channel ($N1$ or $S1$) at any time as shown in Fig. 1c(i). It can use second virtual channel ($N2$ or $S2$) only if it has already routed to negative direction of X dimension (west). Because, the packet cannot take west turn after using ($N2$ or $S2$) as shown in Fig. 1c(ii). It is allowed to take only two 180-degree turns from west to east (W-E) and south to north ($S2$-$N2$) as shown in Fig. 1c(iv) only if it has completed routing in west and south directions, respectively. In short, in order to avoid deadlocks, FTCAR imposes following constraints on routing turns:

1) It prohibits two 90-degree turns ($S2$-W and $N2$-W).
2) It allows 0-degree turns ($S1$-$S1$, $N1$-$N1$, $S2$-$S2$ and $N2$-$N2$) as shown in Fig. 1c(iii). It allows 0-degree turns ($S1$-$S2$, $N1$-$N2$, $S2$-$S1$ and $N2$-$N1$), as shown in Fig. 1c(iii), with some restrictions. It allows these restricted turns only when packet does not need to be forwarded further west.
3) It permits some 180-degree turns (W-E and $S2$-$N2$).

The proposed method FTCAR provides more minimal and non-minimal paths between source and destination by allowing some routing turns which were prohibited in Mad-y and LEAR. For example, E-$N1$ (i.e. a packet moving from east to north-1 virtual channel) and E-$S1$ (i.e. a packet moving from east to south-1 virtual channel) are permitted by FTCAR turn model. However, these permitted routing turns create cycles in CDG, but we have shown that proposed routing method is deadlock free using Duato's well known theorem [21]. Duato has proved that cycles can be allowed in CDG provided that ECDG is acyclic.

C. Deadlock and Livelock Freedom of FTCAR

With non-adaptive routing, packets are routed along single output channel at each node. Thus, in order to achieve deadlock freedom, it is essential to eliminate all cyclic dependencies between network channels. In adaptive routing, packets often have many choices for routing at each node. Thus, it is not mandatory to remove all cyclic dependencies between channels, provided that every packet can be routed on a path whose channels are free from cyclic dependencies. The channels involved in these acyclic routes are considered as *escape channels* from deadlocks (cycles).

The FTCAR can be proved deadlock-free by using Duato's well known theory [21] stated as follows:

Theorem 1: (Duato's Theorem) For a given network I, a connected and adaptive routing function R is deadlock-free if there exists a routing sub-function $R_1 \subseteq R$, which is connected and has extended channel dependency graph acyclic.

Following Duato's terminology, the routing function of FTCAR is denoted by R and the channel set used by routing function R is denoted by C. To prove the FTCAR deadlock-free , we first identify the subset of channels $C_1 \subseteq C$, that defines routing subfunction $R_1 \subseteq R$ that is connected and has an acyclic ECDG. The ECDG must not contain any cycle arising from direct, direct-cross, indirect and indirect-cross dependencies. For FTCAR, the channel set C_1 contains all channels of the network except north-1 ($N1$) and south-1 ($S1$).

Lemma 1: The routing subfunction R_1 is connected.

Proof: The routing subfunction R_1 with the channel set C_1 is same as west-first [2] routing (non-minimal) function. Since non-minimal west-first routing is connected, so we conclude that R_1 is connected. ■

Lemma 2: The extended channel dependency graph of channel set C_1 with additional dependencies introduced by channel ($N1$ and $S1$) of R is acyclic.

Proof: There is no direct-cross dependency between channels of C_1 as a channel belonging to C_1 is always used as escape channel for all the destination for which it can be supplied by R. The additional channels ($N1$ and $S1$) of R can introduce only indirect dependencies between west channels. Because a packet using west channel can further use the west channel of different row and column. But because of this indirect dependency, there is no cycle in ECDG of C_1. The ECDG for C_1 has no dependencies to a west channel from a channel in north, east, or south, so the west channels are always used before all other channels in C_1, if destination is in west. Hence, these indirect dependencies introduce new dependencies between only the west virtual channels and create no cycles using only the west virtual channels. Since there are no indirect and direct dependencies, which produce cycle in ECDG. Therefore ECDG of C_1 is acyclic. ■

Theorem 2: The proposed routing algorithm is deadlock-free.

Proof: Using Lemma 1, Lemma 2 and Theorem 1 , we conclude that FTCAR routing algorithm is deadlock-free. ■

Non-minimal routing algorithms are susceptible to livelock. The FTCAR can be proved to be livelock free using following theorem.

Theorem 3: The proposed routing algorithm is livelock-free.

Proof: We can observe from FTCAR turn model (Fig. 1c) that whenever a packet is forwarded along east output channels, it is not allowed to route the packet back along west output channel. So, in the worst case, the packet may reach to the west border, then it starts to move towards destination column. Similarly, whenever a packet is forwarded along north output channel, it is not allowed to route back in south. It can be noticed that only one 180-degree turn is allowed in each dimension. Therefore, after a certain number of hops, the packet finally arrives at the destination node. Thus, proposed routing algorithm (FTCAR) is livelock free. ■

D. FTCAR: Routing Algorithm

Since FTCAR is a fully adaptive routing method, it can use all minimal paths for each source-destination pair, if the fault is absent. A destination node (**D**) may be located in any of eight directions/quadrants (north, south, east, west, northeast, northwest, southeast, and southwest) with respect to current/source (**C**) node. When the destination node is located in any of quadrant/direction, available minimal paths (S_{min}) between current node (0,0) and destination node (X_1, Y_1) is given by:

$$S_{min} = \frac{(|X_1| + |Y_1|)!}{|X_1|!|Y_1|!}$$

It can be notice that S_{min} is always greater or equal to 2 ($\geq$ 2) for any quadrant (northeast, northwest, southeast, and southwest). Thereby, when the destination is in any of quadrants and there is a single faulty link, packets can always find minimal path to bypass the fault.

If destination node is located in any of direction (east, west, north, and south), S_{min} is always 1. Thus, packets must follow non-minimal paths if any link on the minimal path is faulty. Figure 2 illustrates how FTCAR tolerates single link failures in any of direction. When destination node is located in east (eastward packet), at first the east link is checked. If it is available, the packet is routed through this link. However, if the link is faulty as shown in Fig 2a, the packet can be forwarded through $N2$ or $S2$ link. By inspecting the required turns, it can be seen that eastward packets use either the $N2$-E and E-$S2$ turns or the $S2$-E and E-$N2$ turns to bypass the faulty link. It can be easily verified that all these turns are permitted turns in FTCAR turn model (Fig. 1c). Similarly, we can observe that westward packets can use either the $N1$-W and W-$S1$ turns or the $S1$-W and W-$N1$ turns to bypass the west faulty link as shown in Fig 2b. Similarly, northward and southward packets use the turn W-$N2$, $N2$-E, W-$S2$ and $S2$-E to bypass the north and south faulty links as shown in Fig. 2c and 2d, respectively. The turns used by westward, northward and southward packets are allowed turns in FTCAR turn model (Fig. 1c).

Figures 2e and 2f illustrate two special cases of northward and southward packets on the west border of the mesh. When both current and destination nodes are located on west border, the required turns include $N2$-W and $S2$-W to bypass north and south faulty links, respectively. These turns are prohibited in the FTCAR turn model (Fig. 1c). However, if we allow these prohibited turn on the west border, they cannot result into complete cycle. Thus, these prohibited turns can be allowed on the west border.

Functionality of FTCAR routing algorithm is divided into two phases: route computation and output channel selection. On the basis input channel (on which packet has arrived) and relative position of destination node with respect to current node, routing function computes a set of output channels using turn model explanation discussed in Section III-B.

The selection function selects one output channel from the set of output channels provided by the route computation function. Our selection function prefers adaptive output channels over escape channels, because it results in increased probability of escape output channels being available when they are required to avoid deadlocks. The selection function first checks all qualified output channels corresponding to shortest routes and forwards the packet to the output channel in which the corresponding next hop node has its congestion status flag set to zero. If the congestion status flags of all next hop nodes on shortest routes are set to one, the congestion status flag of each eligible non-minimal route is inspected. If there exist such non-minimal routes, which are not congested, our method selects one of the output channel to forward the packet (and possibly, corresponding to *non-escape channels*).

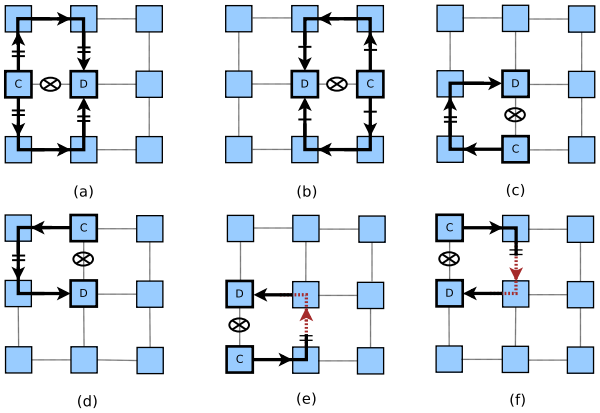


Fig. 2: Tolerating single link failures for packet directions (a) *eastward* (b) *westward* (c) *northward* (d) *southward* (e) *northward on west border* (f) *southward on west border* (red dash lines indicate prohibited turns)

IV. Experimental Setup and Result Analysis

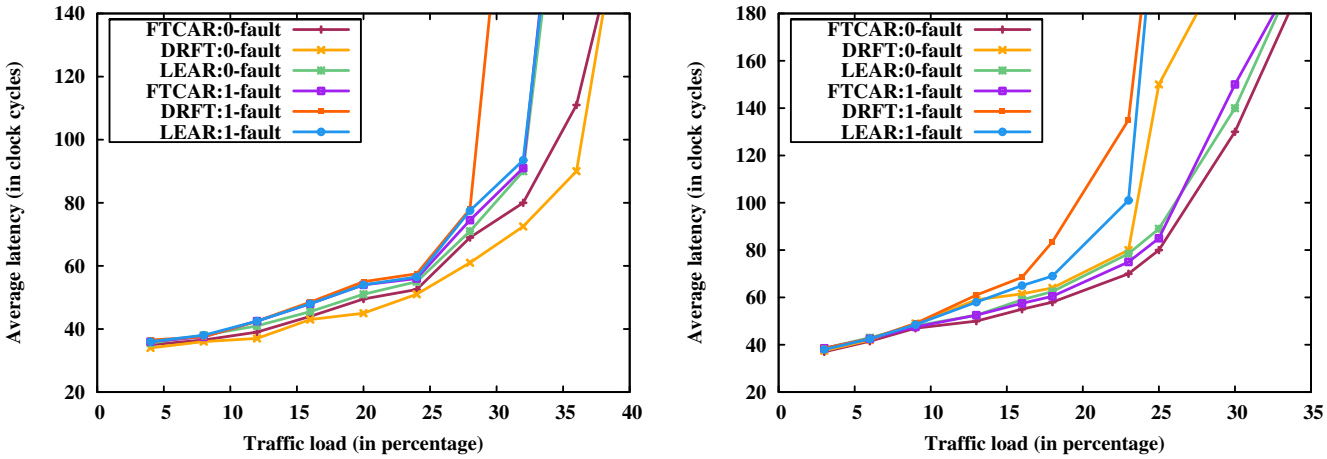
A. Experiment Setup

In order to perform simulation involving various routing schemes, traffic scenarios and power analysis, we modified and extended [22], a cycle-accurate SystemC based open source NoC simulator. We have evaluated proposed routing method (FTCAR) with real and synthetic traffic profiles. To evaluate effectiveness of FTCAR, we have implemented 0 and 1 fault versions of other well-known routing methods DRFT [11], LEAR:0-fault [14] and LEAR:1-fault [13]. As synthetic traffic, we use uniform and hotspot traffic patterns. For realistic applications traffic, we consider traces of selected application suites extracted from benchmark suite E3S [23].

For all experiments of simulation, we consider 7×7 mesh topology. The traffic sources are configured to generate packets of size 8 flits. The input-channel buffer size for each virtual channel is set to 6 flits. The warm-up period for the simulator is set to $15,000$ clock cycles and afterwards; the average performance is measured over another $85,000$ cycles. The traffic is generated over first $75,000$ cycles. Congestion threshold is set to 66% of total buffer size of neighboring node. As communication performance parameter, we consider latency which is defined as the time difference (in clock cycles) between header flit injection from source router and tail flit reception at destination router.

B. Performance Evaluation

1) Uniform Traffic Profile: Under this traffic pattern, each node sends several packets to every other node in the network with same probability. The load-latency graph for uniform traffic model is presented in Fig. 3a. As shown, all algorithms exhibit similar average latency at lower traffic loads. But with increased packet injection rate, it is observed that DRFT performs better than all other routing methods as expected when there is no fault in the network. Because DRFT with no fault is similar to dimension order routing (XY) which incorporates relatively long term and more global information about uniform traffic load characteristic [2]. Since DRFT routes



(a) Under uniform traffic

(b) Under hot-spot traffic

Fig. 3: Average latency in 7×7 mesh at different loads

packets first along X dimension and then in Y dimension, it distributes packets as evenly as possible throughout the network in the long-term. In one-faulty case, it is observed that FTCAR performs better than other routing methods at higher traffic loads. Because FTCAR is fully adaptive algorithm and provides high path diversity than other routing methods when network is slightly congested.

2) Hot-spot Traffic Profile: Under this traffic pattern, some nodes are appointed as hot-spot nodes, which receive some additional hot-spot traffic besides their normal uniform traffic. For simulation, we set nodes (3,4) and (4,3) as hot-spot nodes with 0.3 probability of getting additional traffic. Figure 3b shows average latency under this model for 7×7 mesh. It can be observed clearly that DRFT in contrast with the uniform traffic profile, has a higher average latency than other algorithms when there is no fault in the network. Because, when multiple traffic flows are oriented towards a small subset of "hot spot" nodes, non-adaptive DRFT router will be compelled to forward them towards the same output direction, thus saturating the virtual channel queues. On the other hand, adaptive algorithms can direct packets, destined for same destination, to different output channels. It can also be observed that due to fully adaptive nature and higher adaptiveness (both minimal and non-minimal), FTCAR scheme achieves better average latency than other algorithms by avoiding "hot spots".

FTCAR method leads to smaller average latencies in both faulty and non-faulty cases, because it can more evenly distribute traffic in a congested network using additional paths both minimal and non-minimal than other routing algorithms.

3) Application Traffic Profile: To evaluate the proposed method in a more realistic scenario, we select four application suites office-automation, networking, automotive/industrial and consumer from E3S benchmark suite [23]. This selection is intended to represent various applications used in the real-time embedded systems. Each task is allocated a core (processor) using *minimum execution time scheduler* that executes it in the fastest time. Task mapping strongly depends on particular application traffic. We use random mapping algorithm to compute the locations of cores within NoC to enable honest and intuitive comparisons among routing algorithms. We have executed routing algorithms repeatedly several times using random mapping and the average of simulation results are used. Figure 4 shows average packet latency normalized to DRFT:1-fault routing. FTCAR provides lower latency than other methods across all four application suites. FTCAR shows the greatest performance gain on consumer application traces. Average performance gain of FTCAR is up to 26% across all selected benchmarks vs DRFT and 12% vs LEAR for 7×7 mesh.

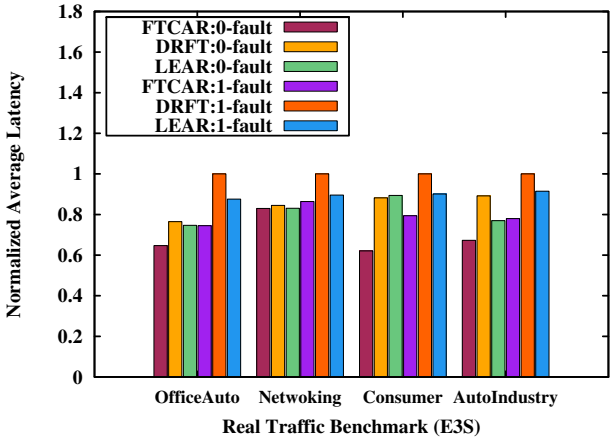


Fig. 4: Performance for application traffic

C. Power Analysis

We integrated an existing NoC power estimation tool ORION [24] with NoC simulator [22]. It estimates total power consumption of a router into various sub-components: input buffers, router control logic (arbiter and crossbar) traversal and channels. We have averaged *0-fault* and *1-fault* versions of FTCAR, DRFT and LEAR. Figure 5 illustrates average

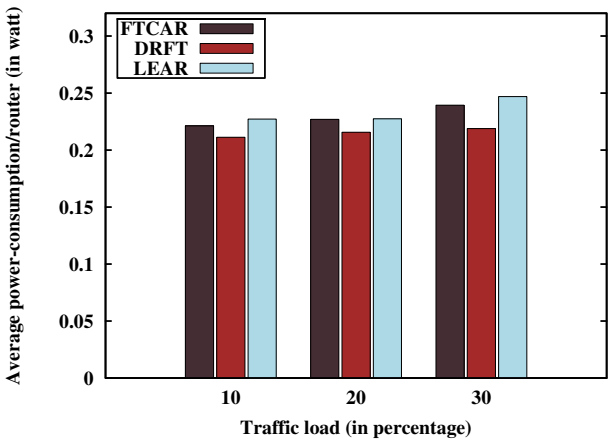


Fig. 5: Power consumption under hot-spot traffic

power consumption under hotspot traffic. It can be observed that DRFT method consumes less power for all traffic load. Because, it generally routes packets through minimal paths. At lower traffic loads, FTCAR performs better than LEAR as it uses minimal paths due to small "hot spots" creation at lower traffic load. However, performance of FTCAR is similar to LEAR at higher traffic loads.

V. Conclusion and Future Work

Acyclic channel dependency graph requirement for deadlock freedom results in low degree of adaptiveness and fault tolerance. In this paper, we presented a novel turn model for 2D meshes to provide high degree of adaptiveness and fault tolerance. We also proposed fault-tolerant and congestion-aware routing algorithm (FTCAR) to mitigate congestion. FTCAR provides higher degree of adaptiveness by allowing cycles in channel dependency graph. It can use minimal or non-minimal routes between source and destination nodes depending on congestion and fault status. Deadlock freedom of FTCAR is ensured using Duato's theory. It can tolerate all single-link failures. It can also handle multiple-links failures depending upon the distance of faults. Results show that FTCAR improves performance by routing packets through non-congested regions when network is congested. Further developments include node fault-tolerance and 3D extension of FTCAR.

Acknowledgment

This work is supported by UK-India Education and Research grant for the project on HiPER NIRGAM (2011-2014).

References

[1] M. Palesi and M. Daneshtalab, *Routing Algorithms in Networks-on-Chip*. Springer, 2014.

[2] C. Glass and L. Ni, "The Turn Model for Adaptive Routing," in *Proceedings of 19th International Symposium on Computer Architecture*, 1992, pp. 278–287.

[3] Z. Zhang, A. Greiner, and S. Taktak, "A Reconfigurable Routing Algorithm for a Fault-tolerant 2D-Mesh Network-on-Chip," in *Proceedings of 45th Design Automation Conference*, 2008, pp. 441–446.

[4] G.-M. Chiu, "The Odd-Even Turn Model for Adaptive Routing," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 11, no. 7, pp. 729–738, 2000.

[5] B. Fu, Y. Han, J. Ma, H. Li, and X. Li, "An Abacus Turn Model for Time/Space-Efficient Reconfigurable Routing," in *Proceedings of 38th International Symposium on Computer Architecture*, 2011, pp. 259–270.

[6] C. J. Glass and L. M. Ni, "Fault-tolerant wormhole Routing in meshes," in *Proceedings of 23rd International Symposium on Fault-Tolerant Computing*, 1993, pp. 240–249.

[7] A. Chien and J. Kim, "Planar-Adaptive Routing: Low-cost Adaptive Networks for Multiprocessors," in *Proceedings of 19th International Symposium on Computer Architecture*, 1992, pp. 268–277.

[8] D. H. Linder and J. C. Harden, "An Adaptive and Fault Tolerant Wormhole Routing Strategy for k-ary n-cubes," *Computers, IEEE Transactions on*, vol. 40, no. 1, pp. 2–12, 1991.

[9] C. J. Glass and L. M. Ni, "Maximally Fully Adaptive Routing in 2D Meshes," in *International Conference on Parallel Processing, volume I*, 1992, pp. 101–104.

[10] F. Chaix, D. Avresky, N.-E. Zergainoh, and M. Nicolaidis, "A Fault-tolerant Deadlock-free Adaptive Routing for On-chip Interconnects," in *Proceedings of 14th Design, Automation Test in Europe Conference Exhibition*, 2011, pp. 1–4.

[11] M. Kumar, Pankaj, V. Laxmi, M. S. Gaur, and S.-B. Ko, "Reconfigurable Distributed Fault Tolerant Routing Algorithm for On-Chip Networks," in *Proceedings of 26th International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems*, 2013, pp. 290–295.

[12] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw, "A Highly Resilient Routing Algorithm for Fault-tolerant Nocs," in *Proceedings of 12th Design, Automation and Test in Europe*, 2009, pp. 21–26.

[13] M. Ebrahimi and M. Daneshtalab, "A Light-weight Fault-tolerant Routing Algorithm Tolerating Faulty Links and Routers," *Computing*, pp. 1–18, 2013.

[14] 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 *Proceedings of 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing*, 2012, pp. 520–524.

[15] M. Ebrahimi, M. Daneshtalab, F. Farahnakian, J. Plosila, P. Liljeberg, M. Palesi, and H. Tenhunen, "HARAQ: Congestion-Aware Learning Model for Highly Adaptive Routing Algorithm in On-Chip Networks," in *Proceedings of 6th International Symposium on Networks on Chip*, 2012, pp. 19–26.

[16] P. Lotfi-Kamran, M. Daneshtalab, C. Lucas, and Z. Navabi, "BARP-A Dynamic Routing Protocol for Balanced Distribution of Traffic in NoCs," in *Proceedings of 11th Design, Automation and Test in Europe*, March 2008, pp. 1408–1413.

[17] M. Li, Q.-A. Zeng, and W.-B. Jone, "DyXY - A Proximity Congestion-Aware Deadlock-free Dynamic Routing Method for Network on Chip," in *Proceedings of 43rd Design Automation Conference*, 2006, pp. 849–852.

[18] P. Lotfi-Kamran, A. Rahmani, M. Daneshtalab, A. Afzali-Kusha, and Z. Navabi, "{EDXY} a Low cost Congestion-aware Routing Algorithm for Network-on-Chips," *Journal of Systems Architecture*, vol. 56, no. 7, pp. 256 – 264, 2010.

[19] M. Kumar, V. Laxmi, M. S. Gaur, S.-B. Ko, and M. Zwolinski, "CARM: Congestion Adaptive Routing Method for On Chip Networks," in *Proceedings of 27th International Conference on VLSI Design and 13th International Conference on Embedded Systems*, 2014, pp. 240–245.

[20] W. J. Dally and C. L. Seitz, "Deadlock-free Message Routing in Multiprocessor Interconnection Networks," *Computers, IEEE Transactions on*, vol. 100, no. 5, pp. 547–553, 1987.

[21] J. Duato, "A Necessary and Sufficient Condition for Deadlock-free Adaptive Routing in Wormhole Networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 6, no. 10, pp. 1055–1067, 1995.

[22] "Nirgam:," http://wiki.mnit.ac.in/mediawiki/index.php/Nirgam/.

[23] "E3S:," http://ziyang.eecs.umich.edu/ dickrp/e3s/.

[24] A. Kahng, B. Li, L.-S. Peh, and K. Samadi, "ORION 2.0: A Fast and Accurate NoC Power and Area Model for Early-stage Design Space Exploration," in *Proceedings of 12th Design, Automation and Test in Europe Conference Exhibition*, 2009, pp. 423–428.