

A Low Latency Scalable 3D NoC Using BFT Topology with Table Based Uniform Routing

Avik Bose[†], Prasun Ghosal^{†,‡}, Saraju P. Mohanty[‡]

[†]Indian Institute of Engineering Science and Technology, Shibpur, Howrah 711103, WB, India

[‡]University of North Texas, Denton, TX 76203, USA

Email: avikbose1145@gmail.com, prasun@ieee.org, saraju.mohanty@unt.edu

Abstract—Due to the limitations of traditional bus based systems, Network-on-Chip (NoC) has evolved as the most dominant technology in the paradigm of communication-centric revolution, where, besides the computation, inter-communication between the cores is an indispensable aspect of a SoC. Furthermore, the emergence of three dimensional integrated circuits (3D-ICs) has resulted in better performance, functionality, and packaging density compared to traditional planar ICs. The amalgamation of these two technologies, the 3D NoC architecture, can combine the benefits of these two new domains to offer an unprecedented performance gain. In this paper, we present a new 3D topological NoC design based on the butterfly fat tree (BFT) topology with an efficient table based uniform routing algorithm for 3D NoC. Extensive simulation experiments have been performed for BFT and compared to mesh, torus, butterfly and flattened butterfly topologies against four performance metrics viz. overall average latency, overall average acceptance rate, overall minimum acceptance rate, and average hop counts. There are significant latency improvements of 43-89 %, 83-88 %, 46-96 %, and 31-95 % over other topologies respectively. Average hop count is improved by 30 % and 13 % over mesh and torus. Also, there are improvements in average acceptance rate and minimum acceptance rate of 1-8 % and 5-14 % respectively for flattened butterfly and 6-9 % and 6-13 % over torus. Results evidently show that BFT is a very good choice for low network latency and faster communication.

I. INTRODUCTION AND MOTIVATION

A. Introduction

For the last decade, as standard CMOS based integrated circuit design is facing severe threats [1], network-on-chips (NoC), and three dimensional integrated circuits (3D IC) have been evolved as alternative and promising sustainable solutions. NoC supports not only processing with multiple cores/IPs but also the communication architecture among themselves implemented on the same chip itself. In addition, 3D integration provides the possibility of integrating numerous heterogeneous modules (such as analog, digital, memory, RF, optical) in multiple device layers of the same chip within a single package. With the tremendous success of the above two technologies, researchers got attracted to boost the performance by integrating these two path breaking technologies and building a 3D NoC. This work presents an advancement in same track.

B. Related Research

Performance of an NoC depends on three major factors, viz. network latency, execution time, and heat dissipation. In

[2], the design of a low-latency on-chip network router is presented. Reference [3] emphasizes the various challenges of the overall design process of an NoC to achieve a functionally correct and reliable system. [4]–[6] present analysis over performance of underlying communication architectures of NoC. The impact of TSV to 3D NoC design was studied in [7]. The use of DTDMA pillar for inter layer communication has been shown in [8]. Details of its operation may be found in [9]. Advantages of 3D ICs are analyzed in [10]. In [11], it is shown that the integration of processor and memory in a stack enables a large increase in performance. 3D ICs were proposed to improve performance of chip multiprocessors in [8]. In addition to the reduced footprints the 3D-NoC chips provide a much better performance as compared to the 2D-NoC chips.

C. Organization of the paper

The overall organization of the rest of the paper is as follows. Section II summarizes the novel contributions of this paper. In Section III, a detailed formulation of the present problem is presented. The proposed solution and necessary details of proposed algorithms are presented in Section IV. Simulation framework, experiments, and results are discussed and analyzed in Section V. Finally, Section VI concludes the paper.

II. NOVEL CONTRIBUTIONS OF THE PAPER

In the present work, a completely new 3D topological design of a NoC based on butterfly fat tree (BFT) topology with a newly proposed table based routing methodology is presented. First, use of BFT topology is justified with extensive simulation results to achieve low network latency and faster communication, which are, two major bottlenecks of today's NoCs. Comparisons show that the proposed design is capable of achieving lower latency even with higher injection rate than existing ones. Second, for inter layer communication in the proposed architecture, we have used Time-Division Multiple Access (TDMA) pillars [8], [9] offering the most reliable technological solution available so far for this purpose. Third, to improve routing methodology, proposed efficient table based routing algorithm is designed in such a manner so that determined paths in cases of both intra and inter layer communication are fast but simple. Lastly and very importantly, the proposed topology is well scalable to fit with

the demand of more and more numbers of cores by today's ever increasing complex computational needs.

III. PROBLEM DESCRIPTION

A common 3D NoC interconnection structure with mesh topology is shown in Fig. 1. Due to the close proximity of layers in 3D NoC structure the signals traveling in the vertical (inter-layer) direction is much faster than the horizontal (intra-layer) in their 2D counterpart [8]. However, an inter-layer connection requires the addition of two more links (up and down) to each router which leads to an increase in complexity as well as the blocking probability inside the router. Moreover, being a multi-hop communication fabric, the traditional NoC routers can not be placed on the vertical path in a NoC as the multi-hop delay and the router delay would overshadow the ultra fast propagation time. Thus it is desirable to have single hop communication among the layers because of the short distance between them [8]. Also, the number of vertical pillars should be kept low to reduce the manufacturing cost of a 3D NoC. It induces a new problem for the IP blocks with close vicinity to the pillar nodes on a layer giving more advantage in case of inter layer communication than those that are at relatively distant position from the pillar nodes. Scalability is also another major challenge towards the design of a proper 3D topology for NoC today. Keeping those in mind our objective is to propose a novel 3D NoC topology with proper routing method that can address all the above issues.

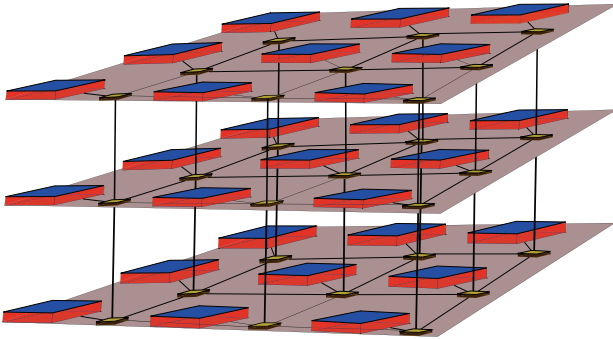


Fig. 1. Connection structure in a typical 3D Mesh NoC topology.

IV. PROPOSED SOLUTION

A. Overall Architecture

Overall architecture of a layer in the proposed design of 3D NoC based on BFT topology is shown in 2. Four BFTs are connected together having four root nodes each (colored red, orange, blue, and green). Root nodes with same color are connected together to form a complete graph as presented in Fig. 3). Reason behind connecting the root nodes in the above manner is to reduce the network latency in terms of hop count.

B. Inter Layer Communication and Routing Architecture

For inter layer communication DTDMA pillars are used that eliminate transactional character commonly associated with buses [8], [9] by employing a dynamic bus arbitration (thus close to 100 % bandwidth efficient). Single-hop

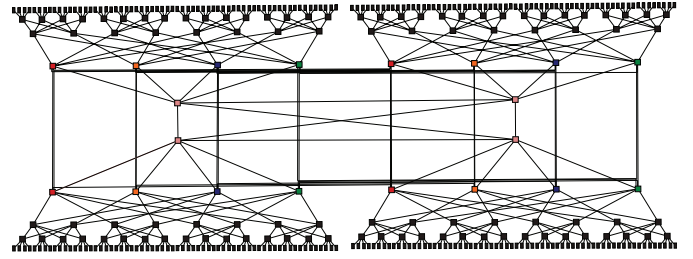


Fig. 2. Logical diagram of four connected Butterfly Fat Trees.

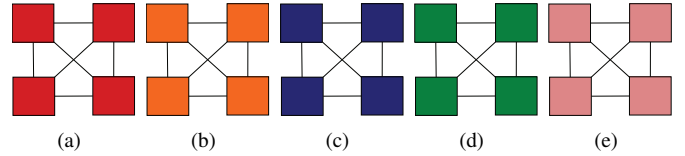


Fig. 3. Connectivity between border and root routers. Each of (a),(b),(c), and (d) is a four root level router, each pertaining to a particular BFT. (e) Border routers of a floor, each of which is dedicated to a particular BFT.

communication and transaction-less arbitrations allow for low and predictable latencies. Furthermore, hybridization of NoC router with bus architectures requires only one additional link (in the place of two) on NoC router.

Fig. 4 depicts the floor planning of a 3D NoC layer for a single BFT. The circular DTDMA pillar node is shown in the center of the picture. It is connected to a special router called as border router (responsible for regulating traffic across different layers of the chip). This router is the gateway for inter layer communication but in a different manner.

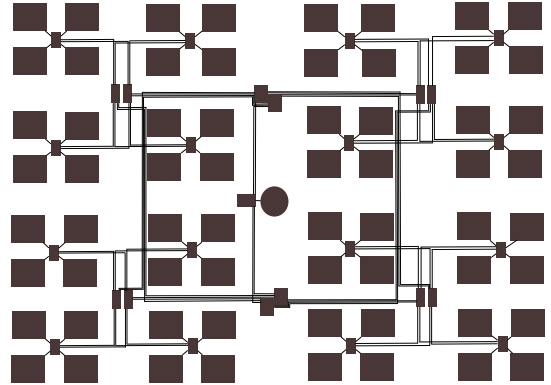


Fig. 4. Floor plan of a single BFT as a locality of a pillar node.

C. Routing Computation Logic

Routing computation logic is also made simplified. Assuming the root level as zero, level 1 routers are shown in Fig. 4, situated in the center of a group of four IP Block groups, where each IP block group contains a leaf (level 2) router and corresponding four IP blocks. Fig. 5 shows the complete layer of a 3D NoC according to our proposed design. As in Fig. 2, the four BFTs are shown here. There are four pillar nodes on the layer. The locality of each pillar node is a floor plan of a BFT according to Fig. 4.

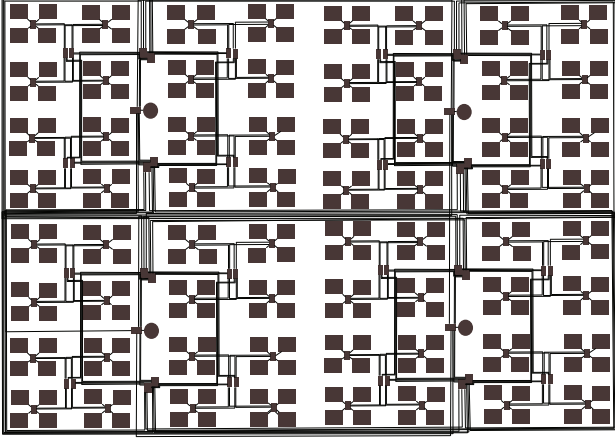


Fig. 5. Floor plan of a complete NoC layer comprising four BFTs as the localities of their respective pillar nodes.

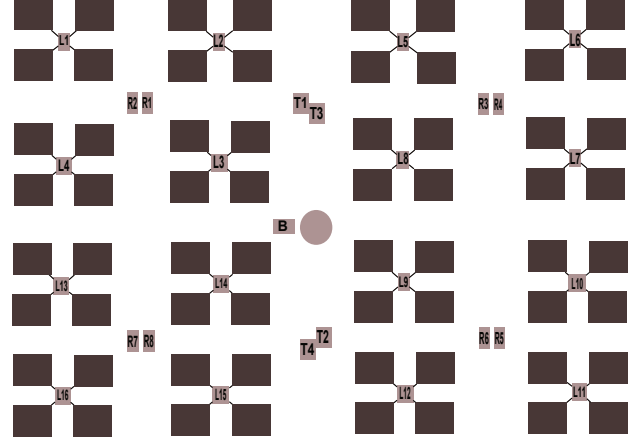


Fig. 6. Distribution of routers across different scopes for a single BFT.

Total hop count



1 hopping from local router of source node to one of two parent (level 2) routers connected to either red and blue or orange and green root routers.

+

1 hopping from parent router to one of those two corresponding root routers (red/blue or orange/green).

+

1 hopping according to Fig. 4 to reach corresponding same coloured root router of destination node's BFT

+

2 hopping to reach local router of destination IP block from corresponding root router.



Total 5 hopping.

D. Different Routing Scopes

From routing perspective, whole chip is divided into several scopes of routing as follows.

1. **Layer scope:** A 3D NoC chip consists of several layers.
2. **Tree scope:** Every layer is made up of four BFTs. In every BFT, there are four regions.
3. **Region scope:** Each region is made up of two level 1 routers and their siblings, shown in Fig. 4, designed according to Figure 2.
4. **Local scope:** Lastly, each region is made up of four localities, where each locality comprises of one leaf router and four IP blocks connected to it according to Figure 4.

Fig. 6 shows the overall distribution of routers in a BFT. Local, Regional, Root and Border routers are denoted by L, R, T, and B respectively. Region 1 has two regional routers (R1, R2), and four local routers (L1, L2, L3, L4) and so on. T1, T2, T3, and T4 are four root level routers. T1 and T3 are connected with R1, R3, R5, and R7. T2 and T4 are connected with R2, R4, R6, and R8. This reflects the exact topology shown in Fig. 2 as logical diagram and Fig. 4 as floor plan. B

is the border router connected to the circular pillar node and four root routers shown in Fig. 6. Border routers of a layer are B1, B2, B3, and B4 respectively for the four BFTs belonging to that layer. Each local router is connected to four IP blocks.

E. Proposed Routing Algorithm

Let, L_s and L_d are local routers connected to source and destination nodes respectively, and R_s and R_d are corresponding regional routers. Pseudo code is presented in Algorithm 1.

Analysis: When source and destination nodes belong to different localities of the same tree, hop count is two. When it's a matter of different regions of a tree, then penalty is of four hop counts and it doesn't matter where two IP blocks are situated on the layer. For different trees, it's five hop counts [described earlier]. Lastly, when source and destination nodes belong to different layers, for the same tree number, it is seven, and it is eight for different tree number. So for all those pairs having source nodes with same scope and also destination nodes with same scope, hop count is always the same irrespective of where exactly the source and destination IP blocks are situated. Therefore, algorithm 1 is termed as uniform routing algorithm for such uniform hopping distance characteristics .

F. Inclusion of Routing Tables

Algorithm 1 is implemented using routing tables stored on routers. A routing table contains only those pieces of information that are related to the scope of the respective router. Different routing tables and their information content are summarized in table I.

G. Addressing Format

The addressing format used to help each router along a path from source to destination is shown in Fig. 7. It is a total $(m + 8)$ bits address. First m bits specify the layer number, next 2 bits specify the tree number out of four BFTs in a layer, next 2 bits contain the region number out of four regions of a BFT, next 2 bits denotes the locality number out of four localities of a region, and the last 2 bits field contains the node number in that locality out of four IP blocks in that locality.

Determine_Path()

Input : Source node index, Source node layer, Destination node index, and Destination node layer.

Output: Routing path.

```

begin
  Locate layer, tree, region, and locality for both source and destination;
  if Layers = Layerd then
    if Trees = Treed then
      if Regions = Regiond then
        if Localitys = Localityd then
          Place flit into respective physical channel;
        end else
          Reach either of two Rs connected to Ls;
          Reach Ld, connected to Rs;
          Place the flit into respective physical channel;
        end
      end else
        Reach either of two Rs connected to Ls;
        Reach either of two Ts connected to Rs;
        Reach respective Rd, connected to Ts;
        Reach Ld, connected to Rd;
        Place the flit into respective physical channel;
      end
    end else
      Reach either of two Rs, connected to Ls;
      Reach either of two Ts, connected to Rs;
      Reach respective Td, connected to Ts;
      Reach respective Rd, connected to Td;
      Reach Ld, connected to Rd;
      Place the flit into respective physical channel;
    end
  end else
    Reach either of two Rs, connected to Ls;
    Reach either of two Ts, connected to Rs;
    Reach Bs, connected to Ts;
    if TreeNumbers = TreeNumberd then
      Reach pillar node connected to Bs;
      Jump to LAYERd through DTDMA pillar;
      Reach Bd from the pillar node, connected to Bd;
      Reach any of the four Td;
      Reach respective Rd, connected to Td;
      Reach Ld, connected to Rd;
      Place flit into respective physical channel;
    end else
      Reach the Border router of LAYERs where respective Tree
      number matches Tree number of destination node of LAYERd;
      Reach pillar node connected to that Border router;
      Jump to LAYERd through DTDMA pillar;
      Reach Bd from the pillar node, connected to Bd;
      Reach any of the four Td;
      Reach respective Rd, connected to Td;
      Reach Ld, connected to Rd;
      Place the flit into respective physical channel;
    end
  end
end

```

Algorithm 1: The Main Routing Methodology.

TABLE I
DIFFERENE ROUTING TABLES ACCORDING TO SCOPES

Local Routing Table	Table (Node Number, Link Number)
Regional Routing Table	Table (Locality Number, Link Number)
Root Routing Table	Table1 (Region Number, Link Number) Table2 (Tree Number, Link Number)
Border Routing Table	Table (Router Number, Link Number)

Layer Number	Tree Number	Region Number	Locality Number	Node Number
m bits	2 bits	2 bits	2 bits	2 bits

Fig. 7. Address format of an IP block

H. Working Principles of Routers using the Address

Routers of different scopes play different roles together in order to forward a flit from source to destination. Pseudo codes of different algorithms executed by local, root, and border routers are presented in algorithms 2, 3, and 4. Regional router also works in a similar fashion between local and root routers (omitted due to paucity of space). Here, L_d , T_d , R_d , l_d , and n denote layer number, tree number, region number, locality number, and node number of the destination node respectively, and L , T , R and l are with similar meaning for current router that have a flit to forward.

Forward_Local_Route()

Input : Flit address.

Output: Next hop.

```

begin
  if Ld = L then
    if Td = T then
      if Rd = R then
        if ld = l then
          Search Table by n to find corresponding link number;
          Place flit into physical channel corresponding to that link found;
        end else
          Forward flit to either of two regional routers, connected to local router;
        end
      end else
        Forward flit to either of two regional routers, connected to local router;
      end
    end else
      Forward flit to either of two regional routers, connected to local router;
    end
  end else
    Forward flit to either of two regional routers, connected to local router;
  end
end

```

Algorithm 2: Forwarding Technique of a Local Router.

Forward_Root_Route()

Input : Flit address.

Output: Next hop.

```

begin
  if Ld = L then
    if Td = T then
      Search Table1 by Rd to find corresponding link number;
      Forward flit to respective regional router connected through that link found;
    end else
      Search Table2 by Td to find corresponding link number;
      Forward flit to respective root router connected through that link found and belongs to BFT of destination node;
    end
  end else
    Place flit into respective physical channel of corresponding border router connected to root router;
  end
end

```

Algorithm 3: Forwarding Technique of a Root Router.

Forward_Border_Route()

Input : Flit address.

Output: Next hop.

```

begin
  if flit comes from any of three other border routers then
    Place flit into respective physical channel the pillar node is connected to;
  end
  if flit comes from pillar node the border router is connected to then
    Forward flit to any of four root routers of BFT the border router is connected to;
  end
  if flit comes from any of four root routers the border router is connected to then
    if  $T_d = T$  then
      Place flit into respective physical channel the pillar node is connected to;
    end
    else
      Search Table by  $T_d$  to find corresponding link number;
      Forward flit to respective border router of the BFT connected through that link found and Tree number matches to Tree number of destination node of  $L_d$ ;
    end
  end
end
end

```

Algorithm 4: Forwarding Technique of a Border Router.

I. Addressing scalability

Scalability of the design can easily be increased by incorporating more BFTs on a floor. For each BFT, one DTDMA pillar and one border router have to be introduced. The border router should be connected to the pillar node to regulate 3D traffic. It has to be connected to rest of the border routers on the respective layer too. Each root level router of distinct colour of the newly introduced BFT should be connected to the rest of same coloured root routers on the respective layer, thus providing a complete connection. With an efficient virtual channel management strategy, flit movement across NoC should definitely work as faster as in the case of four BFTs shown. It may seem costly to add a DTDMA pillar for each BFT we incorporate, but, it should be remembered that, one pillar takes the responsibility of regulating 3D traffic from 64 IP blocks on a particular layer, which is quite high.

V. EXPERIMENTAL RESULTS AND VALIDATION

A. Experiment details

Each simulation has three basic phases viz. warm up, measurement, and drain. Current latency and throughput (rate of accepted packets) for the simulation is determined after each sample period and overall throughput is determined by the lowest throughput of all destinations in the network. Simulation is performed for BFT and compared to mesh, torus, butterfly and flattened butterfly topologies. Simulation results are shown in Fig. 8, 9, 10, and 11. Comparisons are done against overall average latency, overall average acceptance rate, overall minimum acceptance rate, and average hop counts. Comparative improvements over other topologies are summarized in table II.

B. Discussions

It may easily be observed from simulation results that average latency is very low in case of BFT compared to

TABLE II
SUMMARY OF COMPARATIVE IMPROVEMENTS (%) OF PROPOSED NOC OVER OTHER TOPOLOGIES FOR DIFFERENT PERFORMANCE METRICS

Performance metrics → Topologies ↓	Avg Hop Count	Avg Acceptance Rate	Avg Latency	Min Acceptance Rate
Mesh	30	NIL	43-89	NIL
Torus	13	6-9	83-88	6-13
Butterfly	NIL	NIL	46-96	NIL
Flattened Butterfly	NIL	1-8	31-95	5-14

mesh with an improvement of 43-89 %. There is an average improvement of 30 % in case of average hop count. Packet acceptance rate is initially low for BFT but with increasing injection rate it increases. In case of torus, BFT outperforms it in every aspect with 13 % improvement in average hop count, 6-9 % in average acceptance rate, 83-88 % improvement in latency, and 6-13 % in minimum acceptance rate. In case of butterfly the latency of BFT is much smaller (improvement 46-96 %) and packet acceptance rate is low initially for some time, but again it increases with increasing rate of injection. The butterfly topology crashes at 0.032 injection rate only. For obvious reasons the hop count of butterfly is smaller than BFT. In case of flattened butterfly, BFT outperforms it in every aspect (1-8 % improvement in average acceptance rate, 31-95 % latency improvement, and 5-14 % in minimum acceptance rate) except average hop count.

VI. CONCLUSION

Proposed BFT topology can withstand heavy workload while still maintaining low latency, and the acceptance rate also increases with increasing injection rate. This is because of the uniform and load balancing connectivity of BFT where we have more than one path between a pair of source and destination but with same hop count. On the other hand, all other topological designs have failed to balance the load and sometimes crash. If the acceptance rate is very high, and the latency is also very high, then it doesn't make any sense, especially for those applications that require interactive communication between threads. Performance is degraded leading to major performance bottlenecks. Similarly, with moderate acceptance rate and sufficiently low latency it definitely helps the communication between IP blocks. If routers can be designed in such a way that they can have the capability to balance load and control congestion efficiently, then with this design we can achieve a really effective NoC system for interactive applications with threading capability. Future works may be in the direction of investigating the thermal effects and optimizing it accordingly with a suitable core placement strategy, investigating and improving performance using real time application mapping and so on.

REFERENCES

- [1] "International Technology Roadmap for Semiconductors." Available: <http://www.itrs.net/>.
- [2] R. Mullins, A. West, and S. Moore, "Low-latency virtual channel routers for on-chip networks," in *31st Annual International Symposium on Computer Architecture*, 2004.
- [3] L. Benini and G. D. Micheli, "Networks on Chips: A New SoC Paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70-78, 2002.

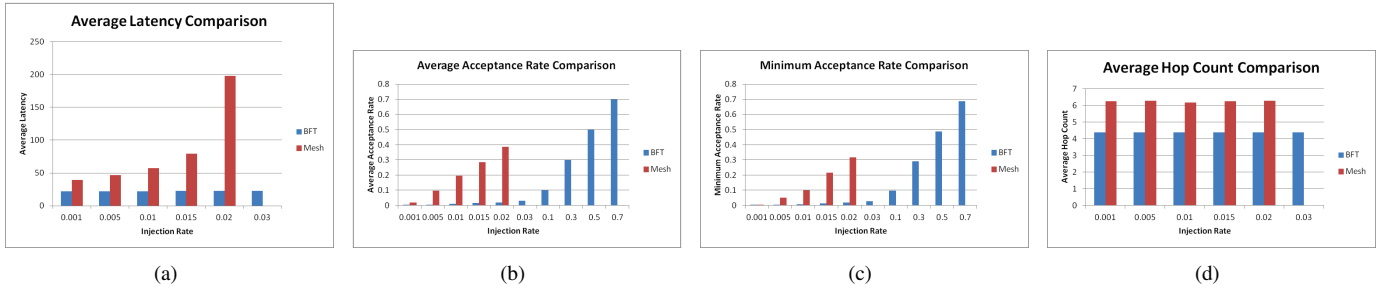


Fig. 8. Simulation Results for Mesh and BFT (a) Overall average latency (b) Average acceptance rate (c) Minimum acceptance rate (d) Average hop count.

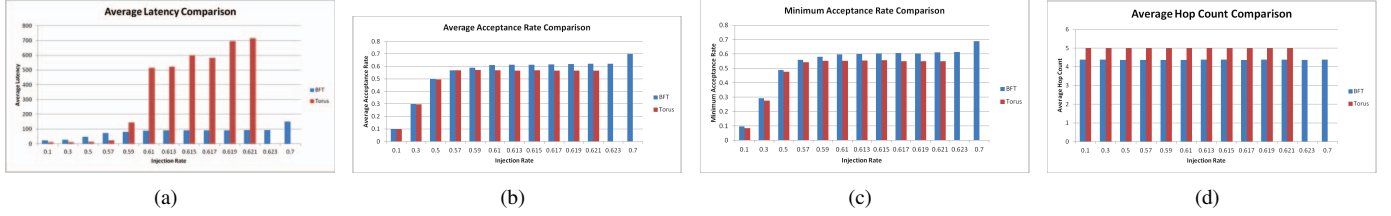


Fig. 9. Simulation Results for Torus and BFT (a) Overall average latency (b) Average acceptance rate (c) Minimum acceptance rate (d) Average hop count.

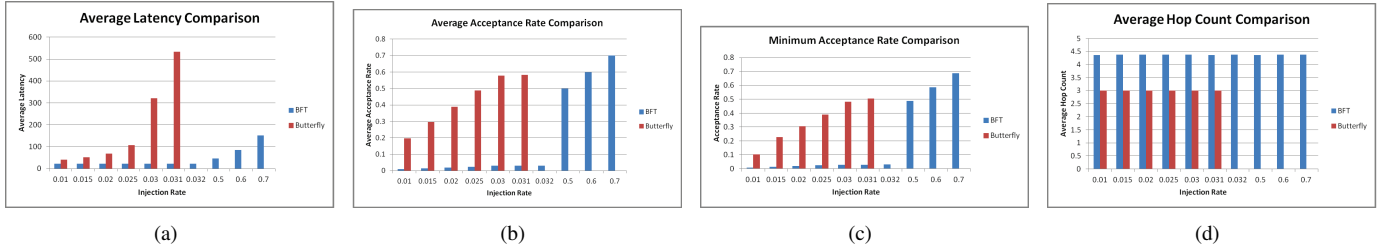


Fig. 10. Simulation Results for Butterfly and BFT (a) Overall average latency (b) Average acceptance rate (c) Minimum acceptance rate (d) Average hop count.

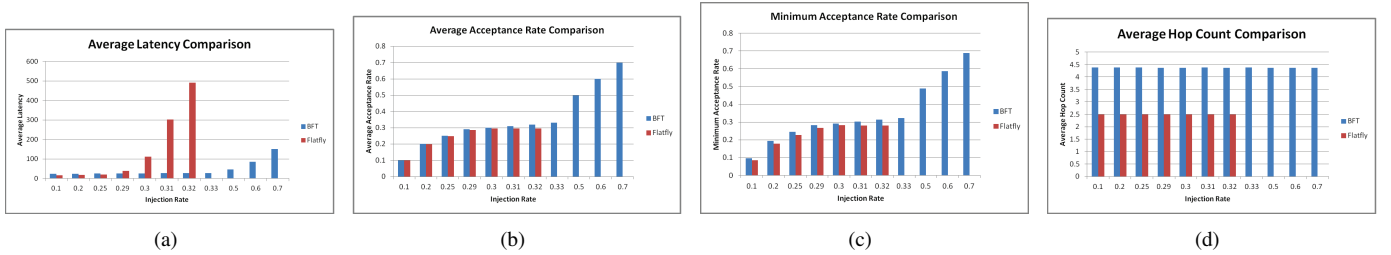


Fig. 11. Simulation Results for Flattened Butterfly and BFT (a) Overall average latency (b) Average acceptance rate (c) Minimum acceptance rate (d) Average hop count.

- [4] K. Lahiri, A. Raghunathan, and S. Dey, "Evaluation of the Traffic-Performance Characteristics of System-on-Chip Communication Architectures," in *Proceedings of the Fourteenth International Conference on VLSI Design*, pp. 29–35, 2001.
- [5] A. Agarwal, C. Iskander, and R. Shankar, "Survey of Network on Chip (NoC) Architectures & Contributions," *Journal of Engineering, Computing and Architecture*, vol. 3, pp. 21–27, 2009.
- [6] F. Gebali, H. Elmiligi, and H. W. El-Kharashi, *Networks-on-Chips - Theory and Practice*. CRC Press, Taylor & Francis Croup, 2009.
- [7] T. C. Xu, P. Liljeberg, and H. Tenhunen, "A Study of Through Silicon Via Impact to 3D Network-on-Chip Design," in *International Conference on Electronics and Information Engineering*, 2010.
- [8] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir, "Design and Management of 3D Chip Multiprocessors Using Network-in-Memory," in *33rd International Symp. Computer Architecture (ISCA 2006)*, pp. 130–141, June 2005.
- [9] C. Nicopoulos, V. Narayanan, and C. R. Das, *Network-on-Chip Archi-*

tectures - A Holistic Design Exploration. Springer, 2009.

- [10] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzone, "Demystifying 3D ICs: the pros and cons of going vertical," *Design & Test of Computers, IEEE*, vol. 22, no. 6, pp. 498–510, 2005.
- [11] P. Jacob, O. Erdogan, A. Zia, P. Belemjian, R. Kraft, and J. McDonald, "Predicting the performance of a 3D processor-memory chip stack," *IEEE Design Test of Computers*, vol. 22, no. 6, pp. 540–547, 2005.