Online Reconfigurable Routing Method for Handling Link Failures in NoC-based MPSoCs

Poona Bahrebar and Dirk Stroobandt
Department of Electronics and Information Systems (ELIS), Ghent University
iGent, Technologiepark-Zwijnaarde 15, 9052 Ghent, Belgium
{*poona.bahrebar, dirk.stroobandt*}*@ugent.be*

***Abstract*—As silicon features approach the atomic scale, the Networks-on-Chip (NoCs) are becoming more susceptible to faults. Resiliency to device failures is, therefore, a key objective in the design of the Systems-on-Chip (SoCs). This paper seeks to address reliability by presenting a routing algorithm for 2D mesh NoCs. Using the proposed method which is designed based on the Abacus Turn Model (AbTM), the healthy paths can be dynamically configured according to the location of faults and congestion in the network. As a result, not only the functionality of the network is maintained in the vicinity of faults, but also a high performance communication can be provided. The presented technique is an adaptive, distributed, deadlock-free, and congestion-aware routing method which does not require routing tables or virtual channels. The experimental results demonstrate the reliability of NoC against multiple link failures with a small hardware overhead penalty.**

***Index Terms*—Network-on-Chip (NoC); fault-tolerance; routing algorithm; reconfiguration; deadlock; turn model;**

I. Introduction

The aggressive technology scaling in the semiconductor industry has introduced two salient challenges: (1) efficient connection of the increasing number of on-chip resources, and (2) effective management of the decreasing transistor reliability [1]. The first problem concerns the limitation of the bus-based and ad-hoc interconnects which cannot scale well with the growing complexity of the Chip Multi-Processors (CMPs). As a result, the *Network-on-Chip (NoC)* paradigm has emerged as a promising alternative to address the heavy communication demands of such complex platforms. A NoC is a shared and distributed interconnection network of programmable routers connected by links which are integrated onto a single chip [2].

The second problem arises from the extreme device scaling in the VLSI circuits. In fact, on-chip interconnects implemented with deep submicron technology running at GHz clock frequencies are prone to failures which threaten the reliability of MPSoCs by low yield and device wear-out. A faulty Intellectual Property (IP) core may be quarantined or powered down entirely since it does not affect the packet transmission between the other cores as long as the connected router and links can continue functioning. However, once a router or link has failed, the faulty component cannot be simply discarded since it will result in the blocking of the packets inside the network. Thus, the interconnect architecture must be able to tolerate partial failure due to its critical role in holding all of the components of the system together [1], [3].

The research in the field of NoC reliability falls into two main categories: (1) intra-router faults (i.e. faults within the *switches*), and (2) inter-router faults (i.e. faults within the *links* interconnecting the switches) [4]. Furthermore, the failures are classified as either *permanent* or *transient*. Permanent faults are introduced by physical damages, such as manufacturing defects and device wear-out. On the contrary, transient faults are introduced by power supply noise, ground bounce, interconnection noise, etc. Transient faults are beyond the scope of this paper since they are temporary and unpredictable [3].

The incorporation of fault-tolerance into NoC design mainly revolves around the *routing methods*. Routing algorithms control the distribution of the traffic throughout the network by determining the paths to deliver the packets from source to the destination. *Deterministic* routing algorithms always use a fixed predetermined path to transmit the packets, oblivious to the state of the network. *Adaptive* routing algorithms, on the other hand, are able to outperform their deterministic counterparts in avoiding the congested regions since the packets can take multiple paths to reach the destination [2], [5].

This paper improves upon our previous work presented in [6] where an **A**daptive and **Re**configurable **F**ault-tolerant routing algorithm (aptly called AReF) was proposed based on the Abacus Turn Model (AbTM) [7]. In [6], the ability of AReF to tolerate faults is narrowed down to the single switch failures. The proposed Enhanced-AReF (E-AReF) is designed to support faulty links such that an 8×8 mesh network remains 100% and 99.94% reliable against single and two link failures, respectively. Inheriting the properties of AReF, E-AReF is a reconfigurable, highly adaptive, and distributed routing scheme which does not rely on routing tables or Virtual Channels (VCs). Moreover, the deadlock-freedom is ensured in E-AReF by following the AbTM rules.

The reminder of this paper is organized as follows. The related works are studied in Section II followed by an explanation of AbTM in Section III. Section IV details the mechanism of E-AReF. Section V is devoted to the simulation results and discussion. Finally, the conclusions are drawn in Section VI.

II. Related Work

Most of the existing routing methods [1], [8]–[10], [13]–[15] rely on detour strategies to sustain the reliability in NoCs. Thus, once a packet faces a faulty component, it is *rerouted* around the fault to reach the destination [16].

978-1-5090-2520-6/16/$31.00

TABLE I
COMPARISON OF E-AREF WITH OTHER FAULT-TOLERANT METHODS

Method	Topology	Fault	Reliability	Adaptive	Minimal	Deadlock-free	Reconfiguration	VCs	Routing Table
Fick et al. [1]	2D Mesh/Torus	link	high	✗	✗	✗	offline	✗	✓
Vicis [8]	2D Mesh/Torus	link	high	✗	✗	✗	offline	✗	✓
Ren et al. [9]	2D Mesh	link	high	✓	✗	✓	✗	✗	✗
uDIREC [10]	Agnostic	link	high	✓	✓	✓	offline	✗	✓
ARIADNE [11]	Agnostic	link	high	✓	✓	✓	offline	✗	✓
MiCoF [3]	2D Mesh	switch	limited	✓	✓	✓	✗	2	✗
MAFA [12]	2D Mesh	link	limited	✓	✓	✓	✗	4	✗
AReF [6]	2D Mesh	switch	limited	✓	✓	✓	online	✗	✗
E-AReF (proposed)	2D Mesh	link	limited	✓	✓	✓	online	✗	✗

However, rerouting is a costly solution as it may introduce *deadlock* or *livelock* which are catastrophic to a network. In general, the methods employing detour can be divided into three main groups, depending on the mechanism they use to create a deadlock-free path around the faulty region: some approaches [1], [9], [13] opt for a more restricted set of path selection heuristics such as the *turn model* [5], some [14] use VCs, and some [1], [8], [10], [14], [15] require offline reconfiguration of routing tables.

The *partially adaptive* routing algorithms designed based on the turn model have a limited fault-tolerance capability (e.g. up to $n-1$ faults in an n-dimensional mesh). This is the result of the routing restrictions imposed by the turn model in order to eliminate the formation of cycles in the network. Exploiting such inferior routes may further aggravate the network performance [16]. Contrarily, the *fully adaptive* routing methods designed using VCs provide a better fault-tolerance capability. However, incorporating VCs imposes extra hardware requirements and complex control logic to the routers [5]. The table-based methods also incur significant overheads to store and compute the routing tables [16].

Regardless of the mechanism being used, most of the detour algorithms suffer from two major drawbacks: (1) The hotspots are more likely to be created around the faults when rerouting is performed. Furthermore, the aggregated congestion by taking longer paths degrades the performance, significantly [3]. (2) Strict restrictions on the number and location of faults do not permit arbitrary fault patterns within the network.

Besides routing algorithms, *redundancy* is an alternative approach to boost the robustness in NoCs. However, redundancy is generally associated with high overhead which is mainly comprised of extra latency and dramatic hardware costs [17].

Several studies [3], [8], [10], [16], [18], [19] look specifically into router design to improve the NoC reliability through microarchitectural modifications. For instance, the mechanism applied in the MiCoF approach [3] connects the input and output router ports directly by adding redundant wires to maintain the connectivity of the network and bypass the faulty router in the vicinity of faults.

Reconfiguration techniques have also been widely utilized to design resilient NoCs. In fact, most of the fault-tolerant routing algorithms perform route-reconfiguration to bypass the faulty elements. Reconfiguration is approached by ARIADNE [11] and Hermes [15] for link failures, by [19] for switch failures, and by Vicis [8] for unconstrained faults. Vicis and uDIREC [10] present unified frameworks for diagnosis and subsequent reconfiguration that are robust to a large number of permanent failures. This comes at the expense of a costly Built-In Self-Test (BIST) unit and complex reconfiguration process. In general, reconfiguration solutions require some extra logic to enable alternative paths compared to a plain NoC implementation. Besides, it is also necessary to avoid deadlocks that can result from the interplay between the old and the new routings in the transition phase, while the network is still being reconfigured [17].

Table I presents a comparison of the proposed method with the recent approaches and their attributes. As can be understood from the literature, it is likely that a combination of different techniques may be required to address fault-tolerance, given the numerous fault situations that may occur in a network. Recent comprehensive surveys on design approaches to combat failures in NoCs can be found in [17] and [20].

III. ABACUS TURN MODEL (ABTM)

Before presenting the proposed scheme, it is necessary to elaborate on AbTM which is used to design E-AReF.

Considering that a 90-degree change of the traveling direction is called a turn [21], eight turns can be formed using the four directions, east (E), west (W), north (N), and south (S), in a 2D mesh. The two abstract cycles formed using these turns are illustrated in Fig. 1(a). The routing function for an interconnection network becomes deadlock-free by breaking the cycles formed by the turns. Chiu [21] proposed the Odd-Even (OE) turn model and proved that a network is deadlock-free if the rightmost columns in the clockwise and counter-clockwise cycles never appear. The OE permitted and

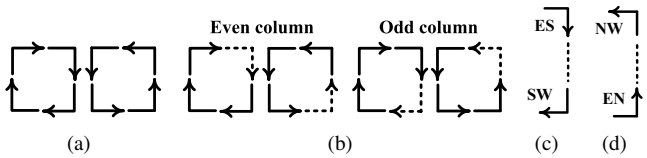

Fig. 1. (a) Clockwise and counter-clockwise cycles formed by eight turns in a 2D mesh [5], (b) permitted and prohibited turns using the OE turn model [21], (c) clockwise, and (d) counter-clockwise rightmost columns [7].

prohibited turns are shown in Fig. 1(b) by the solid and dashed arrows, respectively. Note that the permitted and prohibited turns in OE are static and cannot be changed at run-time [7].

Fu et al. [7] further extended the OE turn model to design a reconfigurable method which is called AbTM. In AbTM, the 2D mesh is resembled by an abacus such that each column corresponds to a wire with two sliding and separately controlled clockwise and counter-clockwise *beads*. The main idea of AbTM is based on the fact that the clockwise rightmost column is formed when an ES turn occurs above a SW turn. Analogously, the counter-clockwise rightmost column is formed when an EN turn takes place below a NW turn (Fig. 1(c)-(d)). Thus, the AbTM rules are regulated as [7]:

Rule 1**:** *In each column, there is a clockwise bead, above which all ES turns are prohibited, and below which all SW turns are prohibited.*

Rule 2**:** *In each column, there is a counter-clockwise bead, above which all NW turns are prohibited, and below which all EN turns are prohibited.*

Rule 3**:** *The clockwise (counter-clockwise) bead holder allows all of the clockwise (counter-clockwise) turns.*

Thus, wherever the beads are located, the clockwise and counter-clockwise rightmost columns cannot be formed utilizing the AbTM rules, and the deadlock-freedom is guaranteed. The distribution of the AbTM prohibited turns in a 4×4 mesh is illustrated in Fig. 2. The solid and hollow ellipses represent the clockwise and counter-clockwise beads, respectively [7].

By moving the beads along a column, the permitted and prohibited turns, and consequently, the degree of routing adaptiveness varies. Taking advantage of this important possibility, the adaptiveness can be adjusted according to the traffic pattern by moving the beads along each column. Thus, AbTM as an online reconfiguration strategy, can dynamically reconfigure itself to grant full adaptivity for the packets traveling towards the hotspots. Note that in AbTM, four turns (i.e WN, NE, WS, and SE) are *trivial* and always assumed to be enabled. On the contrary, the other four *critical* turns (i.e. ES, SW, EN, and NW) are enabled or disabled by moving the beads [7].

Figure 3 presents an example of the AbTM reconfigurable routing in a 3×3 mesh. Assume that the clockwise beads are located in the top row initially, and the source node 5 attempts to transmit messages to the destination node 0 for a long period. As shown in Fig. 3(a), there exists just one admissible path from 5 to 0 because the SW turns are prohibited in the routers below the beads. Since this deterministic path is highly prone to congestion, the source node makes a request to node 2 to activate the prohibited SW turn. Node 2 collects the requests and negotiates with the bead holder (node 8) to release the bead. Once the bead is passed to node 2, the SW turn is enabled in this node and there are two available paths between the source and destination, leading to a more balanced traffic distribution (Fig. 3(b)). Eventually, node 4 requests the bead movement in a similar fashion. The final states of the beads are shown in Fig. 3(c), where full adaptiveness is granted to this source-destination pair without using VCs [7].

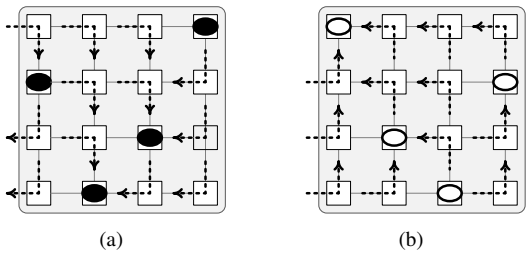

Fig. 2. Prohibited (a) clockwise, and (b) counter-clockwise turns using AbTM [7].

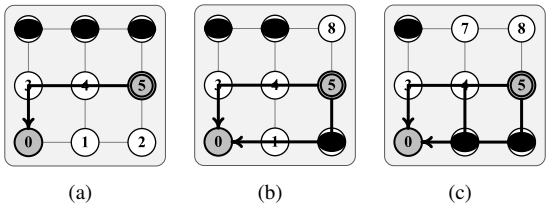

Fig. 3. AbTM reconfigurable routing example [7].

Moving beads needs to be handled with extreme care in AbTM to ensure the deadlock-freedom. In [7], a safe operation called *bead passing* is introduced to keep the network deadlock-free during the reconfiguration. Furthermore, two reconfigurable routing algorithms (i.e. arm-wrestling and tug-war) are proposed to evaluate the requests and determine the direction of the bead movement [7].

IV. Proposed Routing Method

In general, designing a routing protocol that is both fault-tolerant and deadlock-free poses a major challenge [6]. Thus, most of the adaptive fault-tolerant routing methods proposed in the literature [3], [12], [14] exploit VCs to maintain the reliability and deadlock-freedom in the presence of faults.

As indicated in [6], AReF is the first fault-tolerant routing method which is able to minimize rerouting and ensure deadlock-freedom by reconfiguring the healthy minimal paths without utilizing VCs or routing tables. In this work, similar to AReF [6], we adopt the idea of the AbTM reconfigurable routing method and combine that with fault-tolerance. The packet routing regulations in AReF were designed for a single faulty switch. In this paper, we devise E-AReF targeting the faulty links in the network. Moreover, similar to AReF, the packets can be routed through alternative paths using E-AReF.

A. Fault Model

The routers in a NoC are connected through bidirectional (i.e. pairs of unidirectional) links. We have employed a coarse-grained fault model similar to ARIADNE [11] where link failures are assumed to be bidirectional. In other words, a faulty link in each direction is modeled by tagging the entire bidirectional link as faulty. A NoC with two broken links is illustrated in Fig. 4(a). Note that faults may be present on each of the constituent's unidirectional links or both. The corresponding undirected topology after the application of the fault model is shown in Fig. 4(b).

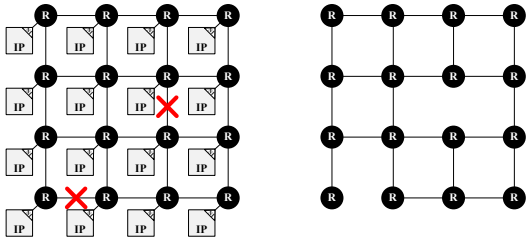

Fig. 4. (a) A 4×4 2D mesh NoC with two faulty links, and (b) the resulting network topology considering the fault model.

B. Fault Information

In order to avoid traversing unnecessary non-minimal paths, each router should be informed about the healthiness or defectiveness of its surrounding components (i.e. *fault information* or *look-ahead*). This information is used by the algorithm to make reliable routing decisions. As shown in Fig. 5, E-AReF requires each router to be aware of the status of its 16 surrounding links (i.e. 2-hop distance look-ahead). In the Minimal and Adaptive Fault-Tolerant Algorithm (MAFA) presented in [12], the required fault information is the same.

C. Tolerating Faulty Links

1) Tolerating Single Faulty Link: E-AReF is able to tolerate any single link failure with 100% reliability. The routing strategy to bypass a single faulty link is explained as follows:

In the cases where the destination is located on the NE, SE, NW, or SW of the source, the packets facing a faulty link can be routed through the minimal paths towards the destination. Fig. 6 illustrates all of the cases where a NE-ward packet faces a faulty link and the destination node, D, is one hop away from the current node, C, along both X and Y dimensions. Note that C can also refer to the source node throughout the text.

Consider case no. 1 in the figure where the paths from C to D are both healthy. In such case, the availability of the EN turn in the right side column is inspected. If the EN turn is permitted according to the current location of the counter-clockwise bead, the packet can be adaptively forwarded to either X or Y directions depending of the congestion condition of the east and north neighbors, respectively. Otherwise, the packet is transmitted to the Y direction. The reason why the Y direction (black path in the figure) is preferred rather than the X direction (blue path) is that using the Y direction, the packet takes a trivial NE turn to reach D. However, choosing the X direction requires the critical EN turn to be activated. Since bead passing is a costly operation, the paths which do not require a critical turn are preferred using E-AReF.

In case no. 2 where a broken link on the right side of C is detected, the packet has to take a NE turn as the only available path to D. This is also true for case no. 5 where the EN link is damaged. Cases 3 and 4 indicate faulty links on the north and northeast side of C, respectively. In those cases, taking the EN turn is inevitable. Hence, if the counter-clockwise bead on the right side of C is located above that node, C makes a request to the bead holder to release the bead. It is worth mentioning that in E-AReF, similar to AReF, the complaints related to the fault have the highest priority. Thus, once such a request is received by the bead holder, it will pass the bead immediately (considering the deadlock-free requirements) to form a path towards D. As can be seen, E-AReF is able to configure the available minimal paths using AbTM so that it is possible to bypass the fault in the network without rerouting the packets.

2) Tolerating Two Faulty Links: The strategy of E-AReF to bypass two faulty links is explained in the following, based on the relative position of the destination with respect to source.

First, we start with the NE-ward packets where the distance between C and D is one hop along both dimensions. All of the cases with two faulty links are illustrated in Fig. 7. If the faulty links affect just one of the minimal paths to D, the other minimal path is used for routing. This is clearly shown in cases 6 and 7 in Fig. 7(a) and (b).

In case no. 8, the NE and EN links are both broken. Since no minimal path is available between C and D, E-AReF inevitably should forward the packets through a non-minimal path. In such case, three scenarios may happen: The packets are normally sent to the Y direction. After two hops, the packet takes a trivial NE turn and then an ES turn to reach D by adjusting the location of the clockwise bead, if necessary. Such a non-minimal path is considered to be healthy assuming the number of faults is limited to two which are already detected. This scenario is shown in the left side of Fig. 7(c). The second scenario occurs when D is located on the top borderline of the mesh. Therefore, using the path of the first scenario is impossible. In this case, C sends the packet to the X direction where after two hops, the counter-clockwise bead has to be moved to form a path towards D. Although one of the last two turns in this non-minimal path is prohibited (dashed in the figure), E-AReF remains deadlock-free and the prohibited turn can be safely taken. This is due to the fact that a cycle

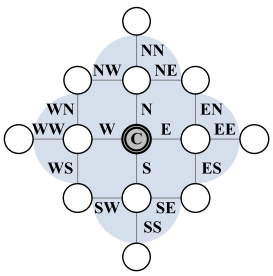

Fig. 5. The fault look-ahead in E-AReF.

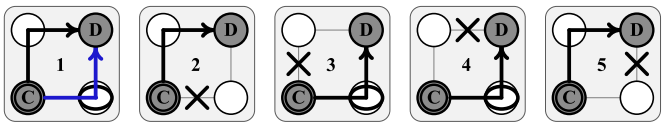

Fig. 6. Tolerating single faulty link in E-AReF using minimal paths when the destination is located on the NE and one hop away from the current node along both dimensions.

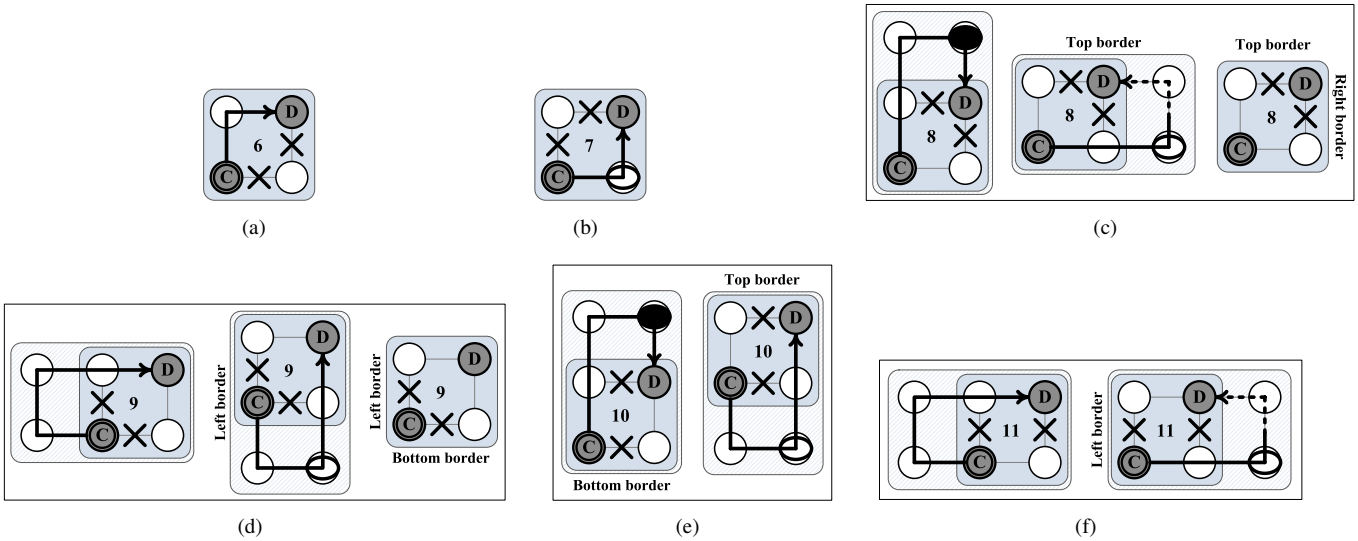

Fig. 7. Tolerating two faulty links in E-AReF when the destination is located on the NE and one hop away from the current node along both dimensions.

cannot be formed in the borderlines, as proven in [13]. Finally, the third scenario is shown in the right side of Fig. 7(c) where D is located on the NE corner of the mesh. As can be seen in the figure, nor a minimal neither a non-minimal path can be formed between C and D. Thus, the packets cannot reach the destination in this particular case and need to be dropped.

As depicted in in Fig. 7(d), the scenarios in case 9 are similar to those in case 8: The packets are normally forwarded to the west to be routed through a non-minimal path with trivial turns. But if C is in the left borderline, the non-minimal path in the south is selected for routing. However, the packets are dropped if C is located on the SW corner of the mesh.

Case no. 10 implies E and NE faulty links where the packets are sent to the north or south if C and D are on the bottom and top borderlines, respectively. It is noteworthy to mention that the packets can adaptively take one of those paths to reach D if both paths are available (i.e. C or D are not on the border).

Finally, when the N and EN links are faulty (case no. 11), C forwards the packet to the west to use a non-minimal path with trivial turns. If C is on the left borderline, the packet is sent to the east. The prohibited turn does not jeopardize the deadlock-freedom considering the broken links in the border.

Now consider all of the cases with two broken links where the distance from C to D is one and two hops along the X and Y dimensions, respectively. Among those cases, routing the packets when both of the faulty links fall within the scope of the look-ahead information of C (considering Fig. 5) is quite straightforward. Hence, only the cases where at least one of the defective links cannot be detected by C are shown in Fig. 8 due to the limited space. The faulty link which cannot be detected by C is marked in red in this figure. The cases with two broken links where the distance from C to D is two and one hops along the X and Y dimensions are similar to those of Fig. 8 and are not repeated here.

As shown in Fig. 8(a), when C detects a faulty link on the east, the packets are routed to the Y direction. After the first hop, the packet stands in one of the cases of Fig. 6 (case no. 4 or 5) and is routed accordingly to arrive at the destination. A similar routing strategy is pursued in Fig. 8(c).

Once the link on the north side of C is faulty and the second possible fault is undetected, the packets are duplicated and one copy is sent to the east and the other one to the west. As depicted in Fig. 8(b), this conservative strategy guarantees the packets to arrive at the destination assuming the number of faults is limited to two. This is due to the fact that depending on the location of the second fault, the packets following one of the paths may fail to reach the destination. Note that if C is on the left borderline, there is no need for duplication and the packet is simply sent to the east. A similar procedure is also followed in Fig. 8(d) where the NE link is broken. However, the packets can be transmitted through the minimal paths.

In Fig. 8(e) where the NN link is faulty, the packets are sent towards the Y direction. After the first hop, the remaining path is determined depending on the location of the second fault.

Lastly, the cases where the distance from C to D is one and two hops along the X and Y dimensions and all of the links within the scope of C are healthy are illustrated in Fig. 8(f). The packet is forwarded to the greater-distance dimension in which the offset to D is not zero. This technique which was employed in MAFA avoids to reduce the offset in one dimension into zero when the offset along the other dimension is greater than one. Therefore, the likeliness of finding a deadlock-free and shorter path towards the destination is increased. For instance, in this particular case, C sends the packet to the Y direction. As it is depicted in Fig. 8(f), after the first hop, the packet stands in one of the cases in Fig. 7(c).

When the distance from C to D is two hops or more along both dimensions, C forwards the packet to the healthy output channel (Fig. 9). If both of the neighbors are faulty, the routing approach is similar to Fig. 7(d). If both of the neighbors

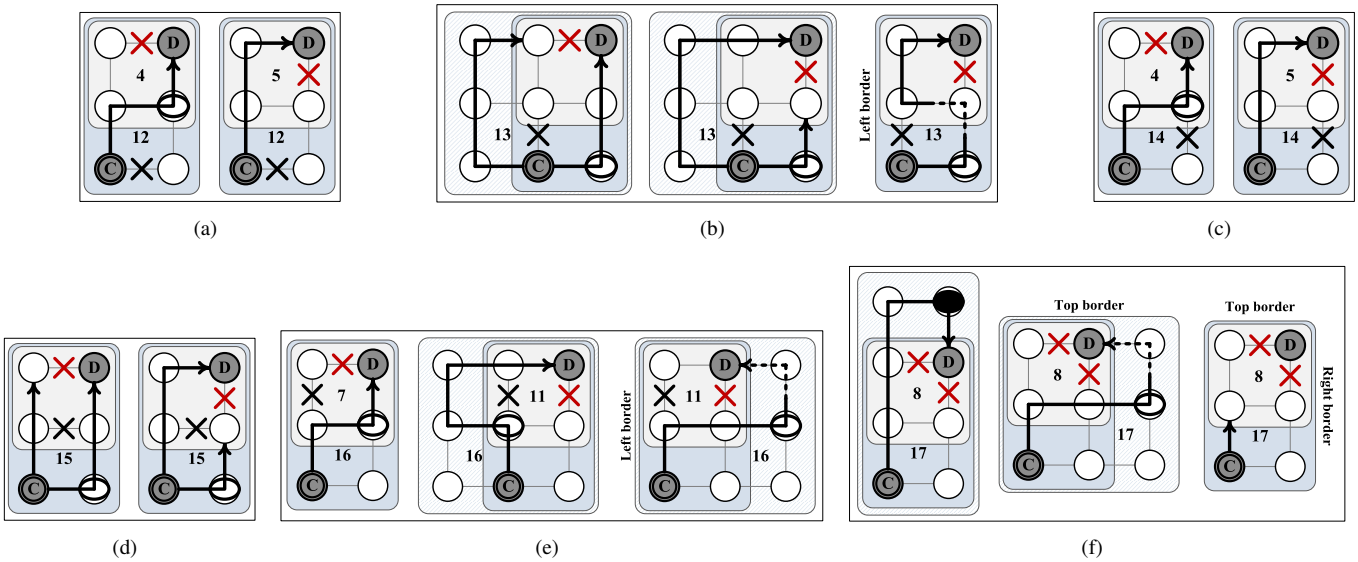

Fig. 8. Tolerating two faulty links in E-AReF when the destination is located on the NE and one and two hops away from the current node along the X and Y dimensions, respectively. The broken link which cannot be detected by the current node is marked in red.

are healthy, the less congested neighbor is selected to send the packet. Afterwards, the packet reaches one of the cases discussed previously. As a result, the degree of adaptiveness in E-AReF increases with the distance towards the destination. But when the packet approaches the destination and the distance is reduced, E-AReF behaves conservatively and its adaptivity is limited in order to bypass faults through deadlock-free and preferably minimal paths.

The routing policy in E-AReF for the NW-, SE-, and SW-ward packets is similar to the NE-ward packets. Thus, those cases are not repeated here. Furthermore, when the source and destination are located in the same row or column, the packets approach D through the minimal path as long as such a path is available. Following the same strategy as discussed in this section, the packet is rerouted through a non-minimal path if C detects a broken link. Note that the bead passing operation in E-AReF is the same as AReF to minimize contention and avoid starvation. More details can be found in [6].

In summary, the main idea behind E-AReF is to reconfigure the locations of the beads to activate the critical turns where necessary in order to take advantage of the available minimal paths without applying VCs. It is worth mentioning that VCs can be added to E-AReF to increase the performance.

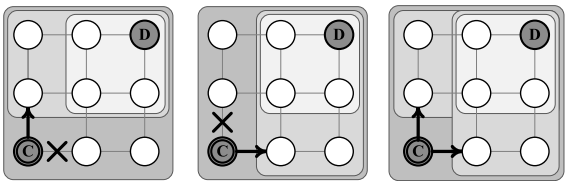

Fig. 9. Tolerating faulty links in E-AReF when the destination is located on the NE and two hops away from the current node along both dimensions.

D. Robustness Analysis

As discussed in the previous section, the network drops the packets if two faulty links are located in diagonal positions in the corners of the mesh. There are 4 such cases in a given network which cannot be supported by E-AReF. In order to analyze the robustness of the network using E-AReF, we first calculate the total number of different combinations where two links are broken in an $n \times n$ network.

$$N_{all\ combinations} = \binom{2n(n-1)}{2} = \frac{(2n^2-2n)(2n^2-2n-1)}{2}$$

where $2n(n-1)$ is the total number of links in an $n \times n$ mesh. Thus, reliability, R, can be calculated by:

$$R = 1 - \frac{4}{N_{all\ combinations}} = 1 - \frac{8}{(2n^2-2n)(2n^2-2n-1)}$$

As shown in Table II, E-AReF is highly robust against two faulty links in the network.

V. Simulation Results and Discussion

In this section, the efficiency of the proposed method is evaluated using a modified version of BookSim 2.0 cycle-based network simulator [2]. The configuration parameters are shown in Table III. The packet size is uniformly distributed between 5 and 10 flits. In each set of simulations, the location of the faulty link(s) is selected randomly. E-AReF is implemented according to the arm-wrestling [7] algorithm. The simulator was warmed up for $10,000$ cycles to be stabilized and the

TABLE II
Robustness of networks using E-AReF against two faults

Network size	4×4	6×6	8×8	16×16
Reliability	98.55%	99.77%	99.94%	99.99%

TABLE III
SIMULATION PARAMETERS

Parameter	Value
Network size & Topology	8×8 2D mesh
Data width	32 bits
Buffer (FIFO) size	8 flits
Congestion threshold (C_{th})	62.5% of the total buffer capacity

results were averaged over the next $200,000$ cycles. The MAFA routing algorithm [12] was selected for comparison against E-AReF as the most similar approach to tolerate link failures (see Table I). Unlike E-AReF, MAFA exploits two VCs along each dimension to ensure deadlock-freedom. In order to compare MAFA and E-AReF fairly, the MAFA buffer queues are equipped with 4 entries in both dimensions.

A. Performance Analysis

1) Uniform Traffic Profile: The average latency curves of E-AReF and MAFA routing methods as a function of the network's message injection rate (i.e. the number of flits injected into the network per cycle per node) are plotted in Fig. 10, for a network without fault, with a single faulty link, and two faulty links. The number of faults are indicated in the parentheses. As shown by simulations, employing MAFA results in a lower average latency compared to E-AReF. The reason is the reconfiguration strategy in E-AReF where the beads are moved according to the traffic history. In fact, the reconfiguration policy in E-AReF (and AbTM, in general) is mostly wrong under the uniform traffic. For instance, consider an interval where data is sent to the SE direction. E-AReF will adjust the locations of beads to provide more adaptivity to that direction. However, SE has the lowest probability to carry bursty traffic in the next interval, according to the uniform traffic pattern [7]. On the contrary, MAFA which provides a static adaptivity achieves better performance results under the uniform traffic for the network with or without broken links.

2) Hotspot Traffic Profile: In the hotspot traffic pattern, a few nodes in the network which are designated as hotspot nodes receive an extra proportion (h) of the traffic in addition to the regular uniform traffic. In our simulations, four hotspot nodes at the center of the network were selected with $h = 40\%$. The average latency curves of MAFA and E-AReF are plotted in Fig. 10. As discussed earlier, adaptivity improves the performance in congested networks to a high extent. Thus, E-AReF is able to outperform MAFA in tackling the congested areas of the network by providing a higher degree of adaptiveness, and thereby reducing the average communication delay. MAFA offers a fixed degree of adaptiveness to the packets in all of the regions of the network oblivious to the traffic condition. Therefore, the performance of MAFA drops significantly with the increasing congestion as a result of the faulty links in the network. E-AReF, on the other hand, is able to dynamically allocate more adaptivity to the bursty traffic which is highly crucial at the occurrence of fault. Note that for traffic profiles with a predictable pattern such as hotspot, the number of reconfiguration operations is small and does not affect the performance negatively.

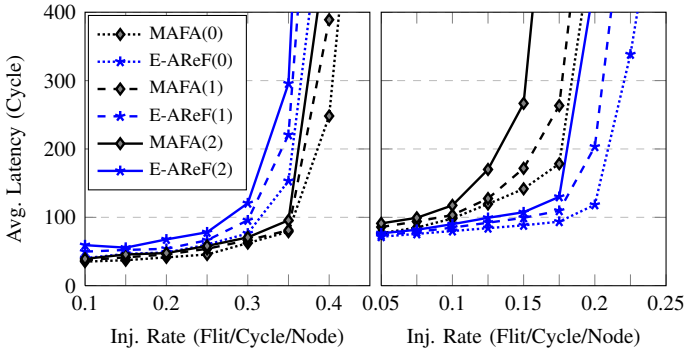

Fig. 10. Performance comparison of an 8×8 mesh under uniform (left) and hotspot (right) traffic profiles.

3) Trace-driven Traffic: For a more realistic traffic analysis, we carried out trace-driven simulations from SPLASH-2 benchmarks across an 8×8 NoC configured with 20 processors and 44 shared L2 cache memory modules. Figure 11 shows the average packet latency across five benchmarks, normalized to MAFA. E-AReF provides lower latency than MAFA and it shows the greatest performance gain for *cholesky* with 22% reduction in latency. The average performance gain of E-AReF across all benchmarks is up to 16%.

B. Reliability Analysis

In order to assess the reliability of E-AReF, the number of faulty links was increased from 1 to 10. The faulty links were selected using a random function with a distinct starting seed to ensure a fair comparison. The reliability is measured as [3]:

$$\frac{\textit{No. of successful packet arrivals at the destination nodes}}{\textit{Total no. of delivered packets}}$$

As can be seen in Fig. 12, both methods are 100% reliable when there is a single defective link in the network. Moreover, E-AReF can maintain a comparable reliability as that of MAFA despite refusing to use VCs. Both methods can tolerate up to ten broken links by more than 80% reliability.

C. Area Analysis

To evaluate the hardware cost of the proposed method, the routers were modeled with VHDL and synthesized by

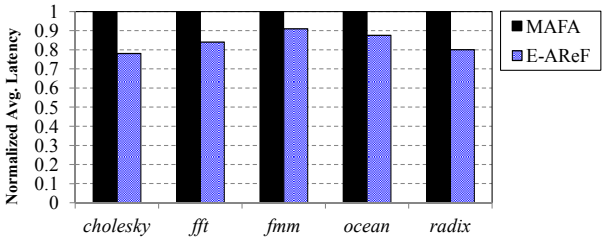

Fig. 11. Performance for application traces.

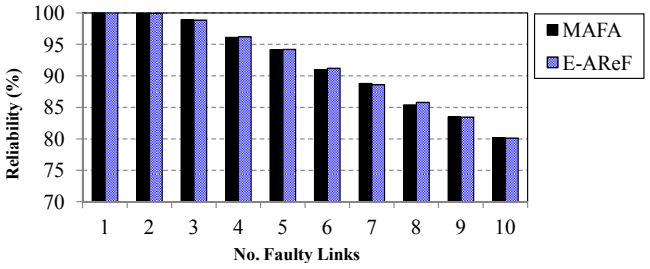

Fig. 12. Reliability evaluation in an 8×8 mesh under uniform traffic profile.

TABLE IV
HARDWARE IMPLEMENTATION DETAILS (μm^2)

Router	FIFO	Xbar-Arb.	Rout.	BIST	Reconf.	Total
Baseline	11,568	1,667	435	–	–	13,670
MAFA	11,620	4,857	1,163	1,142	–	18,782
E-AReF	11,568	1,756	1,436	1,142	1,440	17,342

Synopsys Design Compiler using the TSMC 65nm standard cell library. The E-AReF router was implemented using the universal Logic-Based Distributed Routing (uLBDR) proposed in [22]. The Reconfiguration Unit in E-AReF does not add delay to the critical path of uLBDR (similar to AbTM [7]). Hence, the time efficiency of uLBDR is inherited by E-AReF. We have considered the primary components of area overhead in our model, including the input buffers, crossbar switch and arbiter, routing unit, BIST and the fault information distribution mechanism, and reconfiguration unit. The overhead of each module for MAFA, E-AReF, and the baseline router without the fault tolerance capability is shown in Table IV.

The E-AReF router imposes around 26% area overhead compared with the baseline router which is mainly associated with the routing and reconfiguration technique to tolerate link failures with a graceful performance degradation. Assuming that the router area occupies 11% area of a tile as reported by Intel [7], the increase in the tile area using E-AReF will be less than 2.9% which can be considered negligible. As can be predicted, the MAFA router imposes around 8.3% area overhead in comparison with E-AReF. The area penalty of MAFA is mostly due to the VC mechanism which increases the complexity of the crossbar and arbiter.

VI. CONCLUSION

In this paper, a reconfigurable and fault-tolerant routing method (E-AReF) is presented for wormhole-switched 2D mesh NoCs. The main advantages of E-AReF over the conventional methods can be summarized as: (1) Dynamic reconfiguration of the algorithm according to the location of faults and congestion in the network to tolerate link failures with a high reliability; (2) E-AReF strives to minimize rerouting; (3) E-AReF can route the packets adaptively, even in the vicinity of faults; (4) To the best of our knowledge, it is the first routing algorithms that combats link failures and ensures deadlock-freedom without adding VCs or routing tables.

REFERENCES

[1] D. Fick, A. DeOrio, G. Chen, V. Bertacco, D. Sylvester, and D. Blaauw, "A highly resilient routing algorithm for fault-tolerant NoCs," in *Proc. ACM/IEEE Design Automat. Test in Europe (DATE)*, 2009, pp. 21–26.

[2] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. San Francisco, USA: Morgan Kaufmann Publishers, 2004.

[3] M. Ebrahimi, M. Daneshtalab, J. Plosila, and H. Tenhunen, "Minimal-path fault-tolerant approach using connection-retaining structure in Networks-on-Chip," in *Proc. ACM/IEEE Int. Symp. Networks-on-Chip (NOCS)*, 2013, pp. 1–8.

[4] A. Prodromou, A. Panteli, C. Nicopoulos, and Y. Sazeides, "NoCAlert: An on-line and real-time fault detection mechanism for Network-on-Chip architectures," in *Proc. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, 2012, pp. 60–71.

[5] C. J. Glass and L. M. Ni, "Maximally fully adaptive routing in 2D meshes," MSU-CPS-ACS-51, Michigan State Univ., Tech. Rep., 1992.

[6] P. Bahrebar and D. Stroobandt, "Adaptive and reconfigurable fault-tolerant routing method for 2D Networks-on-Chip," in *Proc. IEEE Int. Conf. ReConFig*, 2014, pp. 1–8.

[7] B. Fu, Y. Han, J. Ma, H. Li, and X. Li, "An abacus turn model for time/space-efficient reconfigurable routing," in *Proc. Int. Symp. Comput. Arch. (ISCA)*, 2011, pp. 259–270.

[8] A. DeOrio, D. Fick, V. Bertacco, D. Sylvester, D. Blaauw, J. Hu, and G. Chen, "A reliable routing architecture and algorithm for NoCs," *IEEE Trans. Comp.-Aided Design Integr. Circ. Syst.*, vol. 31, no. 5, pp. 726–739, 2012.

[9] P. Ren, Q. Meng, X. Ren, and N. Zheng, "Fault-tolerant routing for on-chip network without using virtual channels," in *Proc. ACM/IEEE Design Automat. Conf. (DAC)*, 2014, pp. 1–6.

[10] R. Parikh and V. Bertacco, "uDIREC: Unified diagnosis and reconfiguration for frugal bypass of NoC faults," in *Proc. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, 2013, pp. 148–159.

[11] K. Aisopos, A. DeOrio, L.-S. Peh, and V. Bertacco, "ARIADNE: Agnostic reconfiguration in a disconnected network environment," in *Proc. Conf. Parallel Arch. Compil. Tech. (PACT)*, 2011, pp. 298–309.

[12] M. Ebrahimi, M. Daneshtalab, J. Plosila, and H. Tenhunen, "MAFA: Adaptive fault-tolerant routing algorithm for Networks-on-Chip," in *Proc. Euromicro Conf. Digit. Syst. Design (DSD): Arch., Methods, Tools*, 2012, pp. 201–207.

[13] Z. Zhang, A. Greiner, and S. Taktak, "A reconfigurable routing algorithm for a fault-tolerant 2D-mesh Network-on-Chip," in *Proc. ACM/IEEE Design Automat. Conf. (DAC)*, 2008, pp. 441–446.

[14] D. Lee, R. Parikh, and V. Bertacco, "Highly fault-tolerant NoC routing with application-aware congestion management," in *Proc. ACM/IEEE Int. Symp. Networks-on-Chip (NOCS)*, 2015, pp. 1–8.

[15] C. Iordanou, V. Soteriou, and K. Aisopos, "Hermes: Architecting a top-performing fault-tolerant routing algorithm for Networks-on-Chips," in *Proc. IEEE Int. Conf. Comp. Design (ICCD)*, 2014, pp. 424–431.

[16] W.-C. Tsai, D.-Y. Zheng, S.-J. Chen, and Y.-H. Hu, "A fault-tolerant NoC scheme using bidirectional channel," in *Proc. ACM/IEEE Design Automat. Conf. (DAC)*, 2011, pp. 918–923.

[17] M. Radetzki, C. Feng, X. Zhao, and A. Jantsch, "Methods for fault tolerance in Networks-on-Chip," *ACM Comput. Surv.*, vol. 46, no. 1, pp. 8:1–8:38, 2013.

[18] I. Seitanidis, A. Psarras, E. Kalligeros, C. Nicopoulos, and G. Dimitrakopoulos, "ElastiNoC: A self-testable distributed VC-based Network-on-Chip architecture," in *Proc. ACM/IEEE Int. Symp. Networks-on-Chip (NOCS)*, 2014, pp. 135–142.

[19] J. Heisswolf, A. Weichslgartner, A. Zaib, S. Friederich, L. Masing, C. Stein, M. Duden, R. Klopfer, J. Teich, T. Wild, A. Herkersdorf, and J. Becker, "Fault-tolerant communication in invasive Networks on Chip," in *Proc. NASA/ESA Conf. Adapt. Hardware Syst. (AHS)*, 2015, pp. 1–8.

[20] S. Werner, J. Navaridas, and M. Luján, "A survey on design approaches to circumvent permanent faults in Networks-on-Chip," *ACM Comput. Surv.*, vol. 48, no. 4, pp. 59:1–59:36, 2016.

[21] G.-M. Chiu, "The odd-even turn model for adaptive routing," *IEEE Trans. Parallel Distrib. Syst.*, vol. 11, no. 7, pp. 729–738, 2000.

[22] S. Rodrigo, J. Flich, A. Roca, S. Medardoni, D. Bertozzi, J. Camacho, F. Silla, and J. Duato, "Cost-efficient on-chip routing implementations for CMP and MPSoC systems," *IEEE Trans. Comp.-Aided Design Integr. Circ. Syst.*, vol. 30, no. 4, pp. 534–547, 2011.