A Multipath Network-on-Chip Topology

Kaushik Ray[1] & Alakesh Kalita[2]& Abhijit Biswas[3]
Department of Information Technology,
Assam University, Silchar
Assam, india
e-mail: {[1]ray.kaushik90,
sunucse255[2]abhi.021983[3]}@gmail.com

Md. Anwar Hussain[4]
Department of Electronics and Communication Engineering
NERIST, Nirjuli
Arunachal Pradesh, India
e-mail: ah@nerist.ac.i

***Abstract*— Network-on-Chip (NoC) is a new on chip communication design system that replaced bus-based communication in System-on-Chip (SoC). It provides efficient and scalable communication among the IPs (Intellectual Properties) in a single chip. Mainly, it was introduced to fulfill the growing demand of electronics system which needs to integrate a large number of computational resources into a single chip with an aim to get high performances, And to get efficient performance, the interconnection network plays an important role in chip design. The network architecture or topology should be highly scalable so that it can incorporate a large number of IPs without degrading the performance. In this paper, we proposed a network topology that provides multiple path diversity and high degree of scalability. The architecture of the proposed topology is described along with a packet based adaptive routing algorithm. We simulated the network using an open source simulator named OMNeT++. The topology is tested for average latency, number of events that to be carried in the simulation using a packet based routing algorithm and has been compared with existing Fat tree topology.**

Keywords—Network-On-Chip (NoC); System-on-Chip (SoC); intellectual properties; topology; packet; switch;adaptive routing algorithm;

I. INTRODUCTION

Network-on-Chip (NoC) was introduced as a new on chip interconnection approach to overcome the problems faced in earlier System-on-Chip (SoC) [2] which integrates several computational resources in a single chip. In earlier system-on-chip (SoC) design was based on bus architecture where all the IPs viz CPU, memory, DSPs, etc are interconnected using a single shared bus [8]. This shared bus has its own limitations in terms of scalability, bandwidth requirements, latency, throughput, etc. After that a hierarchical bus architecture also proposed to give partial solution to the above problems [4], but it also falls short in terms of scalability, latency, throughput, etc.

For a long lasting solution to these problems, researcher proposed a network within the chip. The idea for this micro network architecture had been adopted from general computer networks where IP cores (processing units) communicate to each other via switches.

Since, then many new topologies had been proposed viz mesh, torus, fat tree, butterfly etc along with routing algorithms like XY, Odd-Even and so on. Though these topologies and routing algorithms overcome the issue of scalability, throughput, and path diversity. But there are still a huge number of problems for research to get more efficient on chip connection.

The rest of this paper is organized in VI sections. Section II provides a brief review of some related work done in this domain. Section III gives an explanation of our proposed network architecture. Section IV provides the simulation result of our proposed network. In section V we compared the result with an existing topology i.e. fat tree. At last, in section VI, we made a conclusion and presented the future work.

II. RELATED WORK

Network-on-Chip was proposed for on chip interconnection, which is used for packed based communication that provides a high degree of scalability and efficient performance. NoC has been a very vast area for researchers with various aspects like network architecture, routing, fault tolerance, power consumptions, etc. In this section, we are going to review some of the popular works that had been done in this domain to solve architectural problems of NoC.

Earlier, there are several topologies like star, ring, mesh, torus, binary tree and many others. However, among all this mesh and tree topologies like butterfly, Fat tree shows some goods results than the ring and star Topology.

S. Kumar et al. in [9] have proposed a mesh based topology called CLICHE (Chip-Level Integration of Communicating Heterogeneous Elements). The architecture is based on an m X n matrix of switches, interconnecting the IP cores as shown in figure 1.There, each switch with the exception of the edge switches are interconnected to four neighboring switches and one IP block. While its intriguing simplicity is its major advantage. The topology does not scale well in terms of throughput and latency. Also, the die area in terms of diameter became very large for a large size network.

978-1-5090-2552-7/16/$31.00 ©2016 IEEE

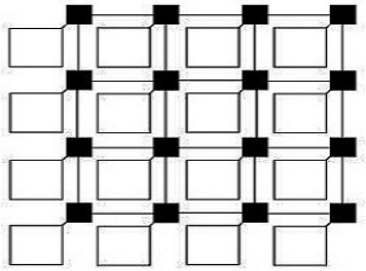

Figure. 1. Cliché Topology

Later, Torus is an advanced version of mesh topology. The problem of high latency between ends of mesh topology is overcame by the Torus topology by joining the end nodes by a single wire as shown in figure 2. Here the diameter between two end nodes has been reduced, due to long wire connection there is still the problem long delay.

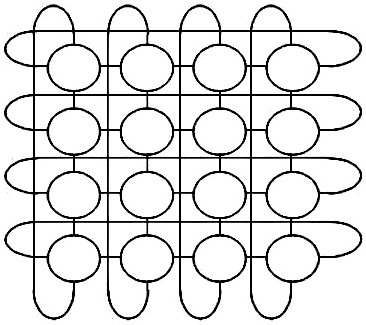

Figure. 2.Torus Topology

Charles E. Leiserson in [5] proposed Fat Tree topology. Fat tree topology showed traits like high scalability and efficiency. In addition, it proved that fat tree is the most efficient network architecture in terms of the amount of hardware used. In this architecture, the clients are at the leaf level as shown in the figure 3 (a). The main disadvantage of fat tree is that it includes large number of switches for a fewer IPs, which consumes most of the chip area.

The Butterfly network [6] is a multistage NoC topology, which uses the general MIN structure of the FAT Tree with the exception that the clients are present at the both end of the tree. Hence, it contains double the number of clients using same number of switches as fat tree that reduces the chip area consumption. The routing in butterfly network is unidirectional. Hence, it uses only deterministic routing. Advantages of the network includes lower network diameter i.e. the order of $N/2\log_2 N$, where N is the number of clients and it is highly scalable. The major disadvantage of butterfly topology [3] is that it does not have path diversity and long wirings.

SPIN (Scalable Programmable Integrated Network) for NoC packet switched network was proposed by Guerrier and Greiner [7]. The SPIN uses a fat tree architecture where every node has four children and every parent is replicated four times at any level. The size of the SPIN is governed by (NlogN)/8 and the number of switches is given by (3N)/4 where N is the number of IP cores. The figure 3 (c) explains the architecture with N=16.

The authors of [1] have performed a partial group based routing on the MIN Fat tree structure and compared the result with a regular MIN Fat tree routing [1]. The partial group based algorithm provides the efficient routing than the regular routing. There, the authors explored a new group based routing for NoC topology.

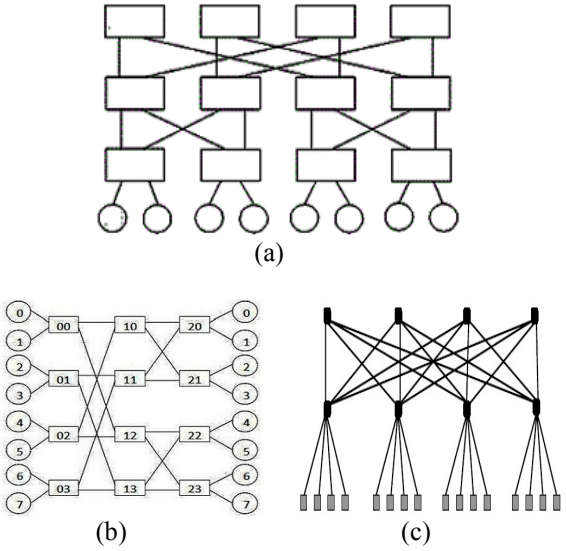

Figure 3: Various Network Topology for NoC (a) Fat Tree, (b) Butterfly (c) SPIN

A comparative study was being carried out by Sonal S. Bhople et al., in [10], where based on average delay of various topologies such as mesh, torus, octagon, spider BT, etc are compared. It is found that octagon; torus and spider had the lowest network delay. The table 1 provides his result for various topologies.

TABLE I. DIFFERENCE OF AVERAGE NETWORK DELAY [10]

Topology	Average Network Delay
Mesh	29
Torus	25
Folded Torus	26
Ring	32
Octagon	21
Spider	24
BT	59
BFT	40
SPIN	40

978-1-5090-2552-7/16/$31.00 ©2016 IEEE

The next few sections will describe our proposed topology along with group based routing and the results are analyzed.

III. Proposed Network Architecture

In this paper we proposed a network topology for NoC, by keeping in mind the advantages and also the limitations present in the earlier topologies. The proposed topology is designed in such a way that it can be easily scalable form 8 to 16, 32, 64, and so on number of clients. This new topology contains more than one communication path between any two clients, which reduce conflicts of packets destined for same destination and provides path diversity. Here the clients are placed at the two edges of the network. The basic network contains 8 clients using six switches as shown in the figure 4.1. For higher number of clients the network scales up as multiple groups as shown in figure 4.2.

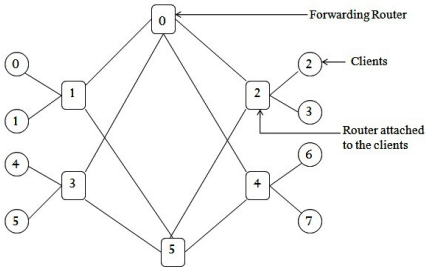

Figure 4.1 : Proposed Topology for 8 clients (Basic)

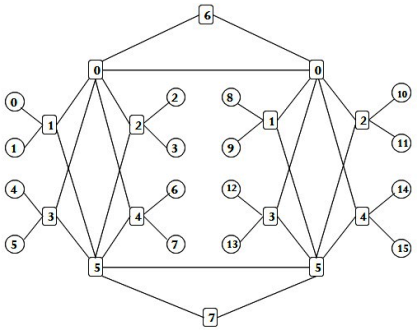

Figure 4.2 : Proposed Topology for 16 clients

The proposed topology has several advantages over the popular existing topologies like fat tree, butterfly and binary trees. As compared with fat tree, this new topology uses less number of switches for the same number of clients and it also provides path diversity which does not exist in butterfly topology. Unlike binary tree, this new topology does not suffer from any root node failure.

A. Switch Design

The switches or routers play the most important role as it performs the whole routing operations. The packets are forwarded from source to destination through one or more switches. In the proposed topology we use homogeneous switches in sense that each router in the topology is same in design and constructed of six ports. These ports are again interconnected to each other using input and output gates. These ports are responsible for handling the packets within the switches. Figure 4.3 explains the internal connections of the ports within the switch. All the switches are interconnected to each other. Switches in the edges are connected to the clients as shown in figure 4.1 and 4.2, the top and bottom switches are used as a gateway router to provide path diversity and scalability.

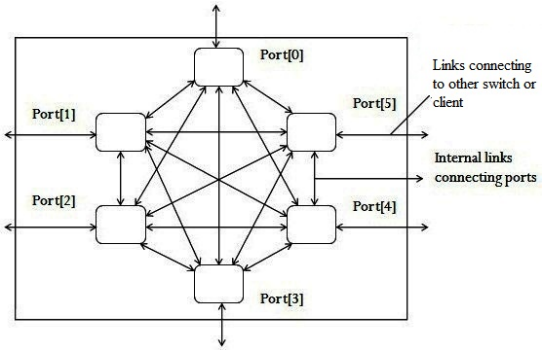

Figure 4.3 : Port connection inside a switch

B. Port Design

As mentioned above every switch has six ports connected with each other via bidirectional links with a fixed data rate. These ports have input and output gate to perform incoming and outgoing operation of packets. A port is composed with three components or modules: *Scheduler, Inport and Opcal* [1].The entire routing operation is done using these three modules. Figure 4.4 shows how these three modules are interconnected within a port.

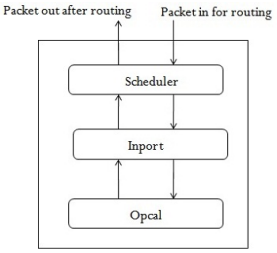

Figure 4.4: Internal view of a Port [1]

The *Scheduler* [1] is basically designed for handling new packets that are received from the other switch or client. When, it receives an incoming packets at any ports from an external link i.e other switch, it sends the packet to *opcal* via *inport* for routing decision. After the calculation of the routing, the packets will be sent out form the switch through the specific calculated port to the next switch. If a packet comes from one of the internal links that connects all the six ports inside a switch, *scheduler* just sends the packet out of the switch through proper output gate.

Inport [1] module just works as an interface between *scheduler* and *opcal*. It just forwards the packet to Opcal, coming from *Scheduler* for routing decision and after the routing decision has been made, it again forwards the message back to *Scheduler* from *Opcal*.

978-1-5090-2552-7/16/$31.00 ©2016 IEEE

Opcal [1] executes the primary part of the routing algorithm. It calculates the output link or port through which the packet needs to send. It decides whether the destination client is attached with the current switch or not and according to that, it calculates the output link. After calculating the output port, it sends the packet back to the *scheduler* via *inport*.

C. Client Design

Clients are considered as IP blocks that are attached to the switches. For our simulation purpose, we keep the architecture of the client as simple as possible. Clients in the network are identified by serial number from 0 to N-1, where N=8, 16, 32, 64, etc. This id also act as source and destination address. Each client has two sub modules, *sink* and *source* as shown in the figure 4.5 [1]. *Source* module only generates new packets and sends out from the client. *Sink* is responsible for receiving any incoming packets addressed to that client.

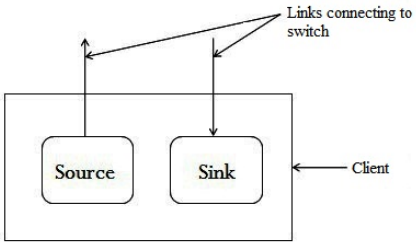

Figure 4.5: Client Architecture [1]

D. Routing Algorithm

Input: New message from external switch or client.
Output: Message forwards to the next switch or client depending on the destination address.
Procedure steps:

1) Input (message)
   a) Extract the source and destination client address.
   b) From the destination client address, calculate the destination group and the destination router.
2) Depending on the destination group, destination router and the destination client, do the following operations:
   a) If the current router's group id is equal to the destination group id, do the following:
      i) If the current router is the destination router, then deliver the message to the destination client attached to the router through the desired port.
      ii) If the current router is not the destination, message is then forwarded to the next router depending on the type of the current router.
         (1) If the current router is a forwarding router, calculate the output port from the below equation and forward the message through that port.
            (int) outport = [7 – (destination client id – 8 X destination group id)]/2;
         (2) If it is not a forwarding router, then the message is forwarded to the nearest forwarding router. If the link to the nearest forwarding router is busy, it is forwarded to the other forwarding router.
   b) If the current router's group id is not equal to the destination group id, do the following:
      i) If the current router is adjacent to the destination group, forward the message to the destination group through the link joining them. If the link is busy then the packet is forwarded to the higher forwarding router that joins the two groups.
      ii) If the current router is not adjacent to the destination group, forward the message to the higher forwarding router having connection to the destination group.
      iii) If the current router is the highest forwarding router joining source and destination group, then the message is forwarded to the next forwarding router towards the destination group.
3) End Procedure.

IV. Simulation Result and Discussion

We design and simulate our proposed network using OMNeT++, an open source simulator based on NED (Network Description) language. Using NED language we designed the network for 8, 16, 32 and 64 numbers of clients. The routing algorithm for the network is executed using C++. In the simulation process all the clients will act as a source which sends packets to one or two static or multiple randomly chosen destination clients using the function *intuniform(x, y);* $0 \leq x \leq y < N$, provided by OMNeT++. Here x represents lower bound and y represents upper bound of destination address. N represents the size of the network. Here we avoid self destination address by subtracting one from the destination address if the destination address is an odd number and adding one if the destination address is an even number.

A. Packet structure

Packet carries all the routing information with source address, destination address and also data or payload. In our proposed work, we use a packet that consists of five fields: packet id, source client address, destination client address, output port number and data or payload. Figure 5.1 shows the complete packet structure and describes as follows:

Packet Id	Source Id	Destination Id	Output Port	Data

Figure 5.1: Packet Structure.

- Packet Id (4 bytes): It is basically an integer number from 1 to n, where n is the number of packets generated in a client.
- Source Id (4 bytes): It is the id of the client from where the packet is send. It is also an integer number from 0 to N-1, where N is the network size.

978-1-5090-2552-7/16/$31.00 ©2016 IEEE

- Destination id (4 bytes): It is the id of the destination client same as source Id, where the packet needs to be sent.

- Output Port (4 bytes): This field defines the port id through which the packet needs to be sent from the switch. It is calculated in the switch and set in the output port field of the packet. This is also an integer type variable.

- Data (4 bytes): In general, data or payload field carries the information needed to send. For our simulation process, we keep this field just as an integer number.

B. Simulation Result

Here we record the simulation result of the proposed network topology in terms of total number of events and average latency of packets. We also present the result in the form of histogram for different network sizes. In our simulation process every clients act as a source and each source generate only one packet of size 20 bytes. Depending upon the destination address generation technique (static or uniform random), the results are tabulated as shown in the below tables. Table II shows the average total number of events for different network size and table III shows the average latency of the network.

TABLE II. AVERAGE TOTAL NUMBER OF EVENTS FOR PROPOSED NETWORK

Network Size	***Average Total Number of Events***		
	Static Destination	***Random Destination***	***Accuracy***
8	133	125	100%
16	349	342	100%
32	956	909.5	100%
64	2529	2481	100%

TABLE III. AVERAGE LATENCY FOR PROPOSED NETWORK

Network Size	***Average Packet Latency in 10^{-8}sec***	
	Static Destination	***Random Destination***
8	11.95	6.75
16	33.65	10.48
32	151.57	23.79
64	1344.26	57.82

In our simulation process, we found that there is no packet lost, hence the accuracy is 100%. Here we observed that average latency increases with the increase of network size. Also, we found that the latency value for static destination address is greater than the randomly distributed destinations. This happens because in static destination mode all the clients send packets to only one or two destination clients. In case of number of events, for static destination mode it is also slightly higher than uniform random destination mode. The results are also presented as histogram in figure 5.2 and figure 5.3.

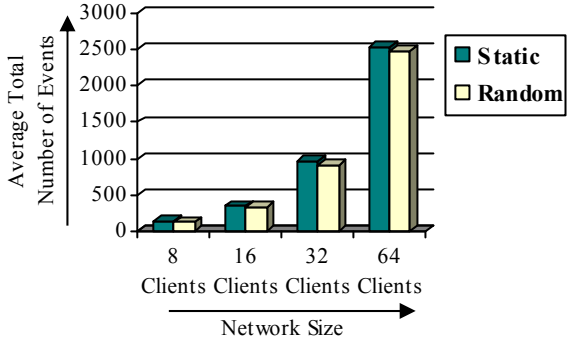

Figure 5.2: Total Number of Events for Proposed Network

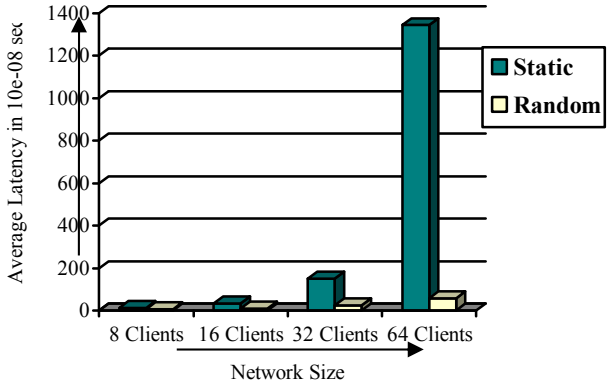

Figure 5.3: Average Latency for Proposed Network

V. COMPARISON WITH EXISTING NETWORK

First, we compared the proposed topology with Fat tree in terms of number of switches used in the network. The comparison is shown in the table IV and also shown as histogram in the figure 6.1. From the below figure we can clearly observe that the new topology is very much efficient in terms of use of hardware compared to fat tree.

TABLE IV. COMPARISON OF TOTAL NUMBER OF SWITCHES USE

Network Size	***Total Switches use***	
	Proposed Topology	***Fat tree***
8	6	12
16	14	32
32	30	80
64	62	192

978-1-5090-2552-7/16/$31.00 ©2016 IEEE

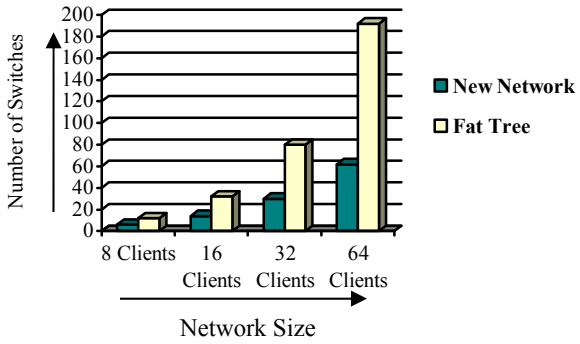

Figure 6.1: Number of switches used

We also compared simulation result of the proposed topology with the existing result of Fat tree, in terms of number of events presented by A. Biswas et al. in [1]. The comparison is done for static and random destination address, which is shown in the table V and VI respectively and also shown in the figure 6.2 and 6.3. The comparison shows that the proposed topology shows better result than fat tree.

TABLE V. COMPARISON OF AVERAGE NUMBER OF EVENTS FOR STATIC DESTINATION ADDRESS

Network Size	***Average Number of Events***		
	Proposed Topology	***Fat Tree (Packet Based Routing)***	***Fat Tree (Partial Group Based Routing)***
8	133	184	134
16	349	529	395
32	956	1421	1099
64	2529	3569	2796

TABLE VI. COMPARISON OF AVERAGE NUMBER OF EVENTS FOR RANDOM DESTINATION ADDRESS

Network Size	***Average Number of Events***		
	Proposed Topology	***Fat Tree (Packet Based Routing)***	***Fat Tree (Partial Group Based Routing)***
8	125	205	135
16	342	527	403
32	909.5	1433	1063
64	2481	3567	2756

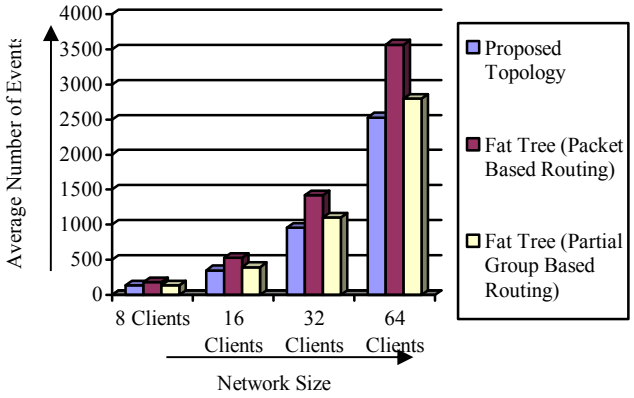

Figure 6.2: Average Number of Events for Static Destination Address

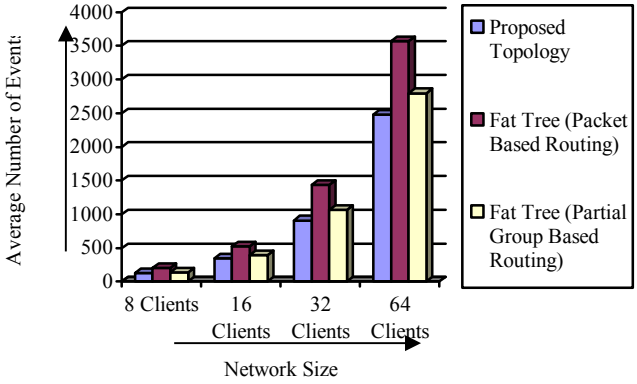

Figure 6.3: Average Number of Events for Random Destination Address

VI. CONCLUSION

Network-on-Chip (NoC) architecture was introduced to provide efficient and scalable communication compared to bus-based SoC system. It replaced the shared-bus communication with network centric on-chip intercommunication approach among different IP cores such as CPU, memory, DSP, etc. Here in this paper we have proposed a new network architecture for NoC that provide scalability and also eliminates some of the limitations presented in earlier NoC architectures. We show that this new topology uses less number of switches than Fat tree topology which reduces the hardware and chip area; also, it provides better path diversity than butterfly network topology and Fat tree. From, the simulation result we observed that the average latency value has increased with network size and it is higher for static destination address. We also compared the results of the new network with Fat tree in terms of number of events. From the comparison, we observed that our proposed network topology shows better results than Fat tree network over the same network condition.

978-1-5090-2552-7/16/$31.00 ©2016 IEEE

Acknowledgment

We would like to thank to the Department of Information technology of Assam University for providing us necessary material for completing the study of this topic. We would also like to thank Mr. Hriday Jyoti Mahanta for his support and help for completing this project work.

References

[1]. A. Biswas, H. J. Mahanta and Md. A. Hussain, "Implementing a partial group based Routing for Homogeneous Fat Tree Network-on-Chip Architecture", IEEE International Conference on ACCCT, May 2014, Ramanathapuram, Tamil Nadu.

[2]. S. Mahadevan, Tobias Bjerregaard, "A Survey of Research and Practice of Network-on-Chip", ACM Computing Surveys, Vol. 38, Article. 1, March 2006

[3]. Neetu Soni and K. Deshmukh, "A Survey on Different Topologies, ing Techniques, and Routing Algorithms for Network on Chip", International journal of Emerging Trends in Science and Technology, IJETST, Vol. 01, pages 380-387, May 2014.

[4]. Partha P. Pande, C. Grecu, Michael Jones, Andre Ivanov and Reseve Saleh, "Performance Evaluation and Design Trad-offs for Network-on-Chip Interconnect Architecture", IEEE transaction on computers, vol. 54, no. 8, pages 1025-1040, August 2005

[5]. Charles E. Leiserson, "Fat-Trees: Universal Network for Hardware Efficient Supercomputing", Computers IEEE Transactions on 100, no. 10, 892-901, 1985.

[6]. H. Moussa, O. Muller, Amer Baghdai, Michel Jezequel, "Butterfly and Benes-based on Chip Communication Networks for Multiprocessor Turbo decoding", 978-3-9810801-2-4/DATE07 © 2007 EDAA

[7]. P. Guerrier and A. Grenier, "A Generic Architecture for On-Chip Packet-Switched Interconnection", Proc. Design and Test in Europe (DATE), pp. 250-256, Mar. 2000.

[8]. Jie Chen, Paul Gillard and Chemg Li, "Performance Evaluation of three Network-on-Chip Architecture", First IEEE Conference on Communication on China (ICCC), Beijing, pages: 91-96August 2012.

[9]. S. Kumar, A. Jantsch, J. Soininen, M. Forsell, M. Millberg, J. Oberg, "A network on chip architecture and design methodology," Proc. IEEE Computer Society Annual Symposium on VLSI, Apr. 2002, pp. 105–112, doi: 0.1109/ISVLSI.2002.10168850973 – 5658, January 2012.

[10]. Sonal S. Bhople, M. A. Gaikwad, "Comparative Study of Different Topologies for Network on Chip Architecture", International Journal of Computer Applications (0975 - 8887), "Recent trends in Engineering technology" 2013.

978-1-5090-2552-7/16/$31.00 ©2016 IEEE