Addressing a New Class of Reliability Threats in 3-Dimensional Network-on-Chips

Ebadollah Taheri, Mihailo Isakov, Ahmad Patooghy and Michel A Kinsy, *Member, IEEE*

***Abstract*—Network-on-Chips (NoCs) are vulnerable to transient and permanent faults caused by thermal violations, aging effects, component wear out, or even transient fault sources. Although some of these faults are addressed by previous research, we show that there are reliability threats in 3D NoCs that go beyond the reliability issues investigated in 2D interconnect networks. First, we highlight one such class of reliability threats and discuss their manifestations in 3D NoCs. Second, we propose a thermal, reliability and performance-aware routing algorithm to tackle (i) previously established fault models and (ii) the new highlighted class of reliability threats in partially connected 3D NoCs. The proposed routing algorithm takes into account the states of routers and both the horizontal and Through Silicon Via (TSV) links, along with the temperatures of routers and cores. It then routes the packets around failed or overheated links and routers, achieving lower latencies by avoiding misrouting. To achieve this, the proposed routing algorithm uses the concept of vertical link announcement to inform nodes in the network of the working condition of vertical links. We evaluate the proposed routing algorithm under a wide range of working conditions using the Access Noxim NoC simulator. Results show that the proposed routing algorithm (i) is able to tolerate almost any number and pattern of vertical link failures, (ii) is reliable against the newly identified reliability threats, and (iii) improves the latency and temperature distribution of the network compared to previously proposed routing algorithms.**

***Index Terms*—Three-dimensional Integrated Circuit, Network-on-Chip, Reliability, Virtual Channel, Router Architecture.**

I. Introduction

With recent advances in VLSI technology, fabrication of stacked 3D and 2.5D ICs has become a reality [1], [2]. This development has prompted architects to explore 3D Network-on-Chips (NoCs) as a design approach to mitigate some of the scalability shortcomings seen in 2D NoCs. The scalability problem of 2D NoCs is based on the fact that their average internode distance grows rapidly with the number of on-chip cores. As a result, 2D NoCs impose a high average internode distance, network delay, and network power consumption [2]. By stacking planes of cores and using 3D NoCs as their communication fabric, the network diameter, network delay, and network power consumption are reduced while maintaining the same number of cores [3]. 3D NoCs utilize Through Silicon Via (TSV) links to connect different planes or layers to each other [4], [5]. Most of the fabricated 3D NoCs are only partially connected in the vertical dimension to afford the high fabrication cost of TSV vertical links [5].

In addition to inheriting the reliability challenges of 2D NoCs, 3D NoCs must deal with new reliability challenges mainly due to the extra steps needed in the fabrication of stacked layers and fabrication of TSV links. These challenges reduce the reliability of vertical links and that of the whole chip as well [3], [6], [7]. For instance, 3D IC fabrication technology increases the power density of modern chips which consequently arises thermal problems for 3D NoCs. The high core temperature not only decreases the lifetime and the mean time to failure of chips, but also results in low reliability and high cooling costs [8], [9]. In order to compensate for the reliability difficulties of 3D chips, incorporation of fault tolerant techniques in the design of 3D NoCs seems mandatory [10]. Major reliability challenges in 3D NoCs are (i) permanent horizontal/vertical link failures, (ii) transient thermal violations, and (iii) crosstalk and soft errors [11].

Routing algorithms which are responsible for sending and receiving packetized data through 3D NoCs are good candidates in solving reliability challenges of 3D NoCs. Reliability and thermal aware routing algorithms [9], [12] are widely utilized to evenly distribute heat generation of routers and cores over the chip area. However, some of the previously proposed routing algorithms for partially connected 3D NoCs suffer from serious performance and power problems. This happens due to either (i) low chance of finding minimal paths for packets, or (ii) extra hardware required to prevent deadlock situations. The usage of non-minimal paths in these routing algorithms greatly affects the performance and power consumption of the network and the chip as a whole [13].

The contribution of this paper is twofold. First, we introduce two new reliability threats that occur in partially connected 3D NoCs. The identified reliability threats have not been addressed in previous research. Our simulations confirm that the proposed reliability threats may affect a notable percentage of packets in the network. Second, we propose a fault tolerant and thermal aware routing algorithm. The proposed routing algorithm uses a TSV index sharing mechanism and efficiently tolerates previously addressed faults, our newly identified reliability threats, and thermal issues of 3D NoCs.

The rest of the paper is organized as follows. In Section II, related routing algorithms are reviewed. In Section III, we introduce new reliability threats which have not yet been addressed by other researchers. In Section IV, the proposed routing algorithm is discussed in detail. Section V gives an analytical model of the proposed algorithm. In Section VI, the simulation results are presented and discussed. Finally, Section VII concludes the paper.

II. Related Work

Complicated fabrication process of 3D ICs results in various permanent and transient faults which may alter correct

E. Taheri, M. Isakov and M. A. Kinsy are with the Adaptive and Secure Computing Systems (ASCS) Laboratory, Department of Electrical and Computer Engineering, Boston University, Boston MA 02215. E-mail: etaheri@bu.edu

A. Patooghy is with the Department of Computer Science at the University of Central Arkansas, Conway, AR 72035. E-mail: apatooghy@uca.edu

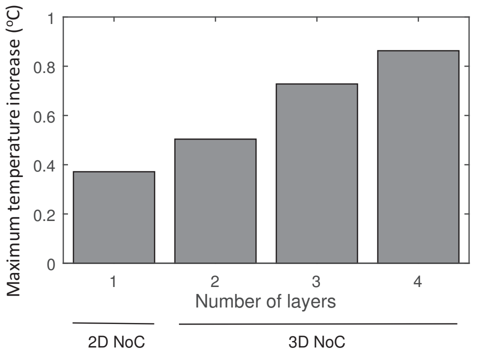


Fig. 1. Maximum temperature increase of nodes for 100K cycles of simulation.

functionality of 3D NoCs [1], [14], [15]. Several papers in the literature have attempted to cover the most important faults of 3D NoCs. Major faults addressed by researchers are as follows: (i) Horizontal link faults [16], [17], [18], this fault model assumes a permanent failure in a horizontal link of an NoC. Such a failure may happen due to fabrication defects, component wear out, high chip temperature, etc. (ii) Vertical link fault [4], [10], [19] considers a permanent failure of a TSV link due to fabrication defects, temperature violations, etc. (iii) Router failure, this model presumes that an on-chip router and all of its connected links have permanently failed [20], [21]. In addition to permanent faults, there is a class of transient fault models including, (i) soft error fault [22], which assumes an unwanted bit-flip in data stored in memory elements (virtual channel buffers) of a NoC router which happens when a high energy particle strikes the chip; and (ii) crosstalk fault model [23] covering errors caused by unwanted wire couplings of adjacent lines of NoC links. Permanent and transient faults may alter the correct functionality as well as the performance of NoCs. Some researchers have tried to estimate the performance loss associated to faults in NoCs [24], [25]. However, the class of permanent faults is considered more serious [11] since permanent faults can affect a much larger portion of packets.

Thermal violations are commonly classified as one of the major sources of permanent faults in NoC components, especially in 3D NoCs. Thermal violations occur due to the 3D NoCs layers which are farther from the heat-sink having a lower cooling efficiency. Thus, parts of the chip may overheat and break down which results in permanent (or at least long lasting) failures. Due to the high impact of thermal violations in 3D NoCs, some researchers [9], [12] have considered them as a separate fault model. To show the severity of thermal issues in 3D NoCs with respect to 2D ones, we compare the thermal behaviors of these interconnect network topologies. In Figure 1, we present the results of the thermal experiments conducted on 2D and 3D NoCs. Simulations are repeated 10 times and the highest temperature increase of routers in all layers is logged and reported. Although simulations are done for only a very short period, i.e., 100K cycles, the results corroborate the fact that 3D NoCs do exhibit severe thermal problems, especially in the layers farther from the heat-sink.

Over the last 15 years, researchers and designers have used various routing techniques to improve load-balancing, boost performance, and mitigate thermal/power issues in NoCs [26], [27], [28], [29], [30], [31]. Similarly, some researchers have proposed fault tolerant routing algorithms [4], [5], [10], [32], [18] to deal with aforementioned fault models in 3D NoCs. Such routings mainly try to find alternative paths and bypass failed components to deliver packets to their destination nodes [33]. In fact, the inherent path redundancy of NoCs has been used by fault tolerant routing algorithms to improve reliability of NoCs. AFRA [10] is a fault tolerant routing algorithm for mesh based 3D NoCs with fully connected vertical links. The routing algorithm bypasses failures on vertical links by misrouting packets to other areas of the network. This routing algorithm can tolerate vertical link faults in only one direction i.e, the routing cannot tolerate failures in both upward and downward vertical links. This is done to achieve deadlock-freedom without the need for an extra virtual channel. Ebrahimi et al. in [32] have proposed a fault tolerant routing algorithm based on the Hamiltonian path strategy. This routing algorithm is deadlock-free because each node is visited only once in the formed Hamiltonian path. The algorithm is non-minimal, power intensive, and degrades the NoC performance. The TAAR routing algorithm [9] reactively changes the default routing paths to reduce the traffic load of high temperature NoC routers. Traffic load reduction allows such routers to cool down, making them available for future operations. Emergency routers in this technique are temporarily turned off and the routing algorithm stops packets from crossing such routers.

To minimize fabrication cost and complexity, 3D NoCs are often designed with only partial connectivity in the vertical dimension [34]. Therefore, a judicious assignment of vertical links to packets traversing layers is needed to lessen router/vertical link overheating that may negatively impact the chip's reliability. Researchers have also used routing algorithms to alleviate the reliability issues associated with partially connected 3D NoCs [4], [34]. Jiang et al. [35] have proposed a deadlock-free routing algorithm using two virtual channels for vertically partially connected 3D NoCs. In this routing, each router needs to know the locations of all vertical links in the same layer, so that it can select the vertical link that offers the shortest path between source and destination nodes. This algorithm imposes extra hardware and time to perform the runtime computation for finding the best vertical link for each packet.

East-Then-West (ETW) [4] is another routing algorithm targeting partially connected 3D NoCs. The algorithm defines two sub-networks and uses them to route traffic packets in a deadlock-free fashion. In the algorithm, an adaptive, low hardware overhead, vertical link assignment scheme is used to route packets around failed vertical links. Using two virtual channels, Elevator-First routing algorithm [5] improves the network performance and simplifies the routing challenges of vertically partially connected 3D NoCs. However, the low degree of adaptivity in this routing algorithm results in low path diversity and in turn no TSV failure tolerance. Foroutan et al. in [13] have proposed an optimization technique for vertical link assignment of each source and destination pair for the Elevator-First routing algorithm. The optimized assignment of vertical links results in a better traffic distribution and even heat generation over the chip area. REDELF routing algorithm [36] adds some rules to Elevator-First routing in order to reduce hardware overhead and alleviate power consumption. While REDELF algorithm is able to work with one virtual channel, it cannot support any fault tolerance and mitigate reliability concerns.

Previously proposed algorithms for either fully or partially connected 3D NoCs use routers' local information to decide how to deal with local horizontal or vertical link failures. Limiting the network state information gathering radius to local routers keeps the network-updating traffic low, but oftentimes it also leads to non-optimal path selections. In this work, we introduce a new approach that expands the information gathering radius needed by routers to make near-optimal fault-tolerant routing decisions while keeping the net impact minimal in terms of router and routing complexities.

III. Proposed reliability threats for 3D NoCs

Previous works published in the literature [4], [19], [37], [17] addressed one or more of the following fault models (i) horizontal link fault, (ii) router fault, (iii) TSV link fault. These methods tried to tackle impacts of such fault models on the reliability of the NoC. They tried to temporarily or permanently bypass the faulty link or router to deliver packets to destinations.

In this section, we introduce two new reliability threats that may appear in 3D NoCs with partial TSVs. These reliability threats are derived from the observation that failed horizontal links significantly limit the path diversity available to packets, i.e., a packet may bypass a failed horizontal link by the aid of existing fault tolerant routings, but it may later fail to reach its destination or TSV link due to the lack of allowed turns. Bypassing failed horizontal links also imposes some unintended turn restrictions on the packets, and the packets may have to change virtual channel to prevent deadlock. Occasionally, some packets must be dropped to make forward progress in the network. Based on this observation, we define the following faults for 3D NoCs, which cover both vertical and horizontal links.

Reliability Threat 1 - Elevator Unreachable: In partially connected 3D NoCs, packets have to use elevator nodes (routers which have working TSV links) to reach their destination layer. Horizontal link failures in the source layer force packets to perform virtual channel changes and therefore will limit packets' allowed turns in the source layer. In this case a packet would be dropped since the packet is not able to reach the intended elevator node even though the elevator is working correctly. Even when the routing algorithm tolerates horizontal link failures, the Reliability Threat 1 may still manifest itself in the form of packet livelock. In fact, Reliability Threat 1 happens primarily due to route restrictions on packets with destination nodes located in other layers. In some cases, these packets may have already bypassed some of the failed horizontal links in the network.

Reliability Threat 2 - Destination Unreachable: In this case, the packet has passed the horizontal failed link(s), as well as its TSV link, but the packet has no more routing options at its destination layer. If the packet reaches a faulty horizontal link at the destination layer it cannot make an allowed turn to bypass the failed link. This results in either a packet drop or packet livelock depending on how the routing algorithm deals with horizontal link failures. In other words, how a fault tolerant routing algorithm uses allowed turns and virtual channel changes in the source layer may affect its fault tolerance capabilities in the destination layer. Reliability Threat 2 may either drop the packet or it may lead to a livelock scenario.

To show validity of the above mentioned reliability threats, we have done a set of experiments to investigate how these may affect packet propagation/delivery during the normal workings of an NoC. We used the Access Noxim NoC simulator to simulate $7 \times 7 \times 3$ and $5 \times 5 \times 3$ networks when up to 10% of all horizontal links have failed during network run-time. We counted the number of packets which have not been delivered to their destinations because of the defined fault models. We have repeated the experiments for three different routing algorithms, namely the Advertiser Elevator (AE) [34], ETW [4], and Elevator-First [5]. We added horizontal link failure tolerance to all three simulated routing algorithms such that the algorithms used their in-layer routing adaptivity to bypass failed horizontal links.

Figures 2.a and 2.b show the impacts of 1 to 30 horizontal link failures on the percentage of dropped packets due to Reliability Threat 1 and 2 in a $7 \times 7 \times 3$ network. Simulations confirm that even a very low number of horizontal link failures e.g., only 1 horizontal link failure, may result in having packet loss due to Reliability Threat 2 (Figure 2.b). Although routing algorithms support fault tolerance for horizontal links, we can still see some packets that fall into the TSV unreachable case (Figure 2.a). We repeated our simulations for a $5 \times 5 \times 3$ network and results are shown in figures 2.c and 2.d. It can be seen that when network size shrinks, the possibility of packet loss due to our proposed reliability threats gets worse. We conclude that our introduced reliability threats are real and do appear in 3D NoCs given the network is equipped to tolerate horizontal link faults. So, we should individually consider these new threats and include them in the design of fault tolerant NoCs.

IV. Proposed Fault Tolerant Routing Algorithm

In a partially connected 3D NoC, several packets should be delivered to other layers of the network while their generating nodes do not have a TSV link to forward the packets vertically. Such packets have to use available routers in the same layer which have a working TSV link. In the proposed routing algorithm, we call routers with a working TSV link *Healthy Elevators*. Packets have to choose the closest healthy elevator with the hope to shorten their path and improve network performance. To do this, in our proposed routing algorithm, every healthy elevator shares an alive signal called *index*. Sending indexes to neighboring routers will inform them that a working elevator can be found nearby. Every working router in turn stores $(the\, maximum\, index - 1)$ and its corresponding

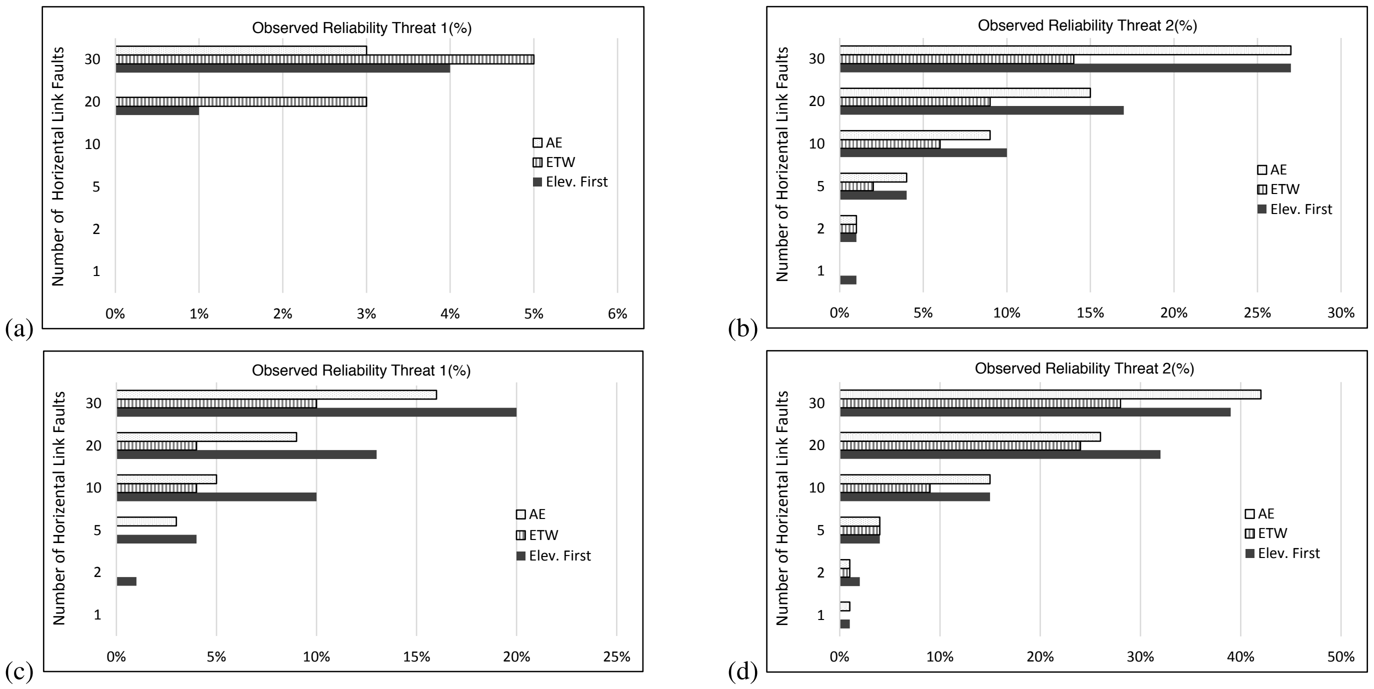


Fig. 2. Occurrence rate of proposed reliability threats in (a), (b) $7 \times 7 \times 3$ and (c), (d) $5 \times 5 \times 3$ networks versus number of horizontal link fault injections.

direction for future use. Also, the router will share this value in the next cycle. This way, 1) packets have more adaptivity in selecting elevator links when there is more than one healthy elevator nearby with the same hop count. This results in a better traffic balance in the network, 2) network is able to tackle permanent and/or transient TSV faults, and 3) a better temperature distribution over the network can be reached (see Section VI). The proposed routing algorithm helps to mitigate the effect of faults in the following scenarios:

- Packet loss due to TSV (vertical) link permanent failures: when a TSV link fails due to the aging effects or due to reaching a very high temperature it fails permanently. We name this fault model as F^P_{TSV}.
- Packet loss due to TSV links transient failures: a TSV link may be temporarily out of service since the temperature of its router is above the allowed threshold. We name this fault model as F^T_{TSV}.
- Packet loss due to horizontal link failure: aging effects, bridging faults, stuck open and stuck shorts, and electromagnetic interference may prevent horizontal links from working properly. We denote this as F_{HL}.
- Packet loss/livelock in the destination layer due to one (or more) horizontal link failures in the source layer (discussed in Section III). We name this by F^D_{DL}.
- Packet loss/livelock in the source layer due to one (or more) horizontal link failures in the same layer (discussed in Section III). We denote this by F^S_{DL}.

The index sharing mechanism helps us to tackle faults F^P_{TSV} and F^T_{TSV}. In addition, the ability of fully adaptive routing of packets at both source and destination layers maximizes the path diversity and in turn tackles faults F_{HL}, F^D_{DL}, and F^S_{DL}. In Section IV-A, we define how healthy elevators announce themselves using the index sharing mechanism. The proposed

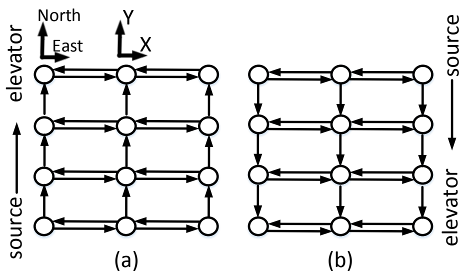


Fig. 3. Virtual networks in the XY plane, (a) northward virtual network for packets searching for a north elevator (b) southward virtual network for packets searching for a south elevator.

fully adaptive routing of packets is described in Section IV-B.

A. Adaptive Selection of Elevator Links

The proposed routing algorithm defines two virtual networks (shown in Figure 3) to transfer packets in each layer of the network. If the packet's destination is in the same layer as its source node (called **intra-layer packet**), based on where the destination is located, one of the virtual networks will be adopted to route the packet. If the packet's destination is located in a different layer (called **inter-layer packet**), the routing delivers it to the nearest healthy elevator, then the packet passes through the elevator link and reaches the destination layer; then the destination node. The northward virtual network, Figure 3.a, is used to send inter/intra-layer packets which need to reach an elevator/destination node located at the north side of their sources. In contrast, the southward virtual network, Figure 3.b, is used to route inter/intra-layer packets which need to reach an elevator/destination node located at the south side of their sources.

1) Index Sharing Algorithm: Algorithm 1 shows the details of the index sharing and index update mechanisms in elevator and non-elevator routers of the network. At the start of the algorithm, healthy up/down elevators share their *Initial Index Value* in the southward and northward virtual networks. A non-elevator router would have four different indexes S_U, N_U, S_D, N_D representing the indexes for the up-elevator in southward virtual network, up-elevator in northward virtual network, down-elevator in southward virtual network, and down-elevator in northward virtual network, respectively. These indexes are stored as part of the routing table and used to route packets. When a router receives an index update, it compares it with stored value and if needed it updates its copy before forwarding it to its neighboring routers.

As an example, Figure 4.a shows northward index sharing in the East, West, and South directions with an *Initial Index Value* (here we assumed 4) to guide northward inter-layer packets. Note that a northward/southward index in fact propagates in the opposite direction i.e., southward/northward. The healthy elevator starts sharing an index of 4 and each router decreases the received index by one and then re-shares the index. Figure 4.b shows a case in which there are two healthy elevators. Based on this example, the circled node receives two indexes from its north and west neighbors. In such situations, the router stores the $(maximum\, received\, index - 1)$ for future packet routings in the northward virtual network.

The same index sharing mechanism is used for the southward virtual network i.e, indexes are shared in the East, West, and North directions starting from a healthy elevator with the initial index value. Finally, the router would have two pairs of $P_1 = (N_U,\, S_U)$ and $P_2 = (N_D,\, S_D)$ values to find the closest up/down elevator in each virtual network.

Figure 5 shows an example of elevator index sharing process for a layer of a 3D NoC with five healthy elevators. For the sake of simplicity in the example, we ignored TSV directions i.e., we assumed that all TSVs are bidirectional. The way NoC routers use these indexes is as follows: source routers check the southward/northward indexes to decide the closest elevator for an inter-layer packet. The greater index value decides the lower distance to a healthy elevator, so the source router will select its corresponding virtual network. As shown in Figure 5.b, source S_1 would select the southward virtual network for its packets since its index is greater $(index_{north} = 0 < index_{south} = 3)$. The source S_1 decides to route its packets in the southward virtual network and uses its south port since this is the port which announced the greater elevator index. Source S_2 selects the northward virtual network for its packets since its northward elevator index is greater than the southward index $(index_{north} = 2 > index_{south} = 1)$.

Source S_3 has two choices because the northward and southward elevator indexes are equal i.e., $(index_{north} = index_{south} = 1)$. This feature gives a degree of adaptivity to the proposed selection method and the whole network in turn. In such situations, the proposed routing algorithm selects a path according to the destination of the packet. The South/North path is selected if the destination is south/north from the source, respectively. All minimal paths to the nearest elevators may be used in the proposed mechanism (see the paths for source 3 and source 4).

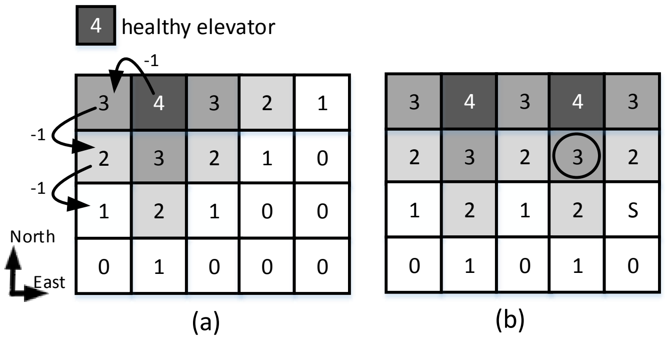


Fig. 4. Elevator index sharing in the northward virtual network for (a) one and (b) two healthy elevators assuming an initial index value of 4.

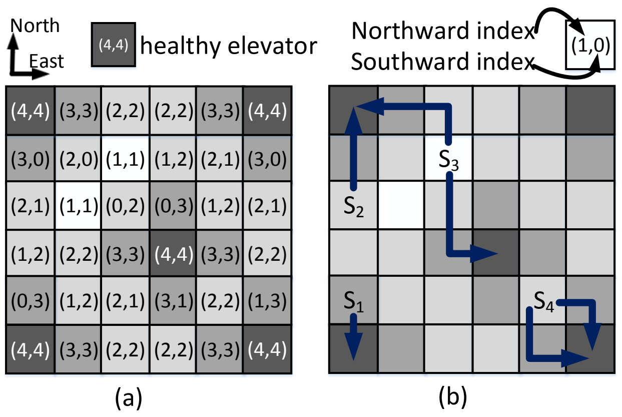


Fig. 5. (a) Elevator indexes of each node based on the received indexes, (b) path selection in a 6×6 node layer of a 3D NoC.

The learning rate of healthy elevator routers in the network is closely related to the initial index value, since the initial value defines the index's life radius. For example, if the initial value is set to 4, then healthy elevator routers will be announced to nodes that are at most 4 hops away. The initial index value needs to be selected effectively. A small initial index value will limit the index's life radius and may prevent its propagation to some parts of the network. Equally, if the initial index value is too large, it will increase the index sharing related traffic in the network while providing no performance benefit, since the routers only store the greatest indexes. In Section V we have an analytical discussion to determine what should the initial index value be to inform all nodes of the network. Obviously, it depends on the number and distribution of TSV links over the network.

2) Router Micro-Architecture: To support the proposed index sharing technique, the router logic is modified to enable the following functionalities. (i) A router knows whether it possesses a vertical elevator or not. (ii) A router is aware of the states of its elevators, i.e, working, thermally disabled or failed. To do this, the router is equipped with a thermal sensor and knows its current temperature [9], [12], [38]. (iii) Healthy elevators generate and share indexes, while other routers only share received indexes. A healthy elevator is a non-overheated router with a working TSV link. If the TSV fails or the router gets overheated, the router shares the index of 0 to announce that its elevator is out of service. The status of a

Algorithm 1 Index Sharing Algorithm

S_U, N_U: Local registers to store Southward/Northward indexes for an up elevator; // to an upper layer.
S_D, N_D: Local registers to store Southward/Northward indexes for a down elevator; // to a lower layer.

```
1: if (local router has a healthy Up elevator) then
2:     $S_U$ and $N_U \leftarrow Initial\ index\ value$;
3: end if
4: if (local router has a healthy Down elevator) then
5:     $S_D$ and $N_D \leftarrow Initial\ index\ value$;
6: end if
7: if (received $S_U >$ local $S_U$) then
8:     local $S_U \leftarrow$ received $S_U - 1$;
9: end if
10: if (received $S_D >$ local $S_D$) then
11:     local $S_D \leftarrow$ received $S_D - 1$;
12: end if
13: if (received $N_U >$ local $N_U$) then
14:     local $N_U \leftarrow$ received $N_U - 1$;
15: end if
16: if (received $N_D >$ local $N_D$) then
17:     local $N_D \leftarrow$ received $N_D - 1$;
18: end if
19: Send local indexes over all the working horizontal links to
    neighboring routers when there is a local index update;
```

non-working router can be revised to ‘functional’ again if its temperature falls below a targeted thermal threshold and it still has a working TSV link. (iv) Routers perform the index sharing on Req/Ack wires which are needed for the handshake mechanism. Without loss of generality, if a router design is based on a credit flow control architecture, then indexes can also be appended to credit flits.

Figure 6 shows the micro-architecture of the modified network router to support the proposed index-sharing mechanism. The *Index sharing unit* is responsible for (a) generating the initial index values for healthy elevators, (b) receiving indexes from neighboring routers and comparing them with its stored indexes, (c) updating local/stored indexes if necessary, and (d) sharing updated local indexes with neighboring routers at the same network layer. In the *Index sharing unit* we have the following components:

Index registers: to enable faster index comparisons, we use 12 index registers to store the indexes that a router may receive. Under this setting, a router can receive up to $2 \times 3 \times 2 = 12$ different indexes. The first term is associated with the upward/downward elevators, the second term is the number of different indexes in each virtual network, and the last term covers the two virtual networks. For example, register $East\, S_U$ stores the index received from the east port of the southward virtual network for upward elevators. The bit width of the register is log_2(initial index value).

Shift modules: since the *Index sharing unit* uses one wire to send and receive indexes, parallel-to-serial then serial-to-parallel operations are performed to transmit the indexes over the common wire.

Control unit: this module controls the state of *Index sharing unit*, it time-multiplexes request signals between adjacent routers and interprets them either as a request or an index.

Algorithm 2 Initial Virtual Channel Selection Algorithm

```
1: if (destination is in the same layer) /* Intra-layer packet */ then
2:     if (destination is northern of source) then
3:         return virtual channel 1;
4:     else
5:         return virtual channel 2;
6:     end if
7: else
8:     if (destination layer is upper than source) then
9:         if ($N_U > S_U$) /* Upward inter-layer packet */ then
10:            return virtual channel 1;
11:        else
12:            return virtual channel 2;
13:        end if
14:    end if
15:    if (destination layer is lower than source) then
16:        if ($N_D > S_D$) /* Downward inter-layer packet */ then
17:            return virtual channel 1;
18:        else
19:            return virtual channel 2;
20:        end if
21:    end if
22: end if
```

In the case where a router needs to send both request and credit bits at a same time, the control unit gives the priority to index bits.

Flow control/index wire: the proposed index sharing mechanism uses a single bit wire between routers. This “flow control/index” wire data is used by the receiving router to interpret the other two controlling wires as req/ack signals or index/index ack (Figure 6).

B. The Proposed Fault Tolerant Routing Algorithm

The proposed routing algorithm uses 3 virtual channels to offer a high degree of adaptivity and fault tolerance at the source and destination layers. The first and second virtual channels are used to define two disjoint virtual networks as shown in Figure 3. For an Intra-layer packet, a router chooses one of the virtual networks based on the location of its destination node. The packet then would be routed fully-adaptive to reach the destination. In Figure 7, pairs $S_1 \rightarrow D_1$ and $S_2 \rightarrow D_2$ are examples of Intra-layer packets.

Based on the stored elevator indexes, any local router chooses one of the virtual networks shown in Figure 3 to start forwarding an inter-layer packet. The packet is routed fully-adaptive and is able to bypass failed horizontal links in its journey to the elevator router. In fact, the router tries to send the packet toward the closest healthy elevator that it knows. The packet is not allowed to change its virtual network in order to prevent deadlocks. Inter-layer packets are routed using the first or the second virtual channel in the source layer. Switching to the third virtual channel takes place when an inter-layer packet leaves its source layer for a destination layer lower than the source. As shown in Figure 7, for the source/destination pair of $S_3 \rightarrow D_3$, the packet keeps using the first virtual channel in the destination layer. However, $S_4 \rightarrow D_4$ packet switches to the third virtual channel when it leaves its source layer, so that the packet can retain its routing adaptivity even in the middle layers. The pseudo code of the proposed routing is shown in

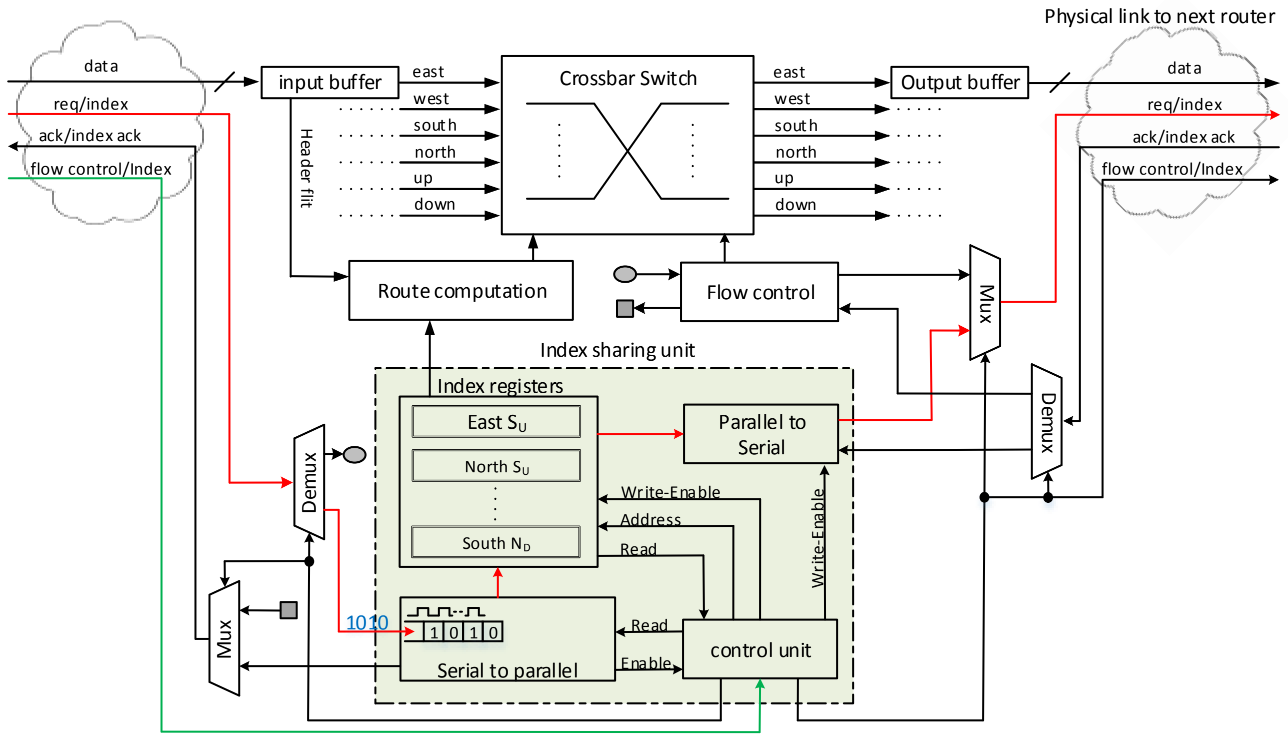


Fig. 6. Block diagram of the proposed router micro-architecture. In non-elevator routers, indexes flow through the red path. The green line performs "flow control/index" multiplexing.

Algorithm 3 which uses the virtual channel assignment policy as shown in Algorithm 2.

As shown in Figure 7 for $S_3 \rightarrow D_3$ packet, the proposed routing algorithm tolerates any permanent or transient TSV failures i.e., fault models F^P_{TSV} and F^T_{TSV}. This is achieved by the fact that the router of a failed elevator link stops sharing indexes. This in turn redirects inter-layer packets to other healthy nearby elevators. Since the process of sending an inter-layer packet to a healthy elevator is distributed, even those inter-layer packets that have left their routers before having the indexes updated, will be soon redirected to an alternative healthy elevator by other intermediate routers. It is worth mentioning that there are no restrictions on the number and pattern of faults F^P_{TSV} and F^T_{TSV} in the network. As long as there is a path in the packet's virtual network towards a healthy elevator in the layer, the proposed routing algorithm will find and use it.

In order to tolerate faults F_{HL}, our proposed routing algorithm tries to bypass failed horizontal links in the source and destinations layers using alternative paths. To deal with faults F^D_{DL} and F^S_{DL} as well, the proposed routing offers a set of allowed misroutes as shown in Figure 8. The proposed misroutes are defined based on the current direction and current virtual channel of the packet. A misroute will take place when the packet has no more adaptivity to bypass a failed horizontal link. To accomplish this, we investigate various possible misroute scenarios and implement mechanisms in the routing algorithm to support them. As shown in Section III, mis-routed packets may have to change VC, and we need to control their path traversal diversity to prevent deadlocks. Although the current framework allows only one misroute for a packet per each VC, it does not mean that a packet cannot bypass multiple failed links. Packets have adaptivity in both source and destination layers, consequently, they are able to bypass failed links without using their misroute options (cf., Figure 7 case $S_5 \rightarrow D_5$).

When a packet uses its misrouting option, it will be routed by dimension order routing at the destination layer. This may be X first (XY) or Y first (YX) routing depending on the misroute scenario experienced by the packet. If the packet encounters no misrouting, the routing algorithm utilizes the maximum possible adaptivity criterion to reach a better traffic balance. Figure 7 shows different examples of our misroute policy for packets $S_5 \rightarrow D_5$ and $S_6 \rightarrow D_6$ which do not have more adaptivity to bypass a horizontal failed links, such packets would use the misroute option and continue with a dimension order routing. However, in the same figure, packet $S_7 \rightarrow D_7$ and $S_8 \rightarrow D_8$ are able to bypass the failed horizontal links since their source routers have not received any index from south and north directions respectively.

C. Deadlock & Livelock Freedom

A livelock is a situation where packets are pursuing a destination but cannot find it. The condition occurs when non-minimal routing algorithms are employed in the network. In order to forbid livelock in the proposed routing algorithm, the 180° turns (the source of livelock) are prohibited.

As we explained, the proposed routing algorithm uses 3 virtual channels for each physical channel. It uses deadlock-free turn-prohibitions in XY, XZ, and YZ planes for the first, second, and third virtual channels as shown in Figure

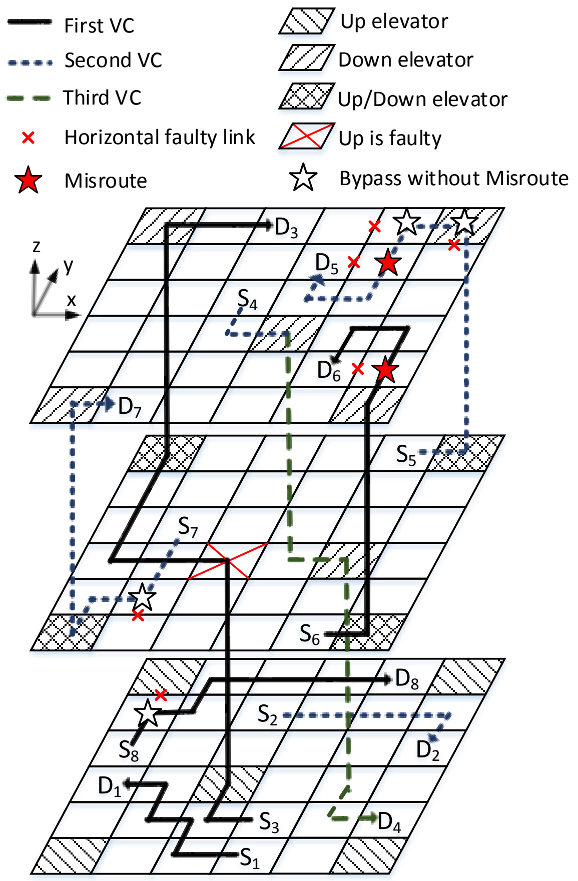


Fig. 7. Packet routing examples based on the proposed routing algorithm.

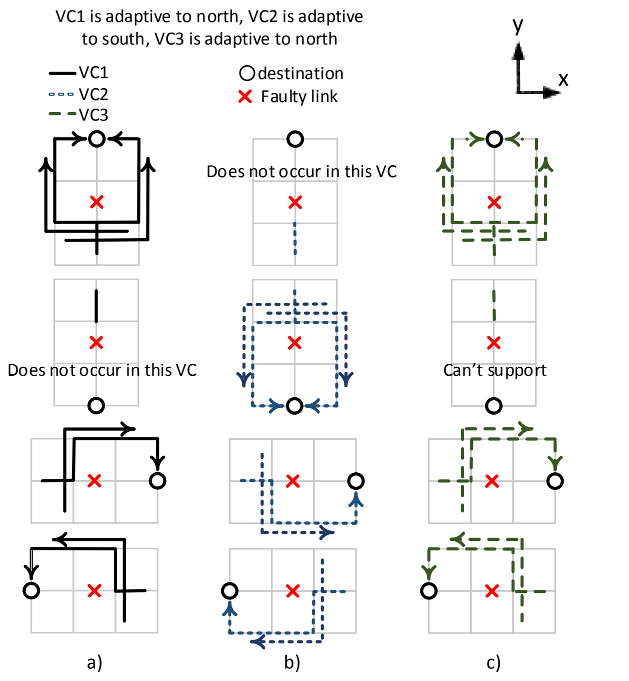

VC1 is adaptive to north, VC2 is adaptive to south, VC3 is adaptive to north
VC1
VC2
VC3
destination
Faulty link
y
x
Does not occur in this VC
Does not occur in this VC
Can't support
a)
b)
c)

Fig. 8. Packet misroute policies when a packet is facing a horizontal failed link to tackle faults F_{HL}, F_{DL}^{D}, and F_{DL}^{S}.

Algorithm 3 Proposed Routing Algorithm

```
1:  if (packet is inter-layer
    & it is not in the destination layer) then
2:      if (current router is a healthy elevator) then
3:          route the packet using elevator link;
4:      else
5:          route the packet adaptively to the nearest healthy elevator;
6:      end if
7:  end if
8:  if (packet is intra-layer
    or (inter-layer & currently in its destination layer)) then
9:      if (packet is in $VC_1$) then
10:         if (destination is northern than elevator) then
11:             minimal-route adaptively or mis-route using $VC_1$;
12:         else
13:             minimal-route adaptively or mis-route using $VC_2$;
14:         end if
15:     end if
16:     if (packet is in $VC_2$) then
17:         if (destination is southern than elevator) then
18:             minimal-route adaptively or mis-route using $VC_2$;
19:         else
20:             minimal-route adaptively or mis-route using $VC_3$;
21:         end if
22:     end if
23:     if (packet is in $VC_3$) then
24:         minimal-route par-adaptive or mis-route using $VC_3$;
25:     end if
26: end if
```

9. Based on this figure, the upward turns to XY plane (i.e., Up-North, Up-East, Up-South, and Up-West) are prohibited in the third virtual channel. Similarly, in the first and second virtual channels, the proposed routing algorithm prohibits the lateral-downward turns (i.e., North-Down, East-Down, South-Down, West-Down). In order to prevent cyclic dependencies between packets of different layers, packets are only allowed to change their virtual channels based on the following set of rules. 1) If the current virtual channel is of the first class, it can be changed to either the second or the third class of virtual channel. This virtual channel change may happen to use the maximum routing adaptivity of the second virtual network at destination layer. 2) If the current virtual channel is of the second class, it can be changed to the third class virtual channel to exploit maximum routing adaptivity in northward virtual network. No virtual channel change from third class, or from second to first class is allowed. The proposed set of turns and virtual channel change prohibitions results in deadlock freedom and no cyclic dependencies between the packets traversing the 3D NoC.

V. Analysis of the Initial Index Value

In this section we address the question "what is the minimal initial index value in the index sharing mechanism that covers all nodes of each plane?". Since indexes are sent over the Ack/Nack handshake signals between two adjacent routers, it can be realized using only two bits. If the initial index value is greater than 3, then multiple cycles can be used to share indexes between adjacent routers. In order to minimize (i) the number of required cycles and (ii) the overhead associated with a multi-cycle index sharing scheme, we examine approaches for determining the minimum initial index value that covers all nodes in the network. The following analytical discussion

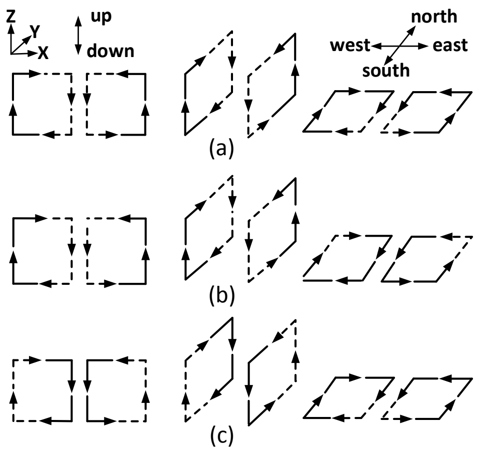


Fig. 9. Allowed turns of (a) first, (b) second, and (c) third virtual channels.

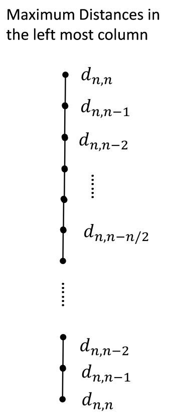


Fig. 10. Different maximum distances when a TSV is placed at the left most column of an $n \times n$ plane.

summarizes our efforts to estimate the minimum initial index value which should be shared by healthy elevators to cover all nodes of a plane. The model is based on the following assumptions.

- Each layer of our NoC is a $n \times n$ 2D mesh.
- We define a mesh radix in our $n \times n$ mesh as $Rad_1 = n$.
- We assume that m nodes in this network have elevators (TSVs links) to connect the other layers of the 3D $n \times n \times k$ NoC.
- We assume that TSVs are distributed uniformly over a layer.

We know that the network diameter in a $z \times p$ 2D mesh network, i.e., $d_{z,p} = (z-1)+(p-1)$. By placing the first TSV in our 2D plane, the maximum distance between the elevator node and other nodes of the plane would be:

- **Worst case:** the elevator node is placed on one of the 4 corner nodes in our $n \times n$ mesh, so the maximum distance would be the same as the diameter of the mesh, i.e., $d_{n,n} = 2 \times (n-1)$.
- **Best case:** the elevator node is placed in the center of the $n \times n$ mesh, so the maximum distance would be the diameter of $\lfloor \frac{n}{2} \rfloor \times \lfloor \frac{n}{2} \rfloor$ mesh, i.e., $d_{\lfloor \frac{n}{2} \rfloor, \lfloor \frac{n}{2} \rfloor}$.
- **Average case:** to calculate this, we initially assume that the TSV is placed somewhere on the most left column of the $n \times n$ mesh. We will later relax this assumption. Based on the place of the elevator in this column we have different maximum distances which are shown in Figure 10. We can say that the average maximum distance between all nodes of the $n \times n$ mesh and the elevator node, assuming that the elevator is located in the most left column of the mesh, ad_1, is as Equation 1.

$$n \times ad_1 = \begin{cases} X & \text{for even n} \\ X + d_{n,n-\lfloor \frac{n}{2} \rfloor} & \text{for odd n} \end{cases} \quad (1)$$

where $X = 2d_{n,n} + 2d_{n,n-1} + ... + 2d_{n,n-\lfloor \frac{n}{2} \rfloor+1}$. To generalize this formula to all margins of our $n \times n$ plane, i.e., left and right most columns, top and the bottom rows, we can calculate AD_1 as Equation 2. It should be noted that the corner nodes are common between a row and a column, but other nodes are not common.

$$n \times AD_1 = \begin{cases} 4 \times (Y) & \text{for even n} \\ 4 \times (Y + d_{n,n-\lfloor \frac{n}{2} \rfloor}) & \text{for odd n} \end{cases} \quad (2)$$

where $Y = d_{n,n} + 2d_{n,n-1} + ... + 2d_{n,n-\lfloor \frac{n}{2} \rfloor+1}$. To relax the assumption, suppose that the elevator node is located in the i^{th} column of the $n \times n$ mesh. The average maximum distance between the elevator node and all other nodes of the network regardless of its position can be calculated by Equation 3 whereat $Z = d_{n-i,n-i} + 2\sum_{j=1}^{\lfloor \frac{n-i}{2} \rfloor -1}$.

$$n^2 \times \overline{AD_1} = \begin{cases} \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} 4 \times (Z) & \text{for even n-i} \\ \sum_{i=0}^{\lfloor \frac{n}{2} \rfloor} 4 \times (Z + d_{n-i,\lfloor \frac{n-i}{2} \rfloor}) & \text{for odd n-i} \end{cases} \quad (3)$$

After placing the first TSV in the $n \times n$ plane, the average maximum distance can be calculated by above equation. Now, we want to consider impacts of placing the next TSV on the plane. To do this, let us first calculate the size of the largest sub-mesh after adding the first TSV. It would help us to calculate the average maximum distance after adding the second TSV. The largest sub-mesh can be approximated as a mesh of $Rad_2 \times Rad_2$ where Rad_2 can be computed by Equation 4

$$Rad_2 = \lceil \frac{\lceil \overline{AD_1} \rceil}{2} - 1 \rceil \quad (4)$$

We can now add the second TSV and calculate the AD_2 the same way as we have done for the first TSV. This time the parameter n, which is in fact our plane size, should be substituted by Rad_2. For considering the third TSV, if the third TSV is placed on the largest remaining sub-mesh then Rad_2 should be considered as the sub-mesh radix, otherwise Rad_2 would be fine. This iteration should be repeated m times to approximately calculate the maximum distance when m TSVs are placed in the network reaching to $\overline{AD_m}$. Finally, since the indexes should cover all nodes of the network to find

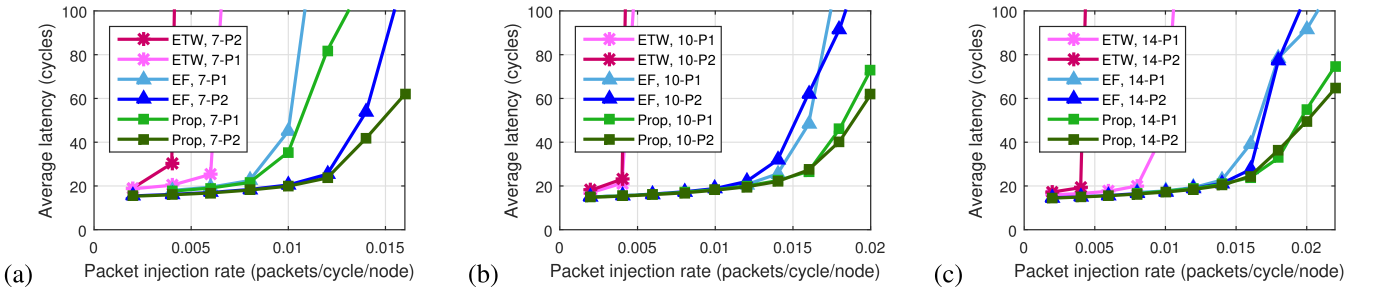
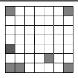
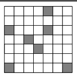
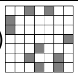
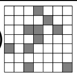
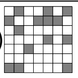
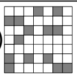


Fig. 11. Average latency for ETW, Elevator-First, and the proposed routing algorithm under the uniform traffic pattern with (a) 7 healthy elevators, (b) 10 healthy elevators, and (c) 14 healthy elevators using TSV placement patterns P1 and P2.

TABLE I
SIMULATION SETUP

Simulator	Access Noxim [39]
Network size	$7 \times 7 \times 3$
Fault & Traffic simulation	100000 cycles
Thermal simulation	4000000 cycles
Switching technique	Wormhole switching
Buffer depth	4 flits
Packet size	8 flits
Traffic pattern	Random uniform, Hotspot
7 TSVs placement patterns	7-P1)  7-P2) 
10 TSVs placement patterns	10-P1)  10-P2) 
14 TSVs placement patterns	14-P1)  14-P2) 

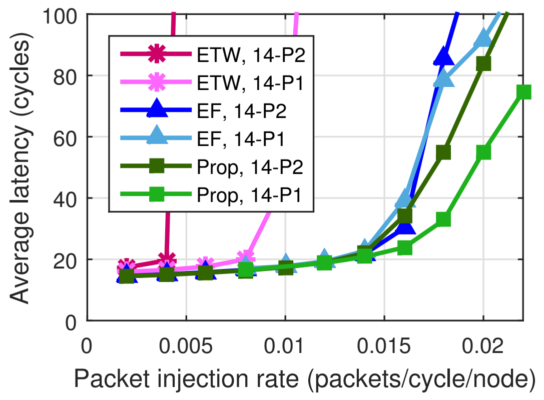


Fig. 12. Average network latency with 14 healthy elevators under the Hotspot traffic pattern.

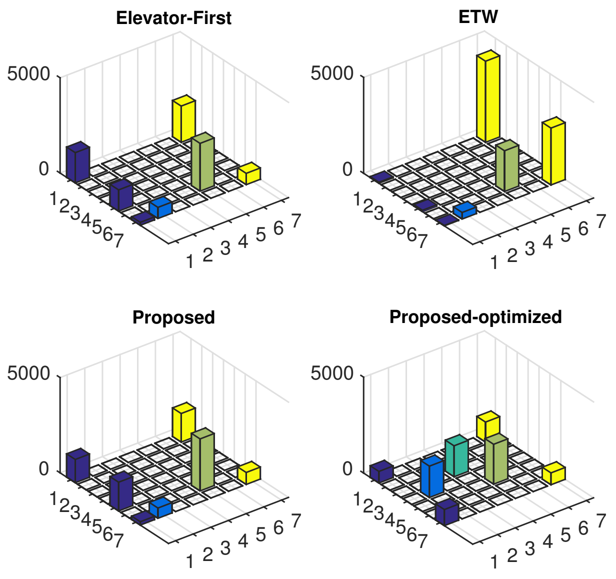


Fig. 13. Elevator utilization for ETW, Elevator-First, and the proposed routing algorithms in a 7 elevator network.

healthy elevators, we set the initial index value to be $2\overline{AD}_m$. To support this, we would need at least $Log_2 2\overline{AD}_m$ bits to have a safe index sharing.

VI. Experimental Evaluations

This section presents the simulation results of the proposed routing algorithm against ETW [4] and Elevator-First [5] routing algorithms in terms of average network delay, network thermal distribution, and fault tolerance. Evaluation data is gathered using the Access Noxim [39] NoC simulator when 7, 10, and 14 TSV links are placed in a $7 \times 7 \times 3$ 3D NoC. To measure the impact of the TSV placement on the routing algorithms' performance, we used two different TSV placement patterns in in the simulation experiments. Details of the simulation setup are summarized in Table I.

A. Performance Evaluation

Figures 11.a, 11.b, and 11.c show the average network delay of the proposed routing algorithm versus ETW [4] and Elevator-First [5] routing algorithms. Here we used the shortest elevator assignment for the Elevator-First routing algorithm since this algorithm must statically know the place of TSVs. As shown in Figures 11.a to 11.c, the proposed routing algorithm has a much better performance in all simulation conditions. The proposed routing algorithm's performance can be attributed to its dynamic TSV assignment using the index sharing protocol. In Figure 11.a where there are only 7 healthy elevators, all the routing algorithms show greater sensitivity to the TSV placement patterns. Further analysis using Matlab shows that the average router distances to the closest elevator

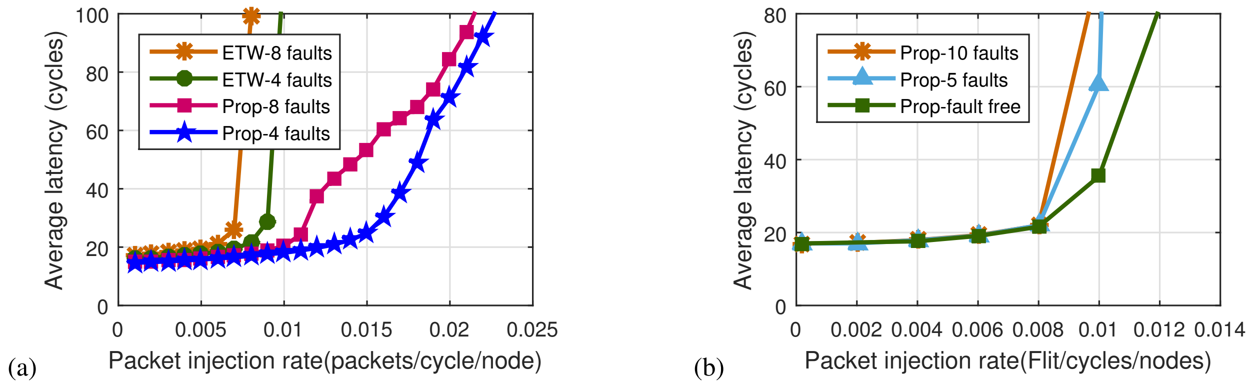


Fig. 14. Average network latency in the presence of (a) elevator failure, F^{P}_{TSV}, and (b) horizontally link failure F_{HL} injections.

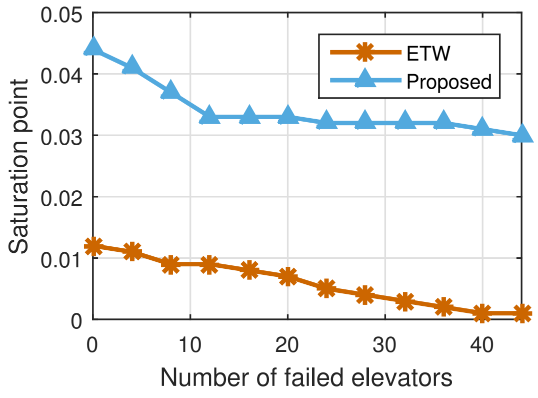


Fig. 15. Saturation point versus elevator failures in the proposed & ETW algorithms.

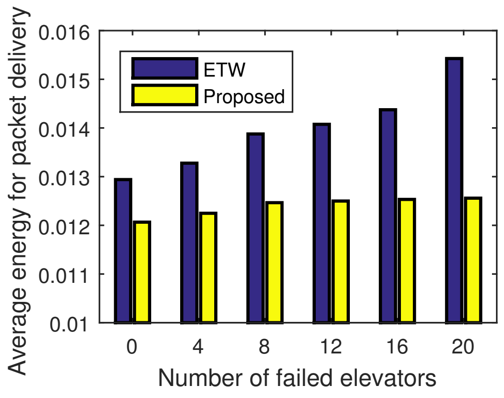


Fig. 16. Energy per packet versus number of failed elevators.

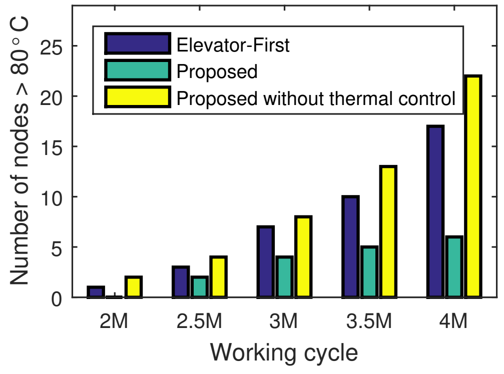


Fig. 17. Number of overheated nodes versus network working cycles.

are 2.02 and 1.61 hops for TSV placement of patterns P1 and P2, respectively. This 20% difference affects all three examined routing algorithms. With 10 and 14 healthy elevators the average distances to closest TSV for patterns P1 and P2 are less pronounced (cf. figures 11.b and 11.c). In these two settings, the proposed algorithm shows the least sensitivity to the TSV placement patterns.

Figure 12 shows the average network delay for different packet injection rates under the *Hotspot Traffic Pattern*, i.e., one node of the network is a hotspot node such that 10% of all generated packets are destined to this node. As shown in Figures 11 and 12, the proposed routing algorithm has the best performance across all the tests while ETW has the worst. The underperformance of the ETW routing algorithm can be attributed to its elevator link selection scheme which pushes most of the packets toward eastern elevator links. To investigate this, we have logged the utilization of elevator links for 100K cycles of simulation at the traffic generation rate of 0.002 packets per cycle. Results which are shown in Figure 13 clearly show that the ETW routing algorithm has uneven elevator utilization. This experiment also shows how the index sharing mechanism evenly distributes traffic among TSV links to reach a semi uniform elevator utilization. Our TSV utilization becomes better under optimal TSV placements, i.e., the best TSV placement for the proposed routing which is found by Matlab exhaustive search. Semi-even TSV utilization prevents overheating a subset of TSVs which in turn improves the network reliability by postponing TSV transient or permanent failures. The set of results in figures 11 to 13 highlight the importance of efficient elevator selection policy in 3D NoCs. Regarding Elevator-First, we should mention that its TSV assignment is done by hand and we did this in the best possible way. This means that we have shown the best case results for Elevator-First algorithm in figures 11 and 12.

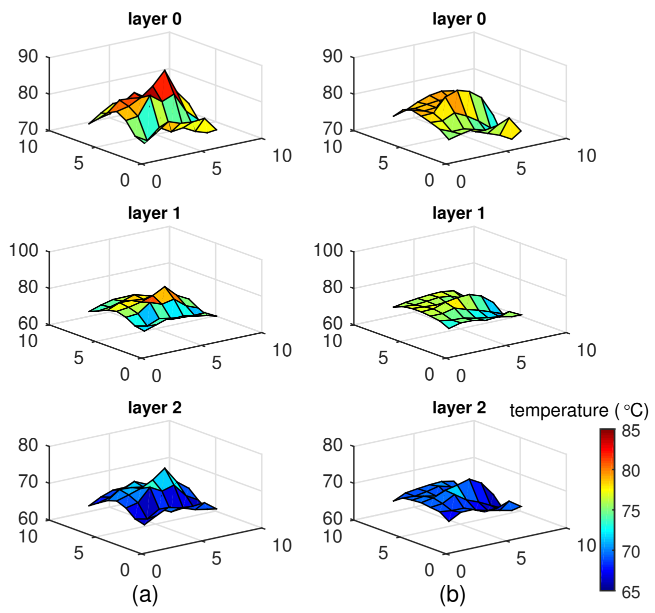


Fig. 18. Thermal distribution for (a) the Elevator-First routing algorithm and (b) the proposed routing algorithm.

B. Reliability Evaluation

The proposed routing algorithm tolerates any pattern of elevator link failures, i.e., faults F_{TSV}^{T} and F_{TSV}^{P} as long as there exists TSV connectivity between layers of the network. Figure 14.a compares average network latency of the proposed routing algorithm with ETW routing under 4 and 8 TSV failure injections, i.e., faults F_{TSV}^{T} and F_{TSV}^{P}. The Elevator-First routing is not reported here because it does not support TSV failures [5]. The proposed routing algorithm provides a better average network delay compared to the ETW routing algorithm. When a packet encounters a failed elevator link in the ETW algorithm, on average it leads to more misrouting. In the proposed routing algorithm when a faulty elevator is detected, it is treated like a non-elevator router, and there are no further attempts to route packets through the faulty TSVs, hence minimizing packet misrouting. Figure 14.b shows the performance cost associated with horizontal link failure resilience - fault model F_{HL}. In general, there is enough path diversity in source and destination layers that with limited route adaptation the performance degradation is minimal.

Figure 15 shows the network saturation point versus the number of failed elevators. In this experiment, we assumed the saturation point as the lowest injection rate at which the average network latency becomes 15 times the network diameter, i.e., $15 \times ND = 180$ cycles. As shown in Figure 15, the proposed routing tolerates up to 44 TSV failure injections while maintaining the network delay at an acceptable level. Figure 16 shows the average energy consumption for packet delivery in the presence of faulty elevators. The proposed routing algorithm has a lower energy consumption compared to ETW routing, because there is less misrouting.

We extended the simulation time to 4M cycles to evaluate the thermal behavior of the proposed routing algorithm and the possibility of occurring F_{TSV}^{T} and F_{TSV}^{P} faults. Figure 17 shows the number of overheated routers in the 4M cycles of network simulation. In this experiment, the proposed routing algorithm is compared with the Elevator-First routing [5]. In this figure, the number of overheated routers when the proposed routing continues sharing indexes for near threshold routers (referred as *"Proposed without thermal control"* in the figure) is also reported. The proposed routing algorithm reduces the number of overheated routers by approximately 75% at the end of the simulation period. The comparison of the number of overheated routers in the proposed routing with and without thermal control shows the great impact of the proposed elevator selection in thermal control of the network. Figure 18 confirms that the proposed routing algorithm has a better heat distribution over the whole chip area which helps all of the introduced fault models.

Finally, we performed a number of horizontal fault injections to count the number of dropped packets in the network due to faults F_{DL}^{D}, F_{DL}^{S}, and F_{HL}. We injected faults into the proposed, Advertiser Elevator (AE), Elevator-First (EF), and ETW routing algorithms. In order to have a fair comparison, we added the index sharing mechanism to the Elevator-First and ETW routing algorithms. We changed the algorithms to avoid faulty horizontal links through the joint use of their calculated alternative paths, misrouting policies, and the added index sharing information. Table II reflects the results. The terms "EF" and "EF+I" denote the original Elevator-First and index-sharing enabled Elevator-First, respectively. The same notation is used for the ETW algorithm. The results show that the index sharing technique helped improve the network reliability of the Elevator-First and ETW routing algorithms by approximately 8% and 5%, respectively. However, the proposed routing algorithm still has the best network reliability. The reason is that in addition to sharing indexes, the routing algorithm needs (i) enough routing adaptation to allow packets to be routed deadlock-free around faulty horizontal/vertical links and (ii) efficient rerouting techniques and policies to minimize path lengths.

The routing of the packets $S_5 \rightarrow D_5$, $S_6 \rightarrow D_6$, and $S_7 \rightarrow D_7$ - in Figure 7 - highlights some of the limitations of the "EF"/"ETW" and "EF+I"/"ETW+I" algorithms. Packets $S_5 \rightarrow D_5$ and $S_6 \rightarrow D_6$ require misrouting in the destination layer to bypass the failed horizontal links. The proposed algorithm can accommodate this operation, whereas the "EF"/"ETW" and "EF+I"/"ETW+I" algorithms will not. In the case of the packet $S_7 \rightarrow D_7$, if the horizontal links of the elevator fail, "EF"/"ETW" will also fail to route the packet, but their index-sharing enabled versions ("EF+I" and "ETW+I") and the proposed algorithm will use the index information to find an alternative healthy elevator.

C. Performance and Hardware Overheads

As described in Section IV, indexes are shared on Req/Ack wires in some consecutive cycles. During the index sharing cycles, the index sending router cannot send executing application data-flits since its request signal is being interpreted as index. Essentially, the proposed algorithm is stealing some

TABLE II
NETWORK RELIABILITY OF THE PROPOSED ROUTING ALGORITHM COMPARED TO OTHER ROUTING ALGORITHMS WITH DIFFERENT NUMBERS OF LINK FAILURE INJECTIONS.

Net. Size	Routing	Network reliability with respect to fault models $F_{DL}^{S}, F_{DL}^{D}, F_{HL}$					
		1 Fault	2 Faults	5 Faults	10 Faults	20 Faults	30 Faults
7×7×3	ETW	99.5%	99.2%	97.8%	94.2%	89.3%	83.9%
	ETW+I	99.8%	99.4%	97.8%	95.3%	92.3%	87.5%
	EF	99.2%	99%	95.8%	91%	85.9%	76.5%
	EF+I	99.5%	99.2%	96.9%	94.2%	91%	83.6%
	AE	99.2%	98.4%	95.4%	91.4%	87.3%	78.6%
	Proposed	100%	99.9%	98.8%	98%	96.7%	94.9%
7×7×5	ETW	99.7%	99.4%	98.2%	95.3%	91.3%	89.5%
	ETW+I	99.9%	99.6%	98.2%	96.7%	94.3%	90.8%
	EF	99.4%	99.4%	97.1%	93.6%	89.9%	85%
	EF+I	99.7%	99.4%	98%	95.8%	93.7%	88.7%
	AE	99.5%	99%	96.8%	93.9%	90.5%	85.9%
	Proposed	100%	100%	99.3%	98.3%	97.3%	95.8%
5×5×3	ETW	99.1%	98.4%	95.6%	87.8%	78.4%	--
	ETW+I	99.3%	99%	95.7%	90.6%	83.2%	
	EF	98.3%	97.3%	92.2%	80%	67.5%	--
	EF+I	99.3%	99.1%	96.4%	88%	74.9%	--
	AE	98.9%	98.3%	93.5%	82.6%	72.3%	--
	Proposed	99.6%	99.4%	98%	93.8%	90.1%	--

TABLE III
AREA ESTIMATES AND OVERHEAD PERCENTAGES FOR THE ELEVATOR-FIRST, ETW, AND PROPOSED ROUTERS

Router	Area (μm^2)	Overhead		
		Over baseline	Over 2-VC	Over 3-VC
Baseline (VC=1)	21241	--	--	--
EF, ETW (VC=2)	35550	67.4%	--	--
Typical with 3 VCs	50893	139.6%	43.2%	--
Proposed (VC=3)	51411	142.0%	44.6%	1%

cycles/bandwidth from the running application to communicate indexes. To evaluate the impact of this bandwidth stealing on other network performance, we simulated two networks: (i) a baseline network without index sharing and (ii) an index sharing-enabled network. A key insight is that the index sharing mechanism allows for the dynamic traffic load balancing amongst the healthy elevators which improves network performance.

In experimental set-ups of 100K simulation cycles and 100 index update events - which will seldom happen in real systems - the performance losses are negligible (Figure 19). The performance penalty associated with the proposed routing algorithm preventing router-overheating is reported in Figure 20. The network has 14 TSVs (pattern P1) with different packet injection rates. It is worth noting that without an overheating prevention strategy, the network will need to power off overheated routers which will incur much higher performance penalties [40], [38] and may result in premature network congestion [41], [40], [9].

To estimate the area overhead of the proposed routing algorithm, we implemented and synthesized the index sharing hardware module. We used the Orion [42] tool to estimate the area of the rest of the router. Both area evaluations are done using a 45nm technology. The area estimates for the baseline router, a 2-virtual channel pipelined router, a 3-virtual channel pipelined router, and the proposed router are reported in Table III). From these results, one can deduce that the area overhead associated with the index sharing registers and the control unit is less than 1% compared to the 3-VC router.

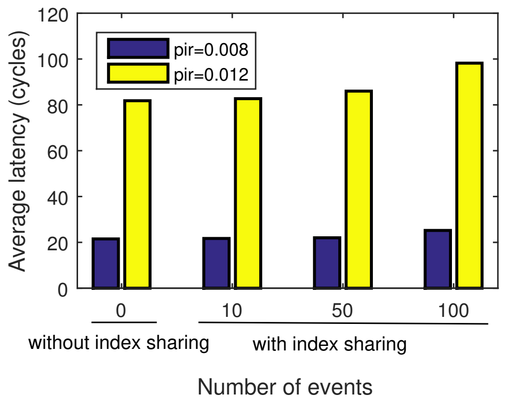


Fig. 19. Network performance loss due to the index sharing mechanism under different numbers of index update events.

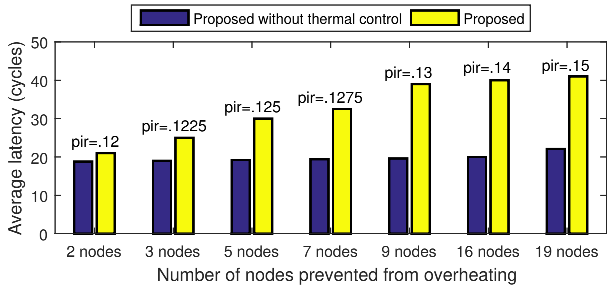


Fig. 20. Network performance loss versus number of overheat-protected routers in the network.

VII. CONCLUSIONS

This paper highlights a new class of reliability threats which have not been previously addressed in partially-connected 3D NoCs. Simulations showed that the discovered threats result in significant packet delivery failure on the chip. To address this issue, we proposed an index sharing mechanism for routers that have working TSV links. Sharing these indexes helps other routers to find nearby working TSV links and minimizes network overheads. Then, we proposed a fault tolerant routing algorithm that utilizes the indexes to find the best fault-free TSV link. Various fault injection experiments showed that our proposed routing algorithm is able to tolerate previously addressed fault models as well as the defined reliability threats. The results show that the proposed routing algorithm (i) improves average latency by at least 18% against Elevator-First and ETW routing algorithms, (ii) offers higher performance in the presence of faulty horizontal and TSV links, and (iii) efficiently manages the network heat generation to reduce the number of overheated routers.

REFERENCES

[1] J. Park, M. Cheong, and S. Kang, "R2-tsv: A repairable and reliable tsv set structure reutilizing redundancies," *IEEE Transactions on Reliability*, vol. 66, no. 2, pp. 458–466, June 2017.

[2] J. H. Lau, "Evolution, challenge, and outlook of tsv, 3d ic integration and 3d silicon integration," in *Advanced Packaging Materials (APM), 2011 International Symposium on*. IEEE, 2011, pp. 462–488.

[3] S. Pasricha and Y. Zou, "A low overhead fault tolerant routing scheme for 3d networks-on-chip," in *Quality Electronic Design (ISQED), 2011 12th International Symposium on*. IEEE, 2011, pp. 1–8.

[4] R. Salamat, M. Khayambashi, M. Ebrahimi, and N. Bagherzadeh, "A resilient routing algorithm with formal reliability analysis for partially connected 3d-nocs," *IEEE Transactions on Computers*, vol. 65, no. 11, pp. 3265–3279, 2016.

[5] F. Dubois, A. Sheibanyrad, F. Petrot, and M. Bahmani, "Elevator-first: A deadlock-free distributed routing algorithm for vertically partially connected 3d-nocs," *Computers, IEEE Transactions on*, vol. 62, no. 3, pp. 609–615, 2013.

[6] L. Jiang, Q. Xu, and B. Eklow, "On effective tsv repair for 3d-stacked ics," in *2012 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2012, pp. 793–798.

[7] C.-T. Ko and K.-N. Chen, "Reliability of key technologies in 3d integration," *Microelectronics Reliability*, vol. 53, no. 1, pp. 7–16, 2013.

[8] E. Taheri, A. Patooghy, and K. Mohammadi, "Cool elevator: A thermal-aware routing algorithm for partially connected 3d nocs," in *Computer and Knowledge Engineering (ICCKE), 2016 6th International Conference on*. IEEE, 2016, pp. 111–116.

[9] K.-C. Chen, S.-Y. Lin, H.-S. Hung, and A.-Y. A. Wu, "Topology-aware adaptive routing for nonstationary irregular mesh in throttled 3d noc systems," *IEEE transactions on parallel and distributed systems*, vol. 24, no. 10, pp. 2109–2120, 2013.

[10] S. Akbari, A. Shafiee, M. Fathy, and R. Berangi, "Afra: A low cost high performance reliable routing for 3d mesh nocs," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2012*. IEEE, 2012, pp. 332–337.

[11] P. Bogdan, T. Dumitraş, and R. Marculescu, "Stochastic communication: A new paradigm for fault-tolerant networks-on-chip," *VLSI design*, vol. 2007, 2007.

[12] S.-Y. Lin, T.-C. Yin, H.-Y. Wang, and A.-Y. Wu, "Traffic-and thermal-aware routing for throttled three-dimensional network-on-chip systems," in *VLSI Design, Automation and Test (VLSI-DAT), 2011 International Symposium on*. IEEE, 2011, pp. 1–4.

[13] S. Foroutan, A. Sheibanyrad, and F. Ptrot, "Assignment of vertical-links to routers in vertically-partially-connected 3-d-nocs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 33, no. 8, pp. 1208–1218, Aug 2014.

[14] L. Li, P. Ton, M. Nagar, and P. Chia, "Reliability challenges in 2.5d and 3d ic integration," in *2017 IEEE 67th Electronic Components and Technology Conference (ECTC)*, May 2017, pp. 1504–1509.

[15] H. Liu, Q. Zeng, Y. Guan, R. Fang, X. Sun, F. Su, J. Chen, M. Miao, and Y. Jin, "Thermal-mechanical reliability assessment of tsv structure for 3d ic integration," in *2016 IEEE 18th Electronics Packaging Technology Conference (EPTC)*, Nov 2016, pp. 758–764.

[16] B. Fu, Y. Han, H. Li, and X. Li, "Zonedefense: a fault-tolerant routing for 2-d meshes without virtual channels," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 1, pp. 113–126, 2014.

[17] V. Janfaza and E. Baharlouei, "A new fault-tolerant deadlock-free fully adaptive routing in noc," in *2017 IEEE East-West Design & Test Symposium (EWDTS)*. IEEE, 2017, pp. 1–6.

[18] H. N. Jouybari and K. Mohammadi, "A low overhead, fault tolerant and congestion aware routing algorithm for 3d mesh-based network-on-chips," *Microprocessors and Microsystems*, vol. 38, no. 8, pp. 991–999, 2014.

[19] A. Charif, N.-E. Zergainoh, A. Coelho, and M. Nicolaidis, "Rout3d: a lightweight adaptive routing algorithm for tolerating faulty vertical links in 3d-nocs," in *Test Symposium (ETS), 2017 22nd IEEE*. IEEE, 2017, pp. 1–6.

[20] P. Ren, X. Ren, S. Sane, M. A. Kinsy, and N. Zheng, "A deadlock-free and connectivity-guaranteed methodology for achieving fault-tolerance in on-chip networks," *IEEE Transactions on Computers*, vol. 65, no. 2, pp. 353–366, Feb 2016.

[21] P. Ren, M. A. Kinsy, and N. Zheng, "Fault-aware load-balancing routing for 2d-mesh and torus on-chip network topologies," *IEEE Transactions on Computers*, vol. 65, no. 3, pp. 873–887, March 2016.

[22] A. Patooghy, M. Fazeli, and S. G. Miremadi, "A low-power and seu-tolerant switch architecture for network on chips," in *Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on*. IEEE, 2007, pp. 264–267.

[23] K. Soleimani, A. Patooghy, N. Soltani, L. Bu, and M. A. Kinsy, "Crosstalk free coding systems to protect noc channels against crosstalk faults," in *2017 IEEE International Conference on Computer Design (ICCD)*, Nov 2017, pp. 385–390.

[24] A. Patooghy, S. G. Miremadi, and M. Shafaei, "Crosstalk modeling to predict channel delay in network-on-chips," in *2010 IEEE International Conference on Computer Design*, Oct 2010, pp. 396–401.

[25] P. Bogdan and R. Marculescu, "Hitting time analysis for fault-tolerant communication at nanoscale in future multiprocessor platforms," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 30, no. 8, pp. 1197–1210, 2011.

[26] M. A. Kinsy, M. H. Cho, K. S. Shim, M. Lis, G. E. Suh, and S. Devadas, "Optimal and heuristic application-aware oblivious routing," *IEEE Transactions on Computers*, vol. 62, no. 1, pp. 59–73, Jan 2013.

[27] H. Matsutani, P. Bogdan, R. Marculescu, Y. Take, D. Sasaki, H. Zhang, M. Koibuchi, T. Kuroda, and H. Amano, "A case for wireless 3d nocs for cmps," in *Design Automation Conference (ASP-DAC), 2013 18th Asia and South Pacific*. IEEE, 2013, pp. 23–28.

[28] H. Matsutani, M. Koibuchi, I. Fujiwara, T. Kagami, Y. Take, T. Kuroda, P. Bogdan, R. Marculescu, and H. Amano, "Low-latency wireless 3d nocs via randomized shortcut chips," in *Proceedings of the conference on Design, Automation & Test in Europe*. European Design and Automation Association, 2014, p. 273.

[29] Z. Qian, P. Bogdan, G. Wei, C.-Y. Tsui, and R. Marculescu, "A traffic-aware adaptive routing algorithm on a highly reconfigurable network-on-chip architecture," in *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*. ACM, 2012, pp. 161–170.

[30] Y. Xue and P. Bogdan, "Improving noc performance under spatio-temporal variability by runtime reconfiguration: a general mathematical framework," in *2016 Tenth IEEE/ACM International Symposium on Networks-on-Chip (NOCS)*, Aug 2016, pp. 1–8.

[31] M. A. Kinsy, S. Khadka, and M. Isakov, "Prenoc: Neural network based predictive routing for network-on-chip architectures," in *Proceedings of the on Great Lakes Symposium on VLSI 2017*, ser. GLSVLSI '17. New York, NY, USA: ACM, 2017, pp. 65–70. [Online]. Available: http://doi.acm.org/10.1145/3060403.3060406

[32] M. Ebrahimi, M. Daneshtalab, and J. Plosila, "Fault-tolerant routing algorithm for 3d noc using hamiltonian path strategy," in *Proceedings of the Conference on Design, Automation and Test in Europe*. EDA Consortium, 2013, pp. 1601–1604.

[33] I. Loi, S. Mitra, T. H. Lee, S. Fujita, and L. Benini, "A low-overhead fault tolerance scheme for tsv-based 3d network on chip links," in *Proceedings of the 2008 IEEE/ACM International Conference on Computer-Aided Design*. IEEE Press, 2008, pp. 598–602.

[34] E. Taheri, M. Isakov, A. Patooghy, and M. A. Kinsy, "Advertiser elevator: A fault tolerant routing algorithm for partially connected 3d network-on-chips," in *2017 IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2017, pp. 136–139.

[35] X. Jiang, L. Zeng, and T. Watanabe, "A sophisticated routing algorithm in 3d noc with fixed tsvs for low energy and latency," *IPSJ Transactions on System LSI Design Methodology*, vol. 7, no. 0, pp. 101–109, 2014.

[36] J. Lee, K. Kang, and K. Choi, "Redelf: an energy-efficient deadlock-free routing for 3d nocs with partial vertical connections," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 12, no. 3, p. 26, 2015.

[37] Y.-Y. Chen, E.-J. Chang, H.-K. Hsin, K.-C. J. Chen, and A.-Y. A. Wu, "Path-diversity-aware fault-tolerant routing algorithm for network-on-chip systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 3, pp. 838–849, 2017.

[38] E. Taheri, A. Patooghy, and K. Mohammadi, "Xyz-zxy: A minimal routing algorithm for dynamic thermal management in 3d nocs," in *Electrical Engineering (ICEE), 2016 24th Iranian Conference on*. IEEE, 2016, pp. 1539–1544.

[39] K.-Y. Jheng, C.-H. Chao, H.-Y. Wang, and A.-Y. Wu, "Traffic-thermal mutual-coupling co-simulation platform for three-dimensional network-on-chip," in *VLSI Design Automation and Test (VLSI-DAT), 2010 International Symposium on*. IEEE, 2010, pp. 135–138.

[40] L. Shang, L.-S. Peh, A. Kumar, and N. K. Jha, "Thermal modeling, characterization and management of on-chip networks," in *Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Computer Society, 2004, pp. 67–78.

[41] C.-H. Chao, K.-Y. Jheng, H.-Y. Wang, J.-C. Wu, and A.-Y. Wu, "Traffic-and thermal-aware run-time thermal management scheme for 3d noc systems," in *Networks-on-Chip (NOCS), 2010 Fourth ACM/IEEE International Symposium on*. IEEE, 2010, pp. 223–230.

[42] A. B. Kahng, B. Lin, and S. Nath, "Orion3.0: A comprehensive noc router estimation tool," *IEEE Embedded Systems Letters*, vol. 7, no. 2, pp. 41–45, June 2015.