Towards an Effective Utilization of Partially Defected Interconnections in 2D Mesh NoCs

Changlin Chen, and Sorin D. Cotofana
Computer Engineering, Software and Computer Technology
Delft University of Technology, Delft, the Netherlands
Email: {c.chen-2, S.D.Cotofana}@tudelft.nl

***Abstract*—In typical NoC systems, most Routing Algorithms (RAs) abandon the interconnection between two adjacent routers if one traffic direction is broken, despite whether the other one is still functional or not. In this paper, we propose a distributed logic based RA, which can efficiently utilize the UnPaired Functional (UPF) links in such partially defected interconnects. The basic fault pattern tolerated by the proposed RA is a fault wall, which is composed of adjacent broken links with the same outgoing direction. Messages are routed around the fault walls along the misrouting contours of the broken links. The proposed RA requires at least 3 Virtual Channels (VCs) and dynamically reserve them to misrouted messages to avoid deadlock. Our experiments indicate that, for random and localized traffic patterns, we achieve an average saturation throughput 20% higher than the Solid Fault Region Tolerant (SFRT) RA, and 22% and 14% higher than the Ariadne routing table based RA, respectively. For the real applications, *sample* and *satell*, our proposal requires a routing execution time with at least 16% shorter than both SFRT and Ariadne. Synthesis results with Synopsis Design Compiler and TSMC 65nm technology indicate that, embedding the proposed RA into a baseline router results in 11% area overhead, which is only 3% higher than that of SFRT. In contrast, Ariadne area overhead is 15% for an 8×8 NoC and increases to 21% for a 10×10 NoC.**

***Keywords*-Networks-on-Chip; Fault tolerant; Routing algorithm; Unidirectional links;**

I. Introduction

As an increasing number of cores are being integrated into one chip, Networks on Chip (NoC) become the de facto many core systems' on chip interconnection strategy for they have higher throughput, lower data transmission latency, and higher scalability than conventional interconnects, e.g., bus and crossbar. However, the transistor miniaturization also makes the NoC links and routers more prone to errors due to various kinds of physical defects which consequences must be properly handled to avoid severe system performance degradation.

When the NoC link fault level is low, i.e., only a small number of its wires are broken, it can still be utilized with Partially Faulty Link Usage Methods (PFLUM), e.g., [1]. Similarly, routers can also be partially defected and still be utilized [2]. However, when the fault level is high, the link has to be treated as broken and some router ports, or even the entire router must be isolated. Moreover, when physical ports or routers are broken, the links incident to them are abandoned. Thus without losing generality, router defects can be considered as equivalent with the situation when links incident to it are broken.

Conventionally, each interconnection between two adjacent NoC routers is composed of a pair of unidirectional links, each link having its own control flow wires and handling either outgoing or incoming traffic. If one unidirectional link is broken, one data transmission direction is lost. Tsai et al. [3] suggest to replace the unidirectional links with bidirectional ones such that when one link is broken, the other one can be utilized in both directions resulting in a half-duplex communication. However, unidirectional links are still attractive as they provide better means to implement the control logic and to address timing error issues [4]. Besides, when an input or output port is broken, a bidirectional link also becomes unidirectional. In view of these observations, in this paper, we focus on NoCs that utilize unidirectional links.

Assuming links have the same fault rate, and links with low fault levels are still utilized by means of PFLUM, the probability that both links in an interconnection are broken is typically low. However, most state of the art Fault Tolerant (FT) Routing Algorithms (RAs), e.g., [5]–[15], abandon the entire interconnection even when only one of the two links is broken. Thus the UnPaired Functional (UPF) links in such interconnections are wasted even though the UPF link utilization can partially preserve the link capabilities and can result in graceful system performance degradation.

In this paper, we propose a distributed logic based fault tolerant RA, which can utilize all the UPF links incident to active routers. The basic fault pattern tolerated by the proposed RA is a fault wall, which is composed of adjacent broken links with the same outgoing direction. Messages are routed around the fault walls along the misrouting contours of the broken links. The proposed RA requires a minimum number of 3 Virtual Channels (VCs) and dynamically reserves them to messages whose transmission is blocked to guaranty deadlock freeness. Our experiments indicate that, for random and localized traffic patterns, we achieve an average saturation throughput 20% higher than the Solid Fault Region Tolerant (SFRT) RA, and 22% and 14% higher than the Ariadne routing table based RA, respectively. For the real applications, *sample* and *satell*, our proposal requires a routing execution time with at least 16% lower than both SFRT and Ariadne. Synthesis results with TSMC 65nm technology indicate that, embedding the proposed RA into a baseline router results in 11% area overhead, which is only 3% higher than that of SFRT. In

978-1-4799-3765-3/14 $31.00 © 2014 IEEE
DOI 10.1109/ISVLSI.2014.70

contrast, Ariadne area overhead is 15% for an 8×8 NoC and increases to 21% for a 10×10 NoC.

The rest of the paper is organized as follows. Section II presents a brief related work survey. Section III introduces the fault pattern validation process. Section IV describes the proposed fault tolerant routing algorithm and proves its deadlock freeness property. Section V evaluates the performance of the proposed algorithm and compares it with closely related work. Section VI concludes the presentation.

II. Related Work

State of the art fault tolerant RAs can be roughly classified into three groups based on the number of tolerated faults and the way they are tolerated.

RAs in the first group, e.g., [5]–[7], can tolerate a bounded number of faults by modifying the baseline RA turn rules around the faults without causing deadlock. Usually the system performance degradation is minimal when faults occur as the modifications are turn based only and do not constraint the VC usage in each of the routers. However, their application scope is limited to NoC systems with low fault rates.

RAs in the second group, e.g., [8]–[12], can tolerate an unbounded number of faults, but the regions formed by the faults must satisfy certain requirements, e.g., the fault regions must have solid shapes like +, L, or T [8], [9]. Otherwise, some fault free routers have to be deactivated to make the shape solid. In [10], UPF links are still utilized when a basic condition is satisfied. This results in a better performance than [9], however, when the basic condition is not satisfied, [10] has to switch to a conventional Solid Fault Region Tolerant (SFRT) RA, e.g., [9]. To avoid deadlock, RAs in this group usually constrain the VC usage on the fault region boundaries, which results in a substantial system performance degradation. The main advantage of such kind of Solid Fault Region Tolerant (SFRT) RAs is that VCs are utilized according to the underlying RA in the fault free area. This provides the best system performance to applications if their tasks are best effort based mapped on the NoC fault free area.

RAs in the third group, e.g., [13]–[15], can tolerate an unbounded number of faults and do not pose any restrictions on the faulty region. The routing path between source and destination routers is determined by searching the best path during message transmission or NoC reconfiguration, when a new fault is detected. These routing paths are stored in routing tables to indicate how the following messages have to be transmitted. RAs in this group have the advantage that they can utilize almost all functional resources as long as they are connected with the NoC and usually require a low number of VCs. However, they also have drawbacks when used in wormhole switched NoCs. On one hand, a routing table is maintained in each router to indicate the output ports to requested destinations. The routing table sizes are proportional to the NoC size, which makes such approaches less scalable than distributed RAs and introduces a high silicon area overhead. On the other hand, when the VC usage is constrained, it is applied throughout the entire NoC, i.e., both in the fault free area and in the faulty region, which results in unjustified performance reduction as it is known [10] that increasing the VC usage flexibility can substantially improve the system performance.

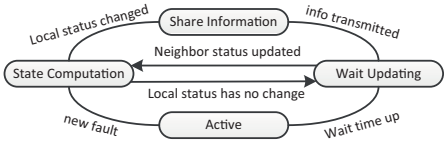


Fig. 1. Flow chart of fault pattern validation FSM in each router.

In view of the previous discussion, we propose a distributed logic based RA, which can tolerate an unbounded number of faults and can efficiently utilize UPF links. The proposed RA provides a better graceful system performance degradation when compared with the aforementioned RAs.

III. Fault Pattern Validation

When a new fault is detected, the fault pattern must be validated before messages can be forwarded. The fault pattern validation is controlled by a Finite State Machine (FSM), illustrated in Fig. 1, implemented in each NoC router. Similar with the fault pattern validation process proposed in [11], we assume that each active router has a self-test mechanism to periodically diagnose itself and all the links incident to it.

The normal router state is *Active*. If a new fault is detected, the router enters the *State Computation* stage and computes its status and the ones of its incident links. A router is deactivated if it has 3 or more broken TX or RX links. If a router has 2 broken TX links in different dimensions, i.e., X and Y, the router is marked as a TX Concave (TC) router. Similarly, an RX Concave (RC) router has 2 broken RX links in different dimensions. If the status of a router, i.e., active or deactivated, and TC or RC, or status of any link, i.e., broken or functional, incident to it did change, the router broadcasts the new status information to the 4 neighboring routers in the East (E), West (W), South (S), and North (N), in the *Share Information* stage via a Triple Modular Redundancy (TMR) protected serial wire. Otherwise, the router enters the *Wait Updating* stage.

Each TC (RC) router sends a TC (RC) flag to its neighbor if the TX (RX) link to (from) that neighbor is still functional. Upon receiving a TC (RC) flag from one direction, e.g., W, the router checks if it has a faulty TX (RX) link in an orthogonal direction, i.e., N or S in this case, and if the neighbor where the flag comes from also has a faulty TX (RX) link in the same direction. If this holds true, the router forwards the TC (RC) flag to the next neighbor, i.e., the neighbor in the W in this case. If a router has received TC (RC) flags from two opposite directions in the same dimension, e.g., W and E, and has a broken TX (RX) link in another dimension, i.e., N or S, the router and all links incident to it are deactivated.

If status information from a neighboring router is received while being in the *Wait Updating* stage before the waiting time up, the router computes the local status again. Otherwise, the router returns to the *Active* stage. The waiting time is

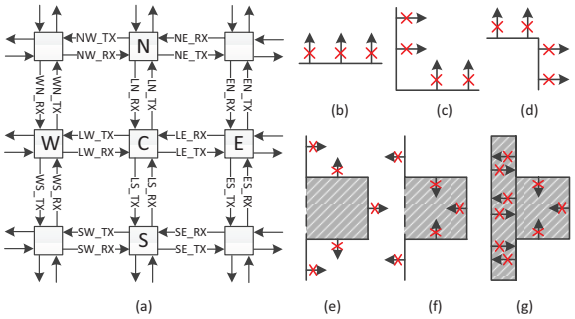


Fig. 2. Validated fault pattern. (a) Routers and links seen by router C; (b-g), Fault Patterns can be tolerated by the proposed RA.

set to be equal with the longest time required to transmit the status information from one NoC edge, e.g., the W edge, to another edge, i.e., the E edge. We note that after the fault pattern is validated, each router is aware of the statuses of 4 routers and 24 links adjacent to it, as illustrated in Fig. 2(a). Data transmission pausing when a new fault is detected and resuming after the fault pattern is validated can be managed with the strategy proposed in [16].

The most basic fault pattern that can be tolerated by the proposed RA is depicted in Fig. 2(b), where adjacent links in the same direction (N in this case) are broken. We note that the number of faulty links and their directions are not restricted in any fault pattern. When such *Fault Wall* exists, only the data transmission in that direction is blocked. The direction of a fault wall is the direction of the broken links that form the fault wall. More complex fault patterns, e.g., the ones illustrated in Fig. 2(c-g), can be formed by several fault walls, which have different fault directions. Some routers may be deactivated to make the fault patterns solid as in Fig. 2(e-g). We note that all fault patterns that can be tolerated by SFRT RAs are tolerable by the proposed RA.

When a link L is broken, some links around it have the highest probability to be utilized by misrouted messages. We refer to these links by employing the concept of *misrouting-contour*, which refers to the neighboring links that can be utilized to create an alternative path to route a packet from L's source to L's destination. For example, the misrouting-contour of LE_TX in Fig. 2 is composed of LN_TX, NE_TX, EN_RX, LS_TX, SE_TX, and ES_RX. We note that there is a nonzero probability that a link in the misrouting-contour of a broken link is also broken.

IV. Proposed Routing Algorithm

In the NoC fault free area, messages are transmitted according to the underlying RA. When a message is blocked by a fault wall, the proposed RA applies. In this paper, we employ the optimal fully adaptive RA Opt-y [17] as the underlying RA, but note that other RAs can also be utilized.

A. Turn Rules

In a 2D mesh NoC the direction of each link is WE, EW, SN, or NS according to the position of its source and

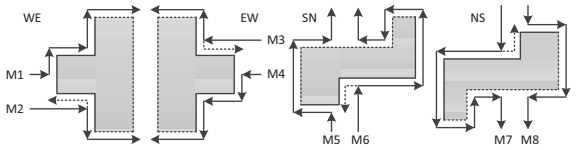


Fig. 3. Misrouting direction of different messages. The dashed boarder of the shadows may not be fault walls.

destination routers. A message which is transmitted from node n_s=(x_s, y_s) to node n_d=(x_d, y_d) is labeled as WE if $x_d > x_s$, or EW if $x_d < x_s$. When the message reaches its destination column, the message type changes to NS if $y_d > y_s$ or SN otherwise and cannot change its type again. WE and EW messages are row messages, while NS and SN messages are column messages [8].

Among the admissible output ports provided by Opt-y, one is determined by e-cube RAs [8], e.g., XY. We name the hop via that port as *e-cube hop*, and the others as *adaptive hops*. The e-cube hops for WE, EW, NS, and SN messages are E, W, S, and N ports, respectively.

A message's status is normal if it is a row message and the e-cube hop is not blocked; or 2) it is a column message, the e-cube hop is not blocked, and the current router is in the destination column. Otherwise, the message status is misrouting. If the e-cube hop of a normal message is on the misrouting-contour of a broken link, only the e-cube hop can be used. If an adaptive hop is on the misrouting-contour of a broken link, the adaptive hop cannot be used.

When a message is blocked by a fault wall, it is misrouted around the fault wall along the misrouting-contours of the broken links. The default misrouting direction is clockwise. However, the misrouting direction should be counter-clockwise if with the default misrouting direction row messages are forced to return to the previous column or column messages are forced to return to the previous row by another fault wall or an NoC edge. This can be anticipated by checking if a TC flag has been received from the port in the clockwise direction. For example, in Fig. 3, messages M1, M4, M5, and M8 are misrouted in clockwise direction, while messages M2, M3, M6, and M7 are misrouted in counter-clockwise direction as otherwise they will be forced to return to the previous column or row as illustrated by the dashed arrows.

For a misrouted row message, its status becomes normal when the e-cube hop, i.e., E or W, is functional. Thus we can simply misroute row messages to S or N until a functional e-cube output port is found. With this routing method, EW links are never used by WE messages, and WE links are never used by EW messages.

To avoid livelock as illustrated in Fig. 4(a), we use the localized routing scheme proposed in [12] to misroute column messages. When a column message is blocked, it anticipates the shortest path to the destination router of the broken link. The misrouting path is stored in the head flit and is dynamically adjusted in each router in case links in the anticipated misrouting path are also broken. For example, when an SN

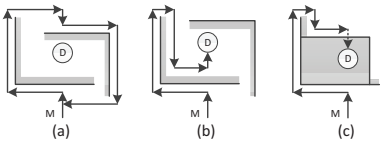


Fig. 4. Misrouting of column messages. The shadows indicate the directions of fault walls. (a) livelock occurs; (b) destination is reached; (c) destination is not reachable

message is blocked by a broken SN link, the shortest path to the next router in the same column is to take the W, N, and E (WNE) hops. If the N hop is blocked after the W hop, the remained misrouting path changes to WNEE. When the message is in the destination column again, its status returns to normal if the next e-cube hop is not blocked (Fig. 4(b)). If the e-cube hop is blocked again and the destination router is still in the North, the message starts another misrouting process. If the destination router is already in the South but the NS link is broken, the destination router must be broken or deactivated and thus unreachable (Fig. 4(c)). In this case, the local Process Unit (PU) absorbs the message and sends an message to the source router to report the error.

With this routing method, NS links are used by SN messages only when their source routers have received RC flags from the S ports, e.g., when M6 is routed to the south in Fig. 3. Similarly, SN links are used by NS messages only when their source routers have received RC flags from the N ports, e.g., when M7 is routed to the north in Fig. 3.

SN Messages never make E-S-W or W-S-E turns. According to the turn rules, a message turns to the S after one E (W) hop only because the destination column is reached or is still in the E (W). Thus it will not turn to the W (E) again. Similarly, NS messages never make E-N-W or W-N-E turns.

B. VC Usage Constraint

When messages are misrouted, some forbidden turns in Opt-y are utilized. To avoid deadlock, extra constraints to the VC usage besides the Opt-y ones must be applied.

Opt-y requires two VC classes, Y_1 and Y_2, in the N and S directions. Among all the turns, two 90^o turns, N-W and S-W, using Y_1 are prohibited, and the 0^o turns from Y_2 to Y_1 are allowed only when the message does not need to route further west. In our proposal, 3 VCs, labed as C_0, C_1, and C_2, are required. C_2 is used as Y_1, C_0 and C_1 are used as Y_2.

When messages are transmitted on misrouting contours of broken links, whether the message status being normal or misrouting, extra VC usage constraints are in place: C_0 is reserved for row messages on the misrouting-contours of broken NS or SN links. C_1 is reserved for NS messages on the misrouting-contours of broken NS links and in SN links whose source routers have received RC flags from their S ports. C_2 is reserved for SN messages on the misrouting-contours of broken SN links and in NS links whose source routers have received RC flags from their N ports. With this VC usage strategy, VCs are only reserved when necessary. For example, in a link which is only on the misrouting contour of a broken

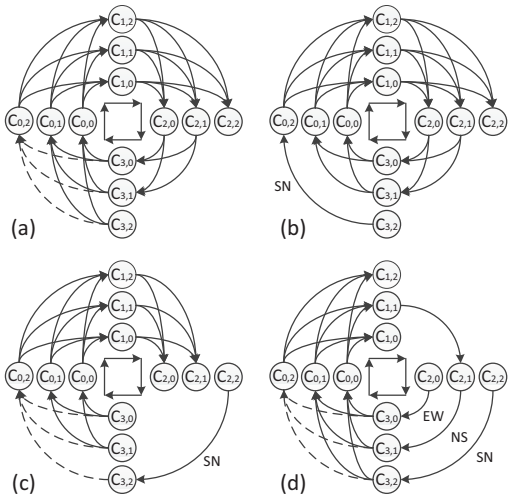


Fig. 5. Channel dependency graphs.

EW link, C_0 is reserved to row messages, while C_1 and C_2 are freely used by all column messages. We note here that if extra VCs are available in the NoC, they can be freely used by all types of messages.

C. Deadlock Freeness

To prove that the proposed RA is deadlock free, we just need to check if deadlock can happen when turns forbidden by Opt-y are used, as it is known that Opt-y is deadlock free [17]. The clockwise Channel Dependency Graph (CDG) of Opt-y is depicted in Fig. 5(a). In the figure, $C_{0,*}$, $C_{1,*}$, $C_{2,*}$, and $C_{3,*}$ are VCs in SN, WE, NS, and EW links, respectively. The turns that are always allowed are illustrated with solid arrows and the turns that are allowed in certain conditions are illustrated with dashed arrows.

Dependency $C_{3,2} \rightarrow C_{0,2}$ only occurs when SN messages are forced by broken SN links to make W-N turns (see Fig. 5(b)). On the misrouting contours, i.e., in the EW and SN links involved in the W-N turns, $C_{*,2}$ is reserved to SN messages. While other channel dependency remain the same as Opt-y. Thus the CDG in Fig. 5(b) is still deadlock free.

S-W turns from $C_{2,2}$ to $C_{3,2}$ only occur to SN messages. The NS and EW links involved in the turn are on the misrouting-contour of broken EW and SN links, respectively. Thus $C_{*,2}$ is reserved to SN messages in these links. According to the turn rules, the other types of messages do not make S-W turns if their statuses are normal, as illustrated in Fig.5(c). The dependency $C_{1,2} \rightarrow C_{2,2} \rightarrow C_{3.2}$ does not exist because SN messages never make E-S-W turns. We can observe that the CDG in the figure is acyclic and thus is deadlock free. If EW messages are forced by broken EW links to make S-W turns, they can only use $C_{*,0}$ in the SN and EW links involved in the turns. Because EW messages never use WE links, the turns from $C_{1,*}$ to $C_{2,0}$ do not exist (see Fig. 5(d)). When NS messages are misrouted, they use $C_{*,1}$ only, thus the turns from $C_{1,0}$ or $C_{1,2}$ to $C_{2,1}$ do not exist. Thus the CDG in

Fig.5(d) is also deadlock free.

By means of a similar analysis, we can also prove that the counter-clockwise CDG is deadlock free, thus we can conclude that the proposed routing algorithm is deadlock free.

V. Evaluation

To put the implication of the proposed RA in a better practical prospective, we evaluate and compare its figure of merit with the one of tightly related FT RAs, i.e., the SFRT algorithm [9] and the routing table based algorithm Ariadne [13]. We note that SFRT minimally requires 3 VCs which are statically reserved to row, NS, and SN messages around each fault region, while Ariadne allows all VCs be freely used by any message type. For a fair comparison, the underlying RA is Opt-y in all cases and every RA is implemented in the context of an 8×8 2D mesh NoC platform [18] at RTL level by using Verilog HDL. The baseline router has 3 pipeline stages and each VC buffer is 4-flit deep.

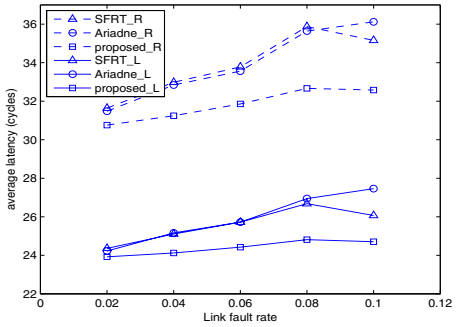
In practice, it is beneficial to utilize more VCs than the minimum, e.g., 3 VCs for our proposal and SFRT, to eliminate the Head of Line (HOL) blocking issues in the NoC system. In our experiments, 4 VCs are utilized by each RA, with one of them being freely used by all message types.

A. Synthetic Traffic

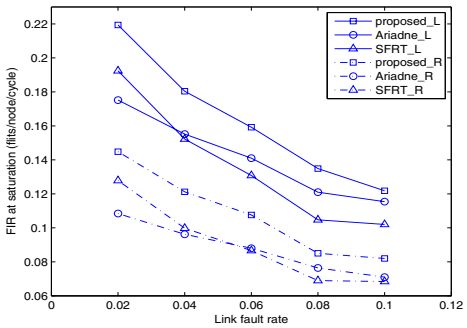
The three RAs' performance is evaluated in NoCs with different fault rates under different synthetic traffic patterns. The average transmission latency at light traffic load, i.e., 0.02 flits per node per cycle, and the saturation points, i.e., the Flit Injection Rate (FIR) for which the average transmission latency approaches infinity, of the considered RAs are illustrated in Fig. 6. Each data point is derived by averaging the results of 100 different fault patterns with the same fault rate and traffic pattern. In the random traffic, the message destinations are uniformly distributed throughout the NoC, while in the localized traffic, 50% of the messages are destined to the 8 nodes adjacent to the source node, which is the case for optimized task mappings, thus can better reflect the system performance in practice.

Although the proposed RA can also be classified into the second RA group (see Section II), it can utilize more resources and has more VC usage flexibility than SFRT, thus has better performance. As we can observe in Fig. 6, when compared with SFRT, our proposal provides a transmission latency reduction of 6% and 5%, and a saturation point increase of 20%, for random and localized traffic, respectively. Note that many routers are deactivated when the link fault rate increases from 0.08 to 0.1 and the SFRT strategy is applied. In such scenario, the total number of packets injected into the NoC per cycle is decreased and the traffic load becomes lighter, thus the average transmission latency is reduced.

Ariadne explores the routing paths between any two routers during the NoC reconfiguration which is triggered by the detection of a new fault. The routing paths, although with limited adaptive capability, are fixed for any traffic pattern. Although the proposed RA has less VC usage freedom than



(a) Average latency;



(b) Saturation points.

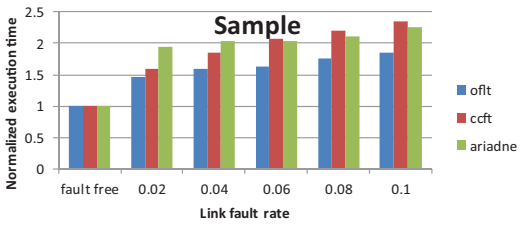
Fig. 6. NoC performance at different fault rates and traffic patterns. In the legend, R means random traffic pattern, and L means localized traffic pattern.

Ariadne around the faults, it transmits messages according to the underlying RA, i.e., Opt-y, in the fault free area, and thus has more balanced traffic distribution. When the link fault rate is low (0.02), the fault free area in the NoC is large, thus our proposal has 34% and 25% higher saturation points than Ariadne for random and localized traffic, respectively. As the fault rate increases, the fault free area shrinks. When the fault rate is 0.10, our proposal still has 15% and 6% higher saturation points than Ariadne for random and localized traffic, respectively. In all the evaluated fault patterns, our proposal has slightly lower ($< 5\%$) transmission latency and on average 22% and 14% higher saturation points than Ariadne for random and localized traffic, respectively.

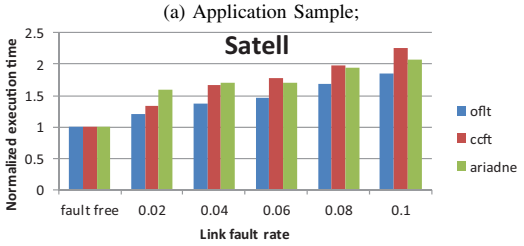
B. Real Traffic

The RAs are also evaluated with recorded traffic patterns derived from a realistic traffic MCSL benchmark suite [19]. Each traffic pattern is a trace of message transmission captured from a real application. The normalized execution time of the two considered applications, i.e., *Sample* and *Satell*, in different fault circumstances is illustrated in Fig. 7.

Although the execution time of an application can be affected by multiple issues, including traffic load, fault pattern, routing algorithm, and so on, our proposal out performs SFRT and Ariadne for both evaluated applications. In particular, the entire execution time of *sample* and *satell* when the proposed



(a) Application Sample;



(b) Application Satell.

Fig. 7. NoC performance with real applications.

RA is used is on average 17% and 16% less than that of SFRT, and 20% and 16% less than that of Ariadne, respectively.

C. Silicon Area Cost

Routers equipped with different RAs are synthesized by using the Synopsys Design Compiler and the TSMC 65nm technology. The silicon area overhead corresponding to different RAs are presented in Table I. From the table we can observe that embedding SFRT, the proposed RA, and Ariadne into a baseline router increases the silicon area cost by 8%, 11%, and 15%, respectively, when the NoC size is 8×8. The proposed RA requires more area than SFRT because the two unidirectional links in each interconnection are separately considered thus more registers are required to record the status of routers and links. It is worth to mention that the area cost of our proposal and SFRT is independent on the NoC size, while the routing table size in Ariadne increases proportionally with the number of NoC routers. For example, for a 10×10 NoC, the area overhead introduced by Ariadne increases to 21%.

VI. Conclusions

In this paper, we proposed a distributed logic based Routing Algorithm (RA) able to effectively utilize the UnPaired Functional (UPF) links incident to active routers, and thus to achieve more graceful performance degradation than state of the art RAs. The basic fault pattern tolerated by the proposed RA is a fault wall, which is composed of adjacent broken links with the same outgoing direction. Messages are routed around the fault walls along the misrouting contours of the broken links. The proposed RA requires 3 Virtual Channels (VCs) and dynamically reserve them to different messages to avoid deadlock. Our experiments indicate that, for random and localized traffic patterns, we achieve an average saturation throughput 20% higher than the Solid Fault Region Tolerant (SFRT) RA, and 22% and 14% higher than the Ariadne routing table based RA, respectively. For the real applications, *sample* and *satell*, our proposal requires a routing execution time with at least 16% lower than both SFRT and Ariadne. Synthesis results with TSMC 65nm technology indicate that, embedding the proposed RA into a baseline router results in 11% area overhead, which is only 3% higher than that of SFRT. In contrast, Ariadne area overhead is 15% for an 8×8 NoC and increases to 21% for a 10×10 NoC.

TABLE I
Area overhead of different routing algorithms. The NoC size is 10×10 for Ariadne* and 8×8 in other cases

	Baseline	SRFT	Proposed	Ariadne	Ariadne*
Area (μm^2)	49716	53839	55282	57011	60286
overhead	100%	108%	111%	115%	121%

References

[1] C. Chen and S. Cotofana. A novel flit serialization strategy to utilize partially faulty links in networks-on-chip. In *Proceedings of NOCS*, pages 124–131, May 2011.

[2] J. Kim, C. Nicopoulos, and D. Park. A gracefully degrading and energy-efficient modular router architecture for on-chip networks. In *Proceedings of ISCA*, pages 4–15, 2006.

[3] W. Tsai, D. Zhen, S. Chen, and Y. Hu. A fault-tolerant noc scheme using bidirectional channel. In *Proceedings of DAC*, pages 918–923, June 2011.

[4] R. Tamhankar and et. al. Timing-error-tolerant network-on-chip design methodology. *IEEE Trans. on CAD-ICS*, 26(7):1297–11310, July 2007.

[5] M. Ebrahimi, M. Daneshtalab, J. Plosila, and H. Tenhunen. Minimal-path fault-tolerant approach using connection-retaining structure in networks-on-chip. In *Proceedings of NOCS*, pages 1–8, April 2013.

[6] C. Glass and L. Ni. Fault-tolerant wormhole routing in meshes without virtual channels. *IEEE Trans. on PDS*, 7(8):620–636, June 1996.

[7] Z. Zhang, A. Greiner, and S. Taktak. A reconfigurabel routing algorithm for a fault tolerant 2d-mesh network-on-chip. In *Proceedings of DAC*, pages 441–446, June 2008.

[8] S. Chalasani and R. V. Boppana. Communication in multicomputers with nonconvex faults. *IEEE Trans. on Computers*, 46(5):616–622, May 1997.

[9] C. Chen and G. Chiu. A fault-tolerant routing scheme for meshes with nonconvex faults. *IEEE Trans. on PDS*, 12(5):467–475, May 2001.

[10] C. Chen and S. Cotofana. An effective routing algorithm to avoid unnecessary link abandon in 2d mesh noc. In *Proceedings of DSD*, pages 374–379, September 2013.

[11] S. Kim and T. Han. Fault-tolerant wormhole routing in mesh with overlapped solid fault regions. *Parallel Computing*, 23(13):1937–1962, December 1997.

[12] A. Vitkovskiy, V. Soteriou, and C. Nicopoulos. A highly robust distributed fault-tolerant routing algorithm for nocs with localized rerouting. In *Proceedings of INA-OCMC*, pages 29–32, January 2012.

[13] K. Aisopos, A. D. abd L. Peh, and V. Bertacco. Ariadne: Agnostic reconfiguration in a disconnected network environment. In *Proceedings of PACT*, pages 298–309, October 2011.

[14] F. Chaix, D. Avresky, N. Zergainoh, and M. Nicolaidis. Fault-tolerant deadlock-free adaptive routing for any set of link and node failures in multi-cores systems. In *Proceedings of NCA*, pages 52–59, June 2010.

[15] V. Puente, J. Gregorio, F. Vallejo, and R. Beivide. Immunet: Dependable routing for interconnection networks with arbitrary topology. *IEEE Trans. on Computers*, 57(12):1676–1689, December 2008.

[16] Y. Kang, T. Kown, and J. Draper. Fault-tolerant flow control in on-chip networks. In *Proceedings of NOCS*, pages 79–86, May 2010.

[17] L. Schwiebert and D. N. Jayasimha. Optimal fully adaptive wormhole routing for meshes. In *Proceedings of Supercomputing*, pages 782–791, November 1993.

[18] Y. Lu, J. McCanny, and S. Sezer. Exploring virtual-channel architecture in fpga based networks-on-chip. In *Proceedings of SOCC*, pages 302–307, September 2011.

[19] W. Liu, et al., A NoC Traffic Suite Based on Real Applications. In *Proceedings of ISVLSI*, pages 66 – 71, July 2011.