DMPedia Lecture Notes in Computer Science & Engineering
Volume 1, pp. 535-547, 2026

International Conference on Multidisciplinary Perspectives in Advanced Computing and Technology (IMPACT 2026)
G. B. Pant University of Agriculture and Technology, Uttarakhand, India. Jan. 10-11, 2026

Reducing Latency in Network-on-Chip Using an Enhanced Mesh Topology

Abhijit Biswas[1*], Sourish Dhar[2] , Yagnyasenee Sen Gupta[3], Arnab Paul[4]

[1,2,4]Department of CSE, TSSOT, Assam University, Silchar, India

[3] Faculty of Science & Technology, ICFAI University, Tripura

abhi.021983@gmail.com[1*], dharsourish@gmail.com[2], ygnyasensee@iutripura.edu.in[3], arnab.paul@aus.ac.in[4]

Abstract

This work presents a 2D Mesh-based topology with four extra lateral links, connected to eight special routes at the corners and the middle of the mesh. This work aims to reduce the network diameter by restricting the maximum hop count to $N-1$ in a regular $N \times N$ mesh. The proposed network is simulated using Dijkstra's shortest-path routing algorithm, and it is found that lateral links are heavily used and incur average latencies of 10.662 and 11.869 cycles for $N = 4$ and $N = 5$, respectively. Also, the simulated results show a 56% reduction in maximum latency with the proposed topology for $N = 5$.

Keywords: *Network On Chip, System On Chip, Latency, Link Utilization, Topology, Hop Count.*

1. Introduction

Network-on-Chip (NoC) has proven itself to be an efficient and scalable communication architecture, driving a paradigm shift in SoC design. Traditionally, dedicated point-to-point connections were utilized for communication between various components of SoC, which greatly impacted the design to manufacturing time, design efficiency, and high power consumption due to unrestricted wire length [1, 2].

Traditional bus architecture was the first attempt to provide a well-formed communication architecture among the components of a SoC, with the advantage of being simple in structure and easy to deploy. However, the bus interconnect quickly saturates as the number of network components and processing elements increases, indicating its poor scalability to large network sizes [3]. NoC advocates an on-chip communication infrastructure, i.e., a well-formed micro network(routers and links) placed between various components, viz., cores, peripherals, and memory blocks, to enable seamless communication by improving overall bandwidth utilization and scalability, as depicted in Figure 1 [1].

Among the various 2D NoC topologies, the Mesh and Torus are most commonly used in commercial deployments [4, 5]. This is due to the simple design, regular grid of routers and links, and straightforward physical implementation in VLSI. In a typical 2-D mesh, each router connects to up to four neighbors (North, South, East, West) and to a local processing element (PE), producing a tiled "city-grid" layout that maps well to floor planning and clock-tree design. This regularity simplifies router design, supports systematic scaling to large core counts, and enables many well-studied routing and mapping methods [6].

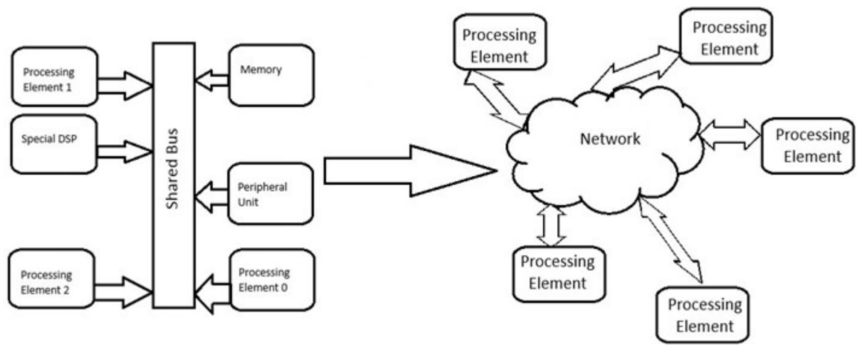


Figure 1: From Bus interconnect to NoC.

This ongoing work is an attempt to showcase the many advantages of 2D mesh topology and suggests possible changes in the topology to design a modified 2D mesh-based topology for Network-On-Chip, which is both scalable and efficient in overcoming the disadvantages of 2D mesh topology.

The remaining paper is organized in 3 sections. Section 2 deals with the literature survey and various aspects of mesh 2D topology. Section 3 presents the proposed modified mesh based NoC topology along with its routing algorithm. Section 4 presents the initial results of the simulation run of the proposed topology and finally section 5 draws the conclusion of the ongoing work.

2. Literature Survey

Pioneering work by [1,2] popularized the paradigm shift from SoC towards NoC and in this context 2D mesh topology became the bedrock of future NoC research, due to its simplicity, scalability and its seamless natural alignment with the planar structure of silicon dies as compared to its contemporaries such as the complex structure of crossbars and heirarchical structures such as trees. Lower wiring overheads, support for distributed communication and scalability to this day remain the strongest of the advantages of 2D Mesh NoC topology. Figure 2 below shows an indirect mesh topology.

The easiest of all routing that mesh can employ is XY/YX routing [7, 8], which a class of dimension order routing (DOR) which takes advantage of the grid like structure of the mesh topology. The routing is straightforward, where each flit is first routed in the X/Y direction and then makes a 90° turn to reach the final router connected to the destination IP. A XY routing algorithm as presented in [8] is rewritten and presented in a simple form in Algorithm 1. Despite its simplicity, the routing results are far from optimal. The fact that XY/YX routing algorithm is deterministic in nature, it always chooses the minimal path to route its packet from source and destination, which increases contention for network resources viz. bandwidth, path etc. Also in parallel communication between many pairs of source and destination pair, XY/YX though it is a deadlock - free routing technique[9, 10], the routing does not completely guaranty a deadlock free environment under complex communication scenarios such as when XY(X first and Y next) and YX(Y first and X next) are combined to improve performance and power consumption[11, 12, 8].

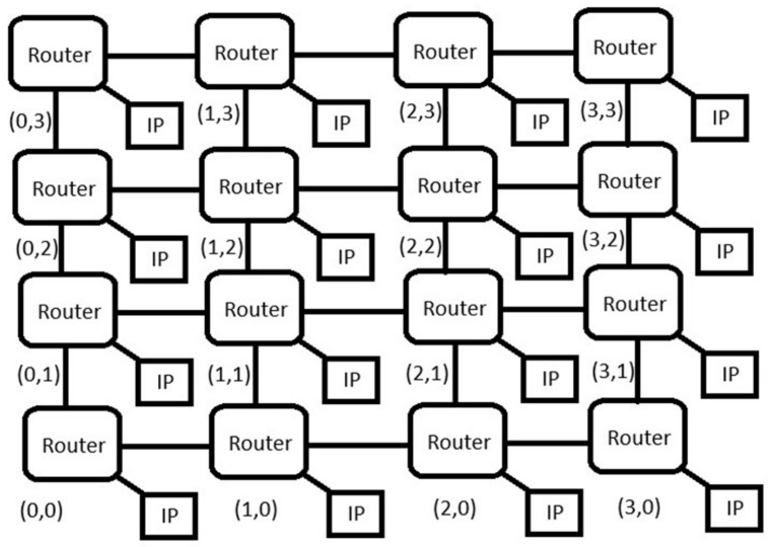


Figure 2: An indirect mesh NoC topology. The nodes marked router are the network nodes which route packets, and the nodes marked IP are intellectual properties or processing nodes.

Glass and Ni [13] in their pioneering work have proposed turn models which exhibit a fair amount of adaptability and are completely deadlock-free. The turn models, such as North-First, West-First, and Negative-First, are based on restricting certain turns during the routing process, which, in an adaptive routing environment, may form a circular dependency in the routing path. Freedom from deadlock is achieved by avoiding circular dependencies among the routed packets.

Algorithm 1: XY Routing Algorithm

```
Input: Injection router $(I_x, I_y)$, destination router $(F_x, F_y)$
Output: Next-hop sequence from $(I_x, I_y)$ to $(F_x, F_y)$
1  Inject packet at router $(I_x, I_y)$. ;                         // Start
   // Phase 1: route along $X$ until reach router $(F_x, I_y)$
2  while $I_x \neq F_x$ do
3      if $I_x < F_x$ then
4          Forward to $+X$ neighbor (East);
5          $I_x \leftarrow I_x + 1$;
6      else
7          Forward to $-X$ neighbor (West);
8          $I_x \leftarrow I_x - 1$;
   // Phase 2: single 90° turn, then route along $Y$ to $(F_x, F_y)$
9  while $I_y \neq F_y$ do
10     if $I_y < F_y$ then
11         Forward to $+Y$ neighbor (North);
12         $I_y \leftarrow I_y + 1$;
```

```
13  else
14      Forward to $-Y$ neighbor (South);
15      $I_y \leftarrow I_y - 1$;
16  Deliver packet to destination IP at router $(F_x, F_y)$.;
```

Ge-Ming Chiu in [9] have proposed an Odd - even turn model, which is fully adaptive and is also deadlock free. Ge - Ming Chiu, restricted the movement of packet in East - North direction and North not permitted to take either an East - South turn when it is an even column or a South - West turn it is in an odd column. A column is odd or even is decided by the value of the x-coordinates. If the X- Coordinate is odd, the column is termed as odd and if the value of X- Coordinate is even it is termed as an even column refer Figure 2. A turn is a 90º turn, when a packet travels from X-axis to Y- axis.

However, the mesh topology suffers from disadvantages such as high network diameter, tight edge bandwidth, and also the hop count increases when the source and destination are diagonally apart. The torus topology was introduced to mitigate some of the disadvantages by introducing Torus topology which employs uses wrap around links where the edge nodes of the Mesh topology are connected to stabilize the node degree throughout the network. However, torus topology has higher hop counts and wrap around links incur higher wire delays [14, 15] as shown in the Figure 3.

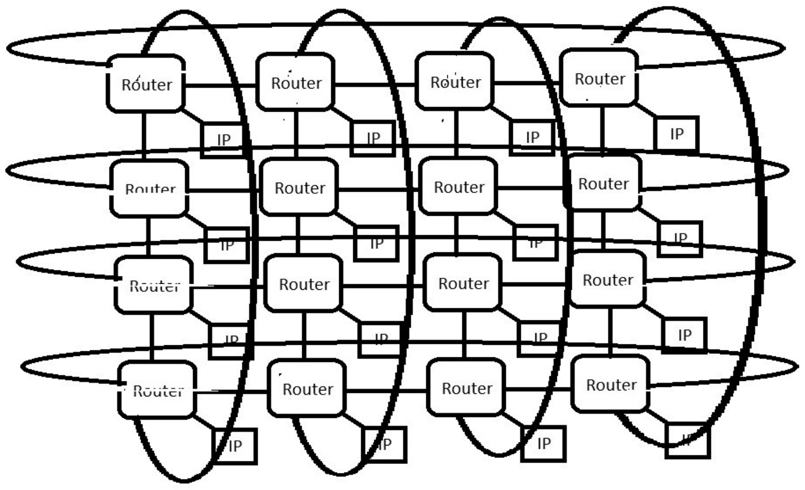


Figure 3: An indirect torus NoC topology. The nodes marked *router* are the network nodes which route packets and nodes marked *IP* are intellectual properties or processing nodes.

The torus topology also employs the routing algorithms suggested earlier, however, to avoid dead- lock, turn models, which are somewhat complex in nature can be adopted or otherwise simply employ virtual channels which gives an escape route to the packets engaged in deadlock [16]. However, using Virtual Channels (VCs) complicates the router architecture.

To tackle the problem of large network diameter authors in [17] proposed diametrical mesh topology and went on to add 8 extra links called the diametrical links which adds bypass channels to strategic nodes pair and effectively brings down the network diameter to half of the regular mesh topology. Although, the authors claim to have reduced the overall latency and power consumption, the eight extra links incur area overhead to the network.

Over the years however, many attempts have been made to minimize the network diameter of the mesh topology and torus topology by adding extra links. D- Mesh connects all the diagonally opposite node of the mesh network, the same has been extended for tori and is named as D - Torus. Another topology SD - Torus also employs extra links to connect nodes which are diagonally opposite to each other, however not all diagonally connected nodes are connected rather only diagonal nodes which are in North - West and South - East direction of each other are connected. Again, XD - Mesh topology connects the four corner nodes i.e. the node at coordinates (0,0), (0,2), (2,0) and (2,2) in a 3X3 mesh to the central node i.e. (1,1) (Refer Figure 2 for coordinate positions). Further, X - Mesh topology was proposed, which utilizes the extra - links described for XD - Mesh along with two more long bypass links connecting the four corners nodes i.e (0,0), (0,2), (2,0) and (2,2), these connections are direct connections bypassing the central node. Parallelly, the same modifications have been made to the Torus network except that the short links connecting the four-corner node with the central node (1,1) are kept as it is, but the long diagonal links are done away with, the resulting topology is named X - Torus. similarly, other topologies such as C^2 - Mesh and C^2 - Torus which employs a singular diagonal bypass link between corner node at North - West corner to the corner node at South - East direction. Further CPB - Mesh and CBP - Torus extends C^2 - Mesh and C^2 - Torus by adding two bypass links to connect the four corner nodes [18].

Most of the topologies suggested above have either very long links which crosses large portion of the chip thereby rendering high propagation delays or have too many short links which destabilizes the node degree as well as complicate the connection structure. Too many short links also effect the die area.

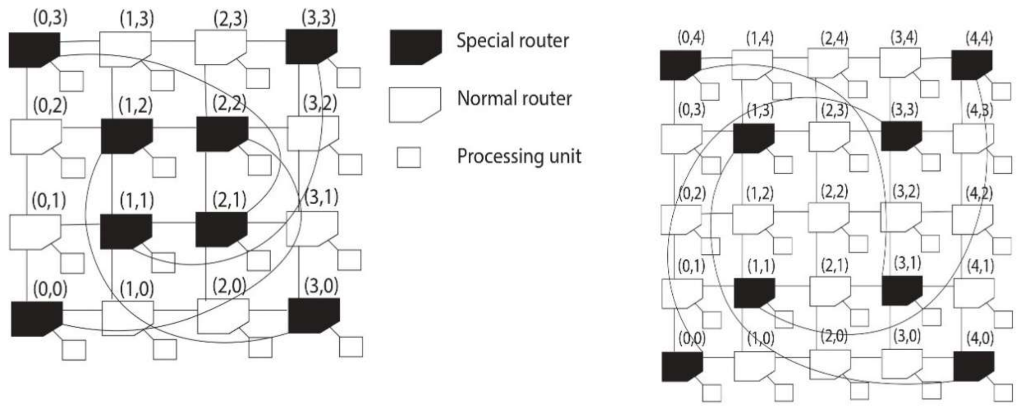
In the next section, we propose an improved modified mesh-based topology, which is a subclass of diametrical 2D mesh [17] but have lesser number of bypass link.

3. Proposed Topology

Here we present an improved mesh-based topology which is a subclass of diametrical 2D mesh topology. However, instead of 8 bypass links, the proposed topology only utilizes 4 bypass link. The main motivation of the proposed topology is to bring down hop count of Torus topology to a maximum of $N - 1$, whenever it is N and also to bring down the hop count of Mesh topology, whenever it is greater than N (Referring Figure 2, consider source destination pair to be at the diagonal positions for example source at (3,0) and destination at (0,3) the hop count in this case will be 6 which is greater than N) to N, whenever possible.

3.1. Topology

The topology of the proposed NxN mesh where $N > 3$ is given below:



(a) proposed topology where $N = 4$ (b) proposed topology where $N = 5$

Figure 4: Proposed improved mesh-based topology

A generalized topology for $N \times N$ clients has been generated for any value of N such that $N >= 4$. One client is linked to each router, and the diagonal links are connected between special 8 routers for each case of N, as shown in Figure 6.

3.1.1. Addressing

The routers and clients are addressed based on their relative position in the X-Y co-ordinate system, considering the 1st Quadrant. The address starts from (0,0) to (N-1, N-1) for any topology of our proposed work. The 4 corner nodes are addressed as (0,0), (0,N-1), (N-1,0) and (N-1, N-1). The next 4 special routers in the middle part of the topology are then located and addressed as $([\frac{N}{2}],[\frac{N}{2}])$, $([\frac{N}{2}]-1,[\frac{N}{2}])$, $([\frac{N}{2}],[\frac{N}{2}]-1)$, and $([\frac{N}{2}]-1,[\frac{N}{2}]-1)$ for any topology of proposed work, such that $N >= 4$ (whether N is even or odd).

3.1.2. Connections

Apart from the connections of a regular mesh the special routers are connected as follows:

1. (0,0) node is connected to $([\frac{N}{2}],[\frac{N}{2}])$
2. (0, N-1) node is connected to $([\frac{N}{2}],[\frac{N}{2}]-1)$
3. (N-1, 0) node is connected to $([\frac{N}{2}]-1,[\frac{N}{2}])$
4. (N-1, N-1) node is connected to $([\frac{N}{2}]-1,[\frac{N}{2}]-1)$

3.1.3 Dijkstra's algorithm based shortest path Routing algorithm

Let the node at which we are starting be called the initial node. Let the distance of node Y be the distance from the initial node to Y. Dijkstra's algorithm [19] will assign initial distance values of unity to all the regular links of mesh topology and a value of 0.5 to the special links to increase its preference. As the algorithm progresses it'll will try to find the appropriate path between source destination pair successively. The path planning stage is executed once per message at injection time. For each message with source *s* and destination *d*, the router treats *G* as a weighted graph and runs Dijkstra's algorithm from *s* to *d* to obtain the minimum-cost path. The resulting path is stored as a sequence of router coordinates and a simple lookup mapping from each router on the path to its successor.

For a message identified by *msg_id*, we conceptually associate a path plan

$$\Pi = (src, dst, path, next) \qquad (1)$$

where:

- src $= s$ and dst $= d$,
- path is the ordered list $[v_0, v_1, \dots, v_k]$ with $v_0 = s$, $v_k = d$,
- next is a map such that next$[v_i] = v_{i+1}$ for $0 \le i < k$.

Algorithm 2 summarizes the planning procedure.

Algorithm 2: Weighted Shortest-Path Planning (Per Message)

```
    Input: source $s$, destination $d$, weighted graph $G = (V, E, w)$
    Output: plan $\Pi = ($src, dst, path, next$)$
 1  if $s = d$ then
 2      path ← $[s]$
 3      next ← ∅
 4      return ($s, d,$ path, next)
    // Initialize distances and predecessors
 5  foreach $v \in V$ do
 6      dist$[v]$ ← $+\infty$
 7      prev$[v]$ ← $\perp$
 8  dist$[s]$ ← 0
    // Priority queue of (distance, node)
 9  $Q$ ← min-priority queue containing $(0, s)$
10  while $Q$ is not empty do
11      $(d_u, u)$ ← extract-min from $Q$
12      if $d_u >$ dist$[u]$ then
13          continue                                  // Skip stale entry
```

```
14    if $u = d$ then
15        break                                   // Shortest path to $d$ found
16    foreach edge $(u, v) \in E$ do
17        $w_{uv} \leftarrow w(u, v)$
18        $d_v \leftarrow d_u + w_{uv}$
19        if $d_v <$ dist$[v]$ then
20            dist$[v] \leftarrow d_v$
21            prev$[v] \leftarrow u$
22            insert or decrease-key $(d_v, v)$ in $Q$
23 if dist$[d] = +\infty$ then
       // Graph is disconnected; fall back to trivial path
24     path $\leftarrow [s, d]$
25 else
       // Reconstruct path $d \longrightarrow s$ and reverse
26     rev $\leftarrow [\,]$
27     $u \leftarrow d$
28     while $u \neq \perp$ do
29         append $u$ to rev
30         $u \leftarrow$ prev$[u]$
31     path $\leftarrow$ reverse of rev
   // Build coord $\rightarrow$ next mapping
32 next $\leftarrow$ empty map
33 for $i \leftarrow 0$ to $|path| - 2$ do
34     next[path[$i$]] $\leftarrow$ path[$i + 1$]
35 return $(s, d,$ path, next$)$
```

By construction, the returned path is a minimum-cost path from s to d with respect to the chosen weights, i.e., it minimizes $\Sigma w(u, v)$ over all routes. When all links have unit cost $w(u, v) = 1$, this reduces to minimal-hop routing. When SP(Special Purpose) links are assigned cost 0.5 and normal links cost 1.0, the algorithm automatically favors routes that use SP links when they genuinely reduce the overall cost, without enforcing any fixed pattern on how many SP hops must be used.

Once the per-message plan Π has been computed, per-hop routing is a simple lookup operation. At each router c, the next router along the chosen path is given by next$[c]$ (if defined). The router then selects the physical output port whose neighbor coordinate matches that next router. If the current router equals the destination, the flit is sent to the local port.

Algorithm 3: Next-Hop Selection Using a Weighted Path Plan

Input: current router c, destination d, plan $\Pi = ($src, dst, path, next$)$, neighbor map Neighbors(c)

Output: next output port port $\in \{$N, S, E, W, SP, LOCAL$\}$

```
1  if $c = d$ then
2  |  return LOCAL
3  end
4  if $c \notin next$ then
   |  // Safety fallback:   use minimal XY if plan is missing
5  |  return XY STEP($c, d$)
6  end
7  $c_{\text{next}} \leftarrow$ next[$c$]
8  $N \leftarrow$ Neighbors($c$)                 // port → neighbor
coordinate
9  foreach (port, coord) in $N$ do
10 |  if $port \neq LOCAL$ and $coord = c_{\text{next}}$ then
11 |  |  return port
12 |  end
13 end
   // If no neighbor matches (should not occur), use XY fallback
14 return XY_STEP($c, d$)
```

We assume that Neighbors(c) returns the map from output ports (N,S,E,W,SP) to neighbor coordinates, and that LOCAL is the local delivery port. Here, XY STEP(c, d) is a standard minimal XY step in the mesh:

$$\text{XY\_STEP}(c, d) = \begin{cases} E, & \text{if } c_x < d_x, \\ W, & \text{if } c_x > d_x, \\ S, & \text{if } cy < dy, \\ N, & \text{if } cy > dy \\ Local, & \text{if } c = d. \end{cases} \tag{2}$$

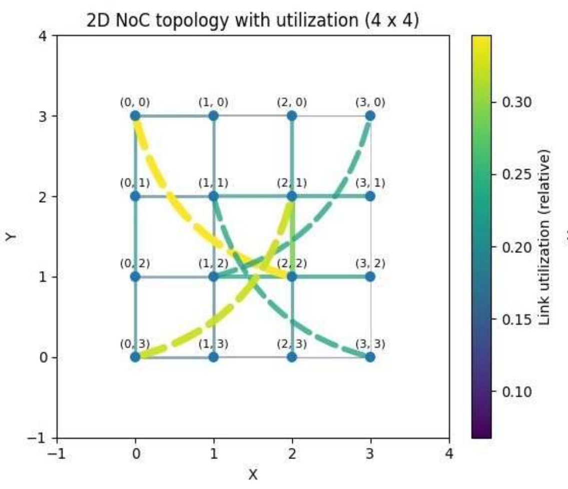
The combination of Algorithms 2 and 3 yields a routing scheme in which:

- Each message uses a globally optimal path under the specified edge weights.
- SP links are naturally preferred when they reduce the total cost, due to their lower weight.
- Per-hop forwarding remains simple: routers only consult a precomputed next-hop table and match it to one of their physical output ports.

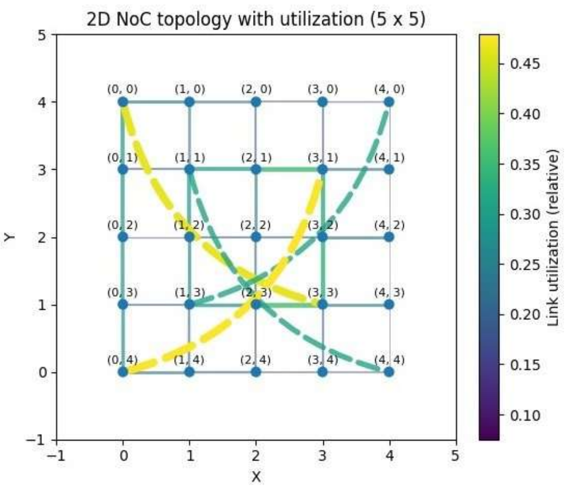
4. Experimental Results

We implemented the above topology and computed the maximum hop count (worst case) and the worst- case latency required for the proposed topology using the Dijkstra's algorithm based shortest path Routing algorithm with a sole objective of reducing the worst case hop count to $N - 1$ in an $N \times N$, which otherwise is $N + (N - 1)$ in a regular Mesh NoC topology of size $N \times N$. The proposed topology is simulated under uniform random traffic for $N = 4$

and $N = 5$. Below is the simulation result showing link utilization of the simulated run.

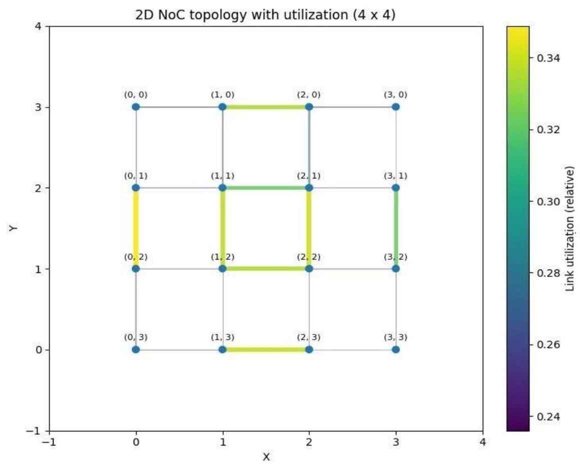


(a) 4×4 network

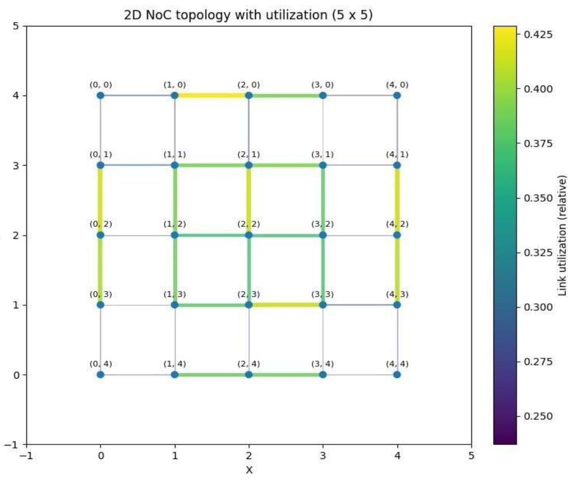


(b) 5×5 Network

Figure 5: Link utilization of the Proposed topology



(a) 4×4 Mesh Topology



(b) 5×5 Mesh Topology

Figure 6: Link utilization of the Mesh

The table below summarizes the simulated run of a 4×4 and 5×5 network size of the proposed network along with the mesh topology. It was found that the average latency of the proposed network is 10.662 and 11.869 cycles, which is lower than the mesh topology. Also, from the table s can be seen that a reduction of approximately 56% is achieved by the proposed topology in terms of maximum latency as compared to mesh topology.

Table 1: Simulated result for 4×4 and 5×5 network size

Topology	**Network Size**	**Maximum Hop Count**	**Maximum Latency**	**Average Latency**
Proposed Mesh Based Topology	4 X 4	3	69	10.662
	5 X 5	4	52	11.869
Mesh Topology	4 X 4	6	72	11.710
	5 X 5	10	92	13.070

5. Conclusion and Future Work

This work presents a mesh based NoC network with four extra links to connect eight special nodes out of which 4 are situated at the corners of the mesh and remaining four at the center. The proposed topology has been simulated utilizing a Dijkstra's algorithm based shortest path Routing algorithm. The simulated results show the average latency to be 10.662 and 11.869 Cycles for $N = 4$ and $N = 5$ respectively which is a bit lower than the mesh topology. Moreover, the proposed topology achieved a reduction of 56% in maximum latency as compared to Mesh topology with $N = 5$. However, the link utilization graph shows heavy use of two of the extra links whereas the other two extra links are being utilized in a slightly moderate way. This network may be further simulated under different traffic pattern to see the network behavior.

References

[1] W.J. Dally and B. Towles. "Route packets, not wires: on-chip interconnection networks". In: *Proceedings of the 38th Design Automation Conference (IEEE Cat. No.01CH37232)*. 2001, pp. 684– 689.

[2] L. Benini and G. De Micheli. "Networks on chips: a new SoC paradigm". In: *Computer* 35.1 (2002), pp. 70–78. DOI: 10.1109/2.976921.

[3] Xinbing Zhou, Peng Hao, and Dake Liu. "PCCNoC: Packet connected circuit as network on chip for high throughput and low latency SoCs". en. In: *Micromachines (Basel)* 14.3 (Feb. 2023), pp. 501.

[4] Inc. Arteris. *Network-on-Chip Mesh Tile Topology Speeds Time-to-Market for System-on-Chip Designs Used for Generative AI.* Accessed: 2025-10-31. Feb. 2024. URL: https://www.designnews. com/semiconductors-chips/revising-chip-design-to-gear-up-for-ai.

[5] Y. Fukazawa, S. Matsuoka, and M. Sato. "The TofuD 6-D Torus Interconnect in the Fugaku Supercomputer". In: *IEEE Micro* 41.3 (2021), pp. 46–54. DOI: 10.1109/MM.2021.3073148.

[6] Alperen Cakin, Selma Dilek, and Suleyman Tosun. "Energy-aware application mapping methods for mesh-based hybrid wireless network-on-chips". en. In: *J.*

Supercomput. 80.11 (July 2024), pp. 15582–15612.

[7] Chawade Mahendra, Mahendra Gaikwad, and Rajendra Patrikar. "Review of XY Routing Algo- rithm for Network-on-Chip Architecture". In: *International Journal of Computer and Communication Technology* (Oct. 2016). DOI: 10.47893/IJCCT.2016.1384.

[8] Wang Zhang et al. "Comparison Research between XY and Odd-Even Routing Algorithm of a 2- Dimension 3X3 Mesh Topology Network-on-Chip". In: *2009 WRI Global Congress on Intelligent Systems*. Vol. 3. 2009, pp. 329–333. DOI: 10.1109/GCIS.2009.110.

[9] Ge-Ming Chiu. "The odd-even turn model for adaptive routing". In: *IEEE Transactions on Parallel and Distributed Systems* 11.7 (2000), pp. 729–738. DOI: 10.1109/71.877831.

[10] Yue Wu, Chao Lu, and Yunji Chen. "A survey of routing algorithm for mesh Network-on-Chip". en. In: *Front. Comput. Sci.* 10.4 (Aug. 2016), pp. 591–601.

[11] Sang Muk Lee et al. "Design of a Deadlock-Free XY-YX Router for Network-on-Chip". In: *Information Technology: New Generations*. Ed. by Shahram Latifi. Cham: Springer International Publishing, 2016, pp. 701–710. ISBN: 978-3-319-32467-8.

[12] Saeid Sharifian Nia, Abbas Vafaei, and Hamid Shahimohamadi. "Deadlock Recovery Technique in Bus Enhanced NoC Architecture". In: *CoRR* abs/1209.3564 (2012). arXiv: 1209.3564. URL: http://arxiv.org/abs/1209.3564.

[13] C.J. Glass and L.M. Ni. "The Turn Model for Adaptive Routing". In: *[1992] Proceedings the 19th Annual International Symposium on Computer Architecture*. 1992, pp. 278–287. DOI: 10. 1109/ISCA.1992.753324.

[14] William J Dally and Brian Towles. *Principles and practices of interconnection networks*. Morgan Kaufmann Publishers Inc., 2003.

[15] M. Mirza-Aghatabar et al. "An Empirical Investigation of Mesh and Torus NoC Topologies Under Different Routing Algorithms and Traffic Models". In: *10th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2007)*. 2007, pp. 19–26. DOI: 10.1109/DSD.2007.4341445.

[16] Surajit Das, Abhijit Das, and Chandan Karfa. "Developing Deadlock-Free Routing Algorithms in Torus NoC: A Formal Approach". In: *ACM Trans. Embed. Comput. Syst.* 24.5s (Sept. 2025). ISSN: 1539-9087. DOI: 10.1145/3762650. URL: https://doi.org/10.1145/3762650.

[17] Poona Bahrebar and Dirk Stroobandt. "Hamiltonian Path Strategy for Deadlock-Free and Adaptive Routing in Diametrical 2D Mesh NoCs". In: *2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. 2015, pp. 1209–1212. DOI: 10.1109/CCGrid.2015.112.

[18] Usman Ali Gulzari et al. "Comparative analysis of 2D mesh topologies with additional communication links for on-chip networks". In: *Computer Networks*

241 (2024), p. 110193. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2024.110193. URL: https://www.sciencedirect.com/science/article/pii/S1389128624000252.

[19] Stuart E. Dreyfus. “An Appraisal of Some Shortest-Path Algorithms”. In: *Oper. Res.* 17.3 (June 1969), pp. 395–412. ISSN: 0030-364X. DOI: 10.1287/opre.17.3.395. URL: https://doi.org/ 10.1287/opre.17.3.395.