Received: October 26, 2016

INASS

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Routing for Center Concentrated Mesh

Akash Punhani[1]* Pardeep Kumar[1] N Nitin[2]

[1] Department of CSE& IT, Jaypee University of Information Technology, Waknaghat, Solan, H.P. India
[2] Department of CSE& IT, Jaypee Institute of Information Technology, Sector 62, Noida, U.P. India
* Corresponding author's Email: akashpunhani@gmail.com

Abstract: As the number of cores increases this affects the performance of the mesh and leads to investigation of new topological concept that is center concentrated Mesh. The topology designed seems to be efficient but the routing algorithm seems to contradict the objectives. In this paper we propose a new routing algorithm which is more time efficient in comparison to the center concentrated mesh (CCM) routing in terms of QoS Parameters and space efficient in comparison of table based routing algorithm. The main idea behind designing the new routing algorithm is due to the identification of hot spot effect at the center nodes created by CCM routing algorithm. Proposed routing algorithm is a hybrid of XY routing and CMM routing. Main objective of the proposed routing algorithm is to reduced hop count and improve the performance in quality of service parameters. It has been observed that the proposed routing algorithm has shown improvement in all the quality of service parameters. This improvement ranges from 10% - 30% as shown by the simulation results. The hop count analysis also provides the improvement of 7%. All the results prove that the proposed routing algorithm is a better substitute to CMM routing algorithm. The routing has also been compared to odd-even algorithm and was found to be slightly better.

Keywords: Mesh interconnection network, Routing algorithm, Latency, Throughput.

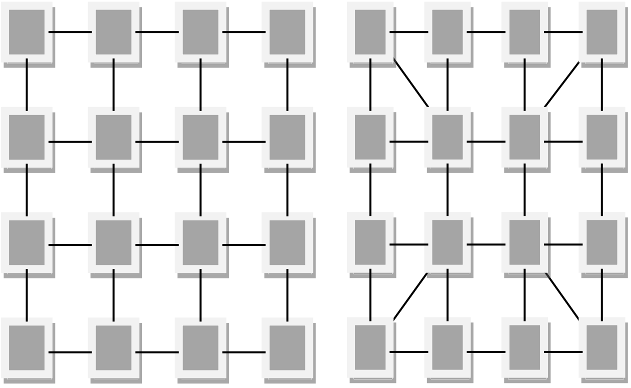
1. Introduction

System on Chips (SoC) consists of large number of cores that may be used for the general computing or for the specific computing [1]. These computing cores need to communicate with each other. The communication between them depends upon communication network which can be simple as bus to variant likes Network on Chips (NoC). The various topologies have been proposed for NoC are based on packet communication [2] – [6]. Tile size based mesh topology comes out to be more efficient in terms of layout and addressing the packets. Topology alone is not responsible effective communication the routing algorithm used [7], [8]; also play a vital role in the communication. If the routing employed is a source based, then it requires huge amount of space and the distributed routing are based on Finite state machines which may require modification according to the faults. The various variants have been proposed in past for the performing faster in comparison the mesh the variant like torus, x-torus, xx- torus, x mesh, sd torus, C^2 Mesh and C^2 torus, EMC^2[9]– [16] have been proposed in the past. The C^2 Mesh and C^2 torus and CC torus topologies have been based on the concept of center concentration [12]–[14]. The analytical parameters used for the analysis of these topologies are degree, diameter, bisection bandwidth seems to be good but traffic analysis of the network could not support the results [17], [18]. This implies the topology designed is good but still less focus has been projected on the routing algorithm used. The CCM routing has been proposed by the authors but the routing algorithm used has been directing all the packets to the center nodes and will require high performance at the center node there by creating congestion in the network. In this paper, the main focus is to identifying the bottlenecks created by the routing algorithm and improves the performance of NoC effectively. The concise objective of this paper can be described by the three points.

DOI: 10.22266/ijies2017.0228.10

1. Propose an oblivious routing algorithm based on the simple hardware's.
2. The routing algorithm proposed should be efficient on various Quality of Service (QOS) parameters.
3. It should utilize the extra links provided in the topology and should not have complex requirement of computing and space.

Before we move further the detailed description of the C^2 Mesh topology is needed. The C^2 Mesh topology is derived from the Mesh topology by connecting the corner nodes to the center nodes. The 4X4 mesh and C^2 mesh have been described in the figure 1(a) and figure 1(b) respectively. From Figure 1 it can be easily inferred that C^2 Mesh is having less diameter in comparison to that mesh topology. Diameter of a topology can be defined as the distance between the two farthest nodes in the topology. The two corner nodes in both the topology seems to be the farthest node, the diameter in the case of 4X4 mesh comes out be 6 but in the case of C^2 mesh it comes out to be 4. The same can be done by the table based approach but the space complexity of the table based routing is more in comparison to the proposed routing. The paper is divided into 6 sections. In section 2, the detailed analysis of the existing routing algorithm is done and highlights the need of designing a new routing algorithm for the center concentrated mesh topologies. The section 3 details about the proposed routing and also describes the routing algorithm in details. The section 4 gives the details of the test bed used to perform the experiment, so that the environment can be regenerated to perform the analysis in the same environment for the future work taking the support of the results discussed in our paper. In section 5 the result obtained on the various parameters are discussed. In section 6 we conclude our paper.



a. Mesh 4X4 b. C^2 Mesh

Figure.1 Describing the Mesh and C^2 Mesh topology

2. Existing Approach

First of all, we start our discussion with the existing method of routing that is the CCM routing algorithm as described in the [12]. Actual routing algorithm for the center concentrated mesh is based on dividing the mesh into 4 sub-meshes. This leads to the 3 categories of nodes in the center concentrated mesh.

1. Corner nodes: The corner nodes are the nodes that are at the extreme of the mesh topology, as there are 4 corners so there are only four corner nodes.

2. Center nodes: As the mesh can have *NXN* nodes this *N* can be odd or even this leads to varying number of center nodes that can be 1 or 4.

3. Elementary nodes: All the remaining nodes can be classified as the elementary nodes. This can be represented by the mathematical formulation as described in the equation below:

$$Node_{Elemetary} = Total\ Nodes - \left(Nodes_{Corner} + Nodes_{Center}\right) \quad (1)$$

According to the proposed algorithm [12], to route a packet from particular source node to a destination node. Initially the classification of the sub-mesh is done keeping in view that every node of the sub-mesh can get one center node. In case of odd number of nodes this center comes out to be common.

After identifying center of source and destination, packet flow can be described with 3 sub sources and 3 sub destinations. These sub sources and destinations are

1. Source to sub-mesh center (*SC*): The packet is moved from the source to sub-mesh center node.
2. Source sub mesh center to the destination sub mesh center (*MM*): The packet is switched from the source sub-mesh center to the destination sub-mesh center.
3. Sub mesh center to destination (*CD*): The packet is send from the destination sub mesh to the destination node.

The total number of hops required can be described the equation given below:

$$H_{Total} = H_{SC} + H_{MM} + H_{CD} \quad (2)$$

Here in above equation H_{Total} is the total hop counts required. The H_{SC}, H_{MM} and H_{CD} are Hop counts required for message to traverse from the source to center, Sub mesh to another Sub mesh center and from center to the destination respectively. The routing algorithm has certain question to answers:

1. Is the path selected from the source to destination is always minimum?
2. If the path selected by the routing is not shortest path, is it positively balancing the load?

To answer these questions, consider the center concentrated mesh described in the Figure 2. Let the source be node 1 and the destination be considered as node 2; Now according to the algorithm the packet will be first routed from source to the center this will be done in a single hop, now again node 6 which is the center will identify the correct sub mesh and transfer the packet to that center. Here packet belongs to the same sub mesh 1 hence it will require zero hop. In third phase center will send the packet to the destination and it will require 1 hop. According to the equation H_{Total} will be the summation of all the three, i.e. H_{Total} = 1+0+1 =2.

Ideally it can be routed directly with the link between 1to 2 in single hop. If we select the source as node 2 and the destination as node 3, i.e. they are belonging to the different sub meshes then again H_{SC} =1, H_{MM} =1 and H_{CD} =1. Therefore, H_{Total} comes out to be 3. But from the figure it can be observed that there is a direct link between 2 and 3, so the H_{Total} can be 1. This answers our first question that the route selected by the routing algorithm is not optimal.

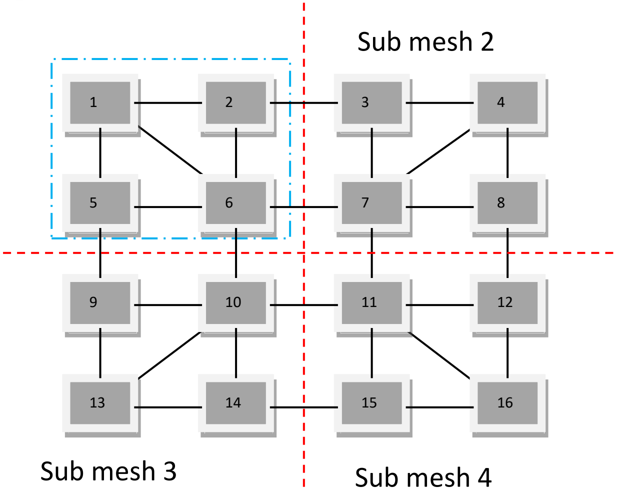


Figure.2 Describes the various sub-mesh in *4X4* C^2 Mesh

To answer the second question, we have to identify the incoming request on the center in any of the sub mesh. The sub mesh 1 will be having *N/2 X N/2* nodes when the mesh is considered to have even numbers of nodes in a row and column.

As we know that every node in sub mesh will first send the packet to the center node that is to be routed to the destination. So the total load on the center will be *N/4 - 1* request. This means that any communication in the mesh will be using only four center nodes for communication and will create a hotspot on the center nodes. This can be reduced to some extended if we can route the adjacent packets without the intervention of the center nodes. Hence this approach will give as better results in comparison to the existing routing algorithm for C^2 mesh.

Another routing approach that can be used for routing the packets is based on the routing tables maintained at each node. These routing tables are based on the shortest path between the two nodes. The similar type of approach has been addressed in the article [13]. In this approach each node has its own routing table that can be used to route the packet to other nodes the size of table increases as the number of nodes increases in the topology. This has been addressed by researchers that the routing based on table will require more memory unit and this will in turn increases the size of router and chip. Even though the best output generated from the table based approach is always same as that generated by the algorithmic approach.

As the topology is being derived from the mesh topology, so the odd-even routing is being used for comparing with the proposed algorithm. In this case also as the routing algorithm is applied it will uniformly distribute the packets without using the extra links present in the topology. The assumption for routing the packet is based on the dimensions which further may lead to adaptive routing.

3. Proposed Algorithm

The Algorithm 1 given below describes the proposed routing that is named as Modified Center Concentrated Routing (MCCM) Algorithm. The proposed routing algorithm has been designed in considering the design on the simple mesh router which is supposed to be simple in design and is not advice to have huge amount of memory and also the computing power of the router is limited to some extent.

Algorithm. 1 Modified Center concentrated Mesh Routing (MCCM) Algorithm

```
INPUT:   Coordinates of Source and Destination (S,D)
Output:  Port Number for Next destination
Step 1:  Evaluate Center Node Coordinates
         If n is even then
         C1← (n/2 -1, n/2-1)
         C2← (n/2, n/2-1)
         C3← (n/2 -1, n/2)
         C4← (n/2 , n/2)
         else
         C1 ←C2←C3←C4←(n/2 , n/2)    // There will be only single center node
         End if
Step 2:  E1←(0,0)
         E2← (0,n-1)
         E3←(n-1,0)
         E4←(n-1,n-1)
Step 3:  Get mesh id of source and destination say s,d
Step 4:  Dxy (sx,sy,dx,dy)  (dx-sx)+(dy-sy) // macros defintion
         Dis← Dxy(Sx,Sy,Dx,Dy)
         DCMM ← Min(Dxy (S,Cs), Dxy (S,Es)+1) +Dxy(Cs, Cd)+ Min(Dxy(D, Cd), Dxy (S,Ed)+1)
Step 5:  If(Dxy < = DCCM)
         Port =XY(S,D)
         Else
         Port =CCM(S,D)
         End If
Step 6   END
```

In the proposed algorithm *S* an *D* are the coordinates of the source and destination, Step 1 is used to evaluate the center coordinates *C1, C2, C3, C4*, Step 2 is used to evaluate the corner coordinates *E1- E4*. In step 3 mesh id is identified using the logic described in algorithm 2. The distance is evaluated by using the macros D_{xy}. This macro function is used to evaluate the length of path generated by *CCM* routing algorithm. Now based on the values either *XY* routing or *CCM* routing is selected. The algorithm for mesh id is given below, and the *XY* routing is the most popular routing algorithm so the detail algorithm can be identified from the various sources [8]. The *CCM* routing has already been proposed by the [12]. The port number is returned by the routing algorithm, *XY* routing algorithm is the routing algorithm for mesh topology which requires only 5 port router so it will return the values from 0-4 and the *CMM* routing is supposed to use the center and corner link which is maintained by an extra link that makes the range of the result from 0-5. If 5 is reported as the port, then it implies that the packet should be routed to the center or to the corner node based on the position of the current node.

Algorithm. 2 For Sub-Mesh identification for the Specific node

```
INPUT:    Coordinates of node(x,y)
Output :  Port Number for Next destination
Step 1:   C← N/2
Step 2:   If (x>0 and y>0 and x<=C-1 and y<=C-1 then
          ID ←1
          End if
Step 3:   If (x>C-1 and y>0 and x<=N-1 and y<=C-1 then
          ID ←2
          End if
Step 4:   If (x>0 and y>C-1 and x<=C-1 and y<=N-1 then
          ID ←3
          End if
Step 5:   If (x>C-1 and y>C-1 and x<=N-1 and y<=N-1 then
          ID ←4
          End if
Step 6    END
```

4. Experimental Setup

To test the performance of the proposed routing algorithm the OMNet++ [19], [20] simulator has been used along with the HNOCS [21] package. The various parameters involved while testing the routing algorithm have been provided in the table 1. These parameters along with the various load factors[22] have been used to test the performance of the network. To generate the random rate of traffic the flit arrival delay has been varied from 80 ns to 8 ns. The parameters under the test are end to end latency, sink bandwidth and loss probability.

Table 1. Describes various parameters used during the experimental setup

S. no.	Parameter Name	Parameter value
1	Rows	4
2	Columns	4
3	Simulation time	2ms
4	Warm up time	240 ns
5	Message length	4 Packets
6	Packet length	8 Flits
7	Flit Size	4 Bytes
8	Traffic type	Uniform
9	Maximum Queued packet	4
10	Channel Bandwidth	8 Gbps

The results are obtained for each node, but to plot the performance an average of the mean values have been taken into the consideration this is given by the expression described in the equation 3

$$Average(P_x) = \frac{\sum_{i=1}^{n} mean(P_{xi})}{n} \qquad (3)$$

P_x here represents the parameters like sink bandwidth, end to end latency and P_{xi} is used to represent the values obtained at particular node. Except of using the simulator the hop count has also be calculated for all the routing algorithms based on the way the packets are being routed from source to the destination.

5. Results and Discussion

The Results have been classified into the four categories.

5.1 End to End Latency

The end to end latency is defined the time require by the packet to reach from source to the destination[23], [24]. This latency is the attribution of various delays that occurs during the flow of packet from source to destination. Latency actually not only depends upon the topology, it also depends upon the factors like routing and flow control mechanism applied. In our case the topology for the observation is the same and flow control mechanism is FIFO. So keeping all other factors constant the effect of routing algorithm on the latency is being studied.

The latency can be described as the attribute of the head latency T_h and serialization latency T_s and queue delays T_q. The total latency T can be given by the equation 4

$$T = T_s + T_q + T_h \qquad (4)$$

Head latency T_h is described as the sum of router delay t_r and propagation delays t_p. As the network is having multiple routers this can be estimated by the hop counts H. So T_r can be represented by the equation below

$$T_r = H \times tr \qquad (5)$$

In case of simulation the exact latency is found by stamping the message with the injection time from the source; let it be represented by T_{source} and the message when reach the sink it timestamp is recorded as T_{sink} .

$$T = T_{\sin k} - T_{source} \qquad (6)$$

The total latency is described by the Equation 6. Now as there are multiple source and multiple sinks so to get average latency we will use the Equation 3. For a good routing algorithm, the average latency of the network should always remain the same or less than that of other routing algorithms under comparison. The graph in the figure 3 described below shows the comparison of the four routing algorithms and from the graph it can be concluded that the proposed routing is better in comparison to that of existing routing algorithm on all the existing cases of load factor that may be 0.1 to the maximum load of 1.0. At the small load factor from 0.1 to 0.5 the difference between the routing algorithm latency is small but as the load increases from 0.6 to 1.0 the latency of the CCM routing algorithm has increased at faster rate in comparison to the proposed routing algorithm and the latency of the odd-even routing has shoot up a lot.

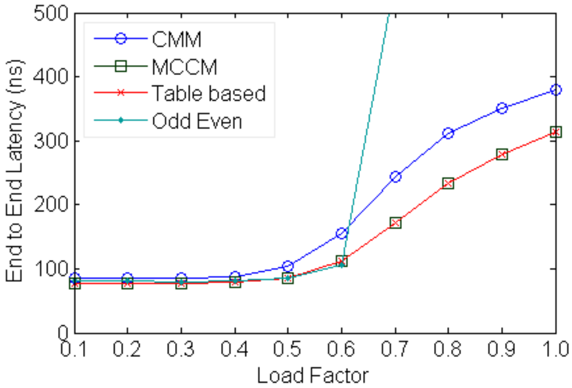


Figure 3. Describes the End to End latencies at different Load factors

5.2 Bandwidth

It is defined as the rate at which the packet is transferred from source to destination. The bandwidth of the network depends upon the bisection bandwidth of the network. The bisection bandwidth of the network can be defined as product of bisection width of the topology and channel bandwidth[8], [25]. This can be described by the equation 7 described below

$$B = B_w \times C_b \qquad (7)$$

In the equation 7, B is representing the Bisection bandwidth of the topology, B_w represents the bisection width and C_b is used to represent the channel bandwidth. Bisection width of the topology can be defined as the minimum number of nodes that is to be removed such that the network gets divided into 2 equal halves. If the network is having odd number of nodes, then this can be done on the basis of approximation. In simulation the observation of the sink bandwidth is done on the basis of the number of packets received at the sink. This is then converted into the bytes by multiplying it with packet size (*P*) and flit size (*F*). The conversion can be easily done using the equation 8

$$Data\ \mathrm{Re}ceived = N \times P \times F \tag{8}$$

The Data received is then converted into rate by dividing it with the Time T (difference of simulation time (T_{sim}) and warm up period (T_{warm})) as described in the equation 9 and 10.

$$T = T_{sim} - T_{warm} \tag{9}$$

$$B = \frac{Data\ \mathrm{Re}ceived}{T} \tag{10}$$

As we know that for a good routing algorithm the sink bandwidth should increase in comparison to that of existing routing algorithm. The same can be observed from the figure 4 described below that the sink bandwidth of the MCCM have improved at higher loads that is 0.5 to 1 the table based routing was almost same as the MCCM but the odd even routing algorithm has shown slight drop in the performance.

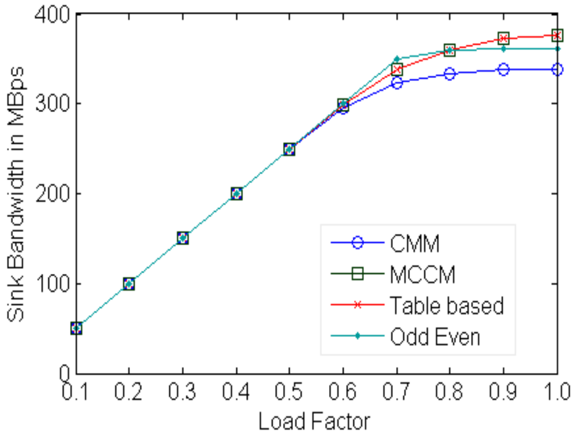


Figure 4. Sink Bandwidth at various Load factors

5.3 Loss Probability

The Loss probability is defined by the simulation on the basis of the number of generated packets P_{gen} and number of packets queued P_{queue}. Loss probability can be given by the relation below in the equation below[21]:

$$P_{loss} = 1 - \frac{P_{gen}}{P_{queue}} \tag{11}$$

The detailed analysis of the routing algorithms based on the simulation for the various load factors have been plotted in the figure 5. The smaller the loss probability better is the routing algorithm, the same is shown by the figure 5, as the network start to get loaded the loss probability of the CCM routing algorithm starts to increase at a faster rate in comparison to the proposed routing algorithm. The change has been observed from the rate of 0.5 and has significantly increased up to the load factor of 1.0. The factor is trending towards 1 for odd even routing algorithm.

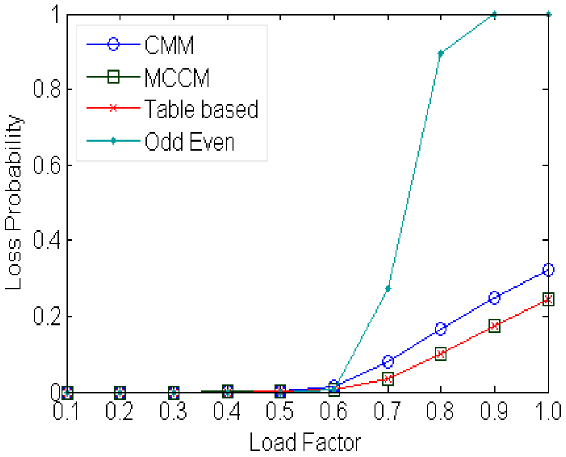


Figure 5.Described the loss probability at the various load factors

5.4 Hop Count Analysis

It is defined as the number of links that are to be used to traverse form the source to destination. It is very important factor in the analysis of the topology and the routing algorithm used. Topological analysis can be done easily using the Dijkstra's Algorithm[26] but in case of routing algorithm hop count analysis is done on the basis of the routing algorithm. As in this paper the comparison is done for the four routing algorithm and described in the figure 6, the routing algorithm with the small or same hop count will be considered as better. To do the analysis, the topology edges are assigned a weight of 1 unit. The figure 6 shows the comparison of the hop count using the source as the first node and other node as the destination node. On the X axis the node number are represented and on Y axis the hop count to particular destination is shown.

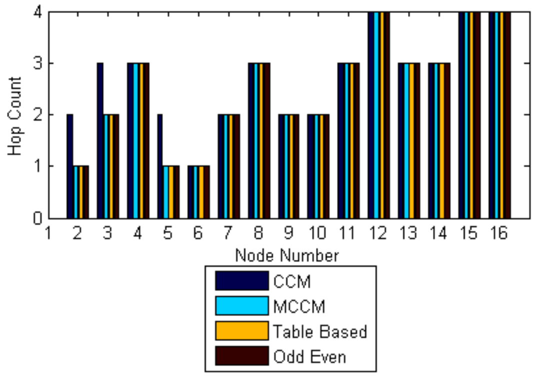


Figure 6. Describing the Hop Count from source node 1 to various other nodes

The average hop count of each node behaving as source to every other node has been described in the figure 7 below.

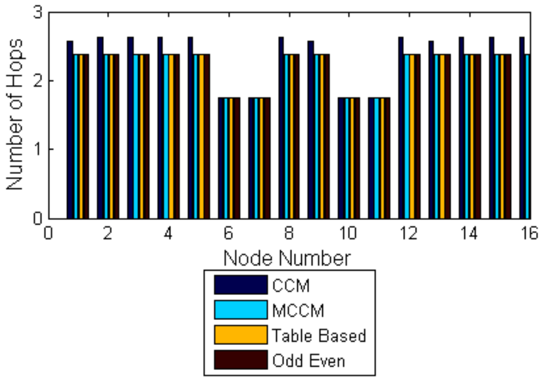


Figure 7. Describing the average hop count of the considering each node as source to other nodes

The table based routing algorithm and the MCCM routing algorithm have been compared for the memory requirement of the two algorithms The memory consumption of the MCCM algorithm is always constant and is always fixed so it can be referred as O(c). The memory consumption for table based routing increases with O(n^2). Similarly, MCCM routing proposed is similar to that of the CCM they both have the constant space and time complexity but the main advantage of MCCM routing is that it uses the minimal hops whereas CCM on other hand focus on first sending the data to the centre node this makes the extra hops and as the number of hops increases it will increase the latency and creates the large queues of packets there by reducing the performance.

6. Conclusion

The analysis of the routing algorithm has proved the importance of the routing algorithm in NoCs. The various parameters that were under observation have shown improvements in the performance. Results are generated for the various load factors but as the load factor increases the performance of MCCM starts to increase and the maximum improvement can be observed at the load factor of 1. This gives identification that the topology has not been saturated at the load factor 1 and the load has been evenly distributed in comparison to that in the case of CCM routing algorithm. Still we can see there is a tough competition between MCCM and the table based routing algorithm but the memory requirement shows that it is better to use MCCM in comparison to table based routing algorithm. It has been identified that the bandwidth has increased by 10% in comparison to CMM. The latency has also reduced by 21% and loss probability has improved by 30% in comparison to CMM routing. Based on the analytical analysis the hop count for the 4X4 topology has reduced by 7%. This all suggest that MCMM is the better substitute for the CMM routing. The odd-even routing which is also giving the good bandwidth results but the latency of the network has made sharp increase makes it infeasible to use based on the dimensions. The odd even routing is better when they are supported by hardware stress values but we are designing routing algorithm for the simple routers. In future we are going to study in detail comparison of including the heterogeneous link in the proposed topology and the relation to balance the load in the topology. The stated routing algorithm can be improved for the C^2 Torus also. The proposed routing algorithm is FSM based routing and does not require any extra memory to store the tables. There by the reducing the space and area.

References

[1] Mejia, M. Palesi, J. Flich, S. Kumar, P. Lopez, R. Holsmark, and J. Duato, *Region-Based Routing: A Mechanism to Support Efficient Routing Algorithms in NoCs*, Vol. 17, No. 3, pp. 356–369, IEEE Trans. Very Large Scale Integr. VLSI Syst, Mar. 2009.

[2] J. Flich, A. Mejia, J. Duato, and D. Valencia, “Region-Based Routing : An Efficient Routing Mechanism to Tackle Unreliable Hardware in Network on Chips”, In: *Proc. of First*

International Symposium on Networks-on-Chip (NOCS'07), IEEE , pp. 183–194, 2007.

[3] P. Guerrier and A. Greiner, “A generic architecture for on-chip packet-switched interconnections”, In: *Proc. of the conference on Design, automation and test*, Europe, pp. 250–256, 2000.

[4] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Öberg, K. Tiensyrjä, and A. Hemani, “A network on chip architecture and design methodology”, In: *Proc IEEE Computer Society Annual Symposium on VLSI*, pp. 105–112, 2002.

[5] J. Balfour and W. J. Dally, “Design trade-offs for tiled CMP on-chip networks”, In: *Proc. of the 20th annual international conference on Supercomputing*, pp. 187–198, 2006.

[6] E. Bolotin, A. Morgenshtein, I. Cidon, R. Ginosar, and A. Kolodny, “Automatic hardware-efficient SoC integration by QoS network on chip”, In: *Proc. 11th IEEE International Conference on Electronics, Circuits and Systems, ICECS 2004*, pp. 479–482, 2004.

[7] W. J. Dally and B. Towles, “Route packets, not wires: on-chip interconnection networks”, In *Proc. of Design Automation Conference*, pp. 684–689, 2001.

[8] W. J. Dally and B. P. Towles, *Principles and practices of interconnection networks*. Elsevier, 2004.

[9] L. Yu-hang, Z. Ming-fa, W. Jue, Li-min Xiao, and G. Tao, “Xtorus: An Extended Torus Topology for On-Chip Massive Data Communication”, In *Proc. 2012 IEEE 26th International of Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW)*, pp. 2061–2068, 2012.

[10] W.H. Hu, S. E. Lee, N. Bagherzadeh, Z. Xiao-jing, H. Wei-wu, M. Ke, and Z. Long-bing, *Xmesh: a mesh-like topology for network on chip*, Netw. Chip Archit., p. 14, 2008

[11] Y. Wang, H. Du, and X. Shen, *Topological properties and routing algorithm for semi-diagonal torus networks*, J. China Univ. Posts Telecommun., vol. 18, no. 5, pp. 64–70, 2011.

[12] L. K. Arora and Rajkumar, “C^{2}Mesh”, In: *Proc of IEEE 3rd International Advance Computing Conference (IACC)*, 2013, pp. 282–286, 2013.

[13] S. Yadav and C. R. Krishna, “CCTorus: A New Torus Topology for Interconnection Networks”, In: *Proc Int’l Conference on Advanced Computational Technologies & Creative Media (ICACTCM2014)*, pp. 14–15, Aug, 2014.

[14] Manish Bhardwaj, *C^2 Torus New Interconnection Network Topology Based on 2D Torus*, Am. J. Netw. Commun., vol. 4, no. 3, pp. 1–4, 2015.

[15] A. Punhani, Nitin, and P. Kumar, “A modified diagonal mesh interconnection network,” In: *Proc. 2014 Annual IEEE India Conference (INDICON)*, pp. 1–6, 2014.

[16] R. Vaish, U. Shrivastava, and Nitin, *On a deadlock and performance analysis of ALBR and DAR algorithm on X-Torus topology by optimal utilization of Cross Links and minimal lookups*, J. Supercomput., vol. 59, no. 3, pp. 1252–1288, 2012.

[17] A. Chauhan, A. Punhani, and Nitin, “EMC2Mesh”, In: *Proc. of 2015 Annual IEEE India Conference (INDICON)*, pp. 1–5, 2015.

[18] A. Chauhan, A. Punhani, and Nitin, “Comparative analysis of traffic patterns on centre connected topologies based on burton normal form”, In: *Proc. of TENCON 2015-2015 IEEE Region 10 Conference*, pp. 1–6, 2015.

[19] A. Varga and others, “The OMNeT++ discrete event simulation system,” In: *Proc. of the European simulation multi conference (ESM2001)*, vol. 9, p. 65, 2001.

[20] A. Varga, *OMNeT++*, In Modelling and Tools for Network Simulation, Springer, pp. 35–59, 2010.

[21] Y. Ben-Itzhak, E. Zahavi, I. Cidon, and A. Kolodny, “HNOCS: Modular open-source simulator for Heterogeneous NoCs”, In: *Proc. of 2012 Intl. Conf. on Embedded Computer Systems: Architectures, Modelling and Simulation, IC-SAMOS 2012*, pp. 51–57, 2012.

[22] Nitin and D. S. Chauhan, *Comparative analysis of Traffic Patterns on k-ary n-tree using adaptive algorithms based on Burton Normal Form*, J. Supercomput., vol. 59, no. 2, pp. 569–588, 2012.

[23] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, *Performance evaluation and design trade-offs for network-on-chip interconnect architectures*, IEEE Trans. Comput., vol. 54, no. 8, pp. 1025–1040, Aug. 2005.

[24] P. P. Pande, C. Grecu, A. Ivanov, and R. Saleh, “Design of a switch for network on chip applications”, In: *Proc of 2003 Intl. symp. on Circuits and Systems, 2003. ISCAS’03.*, vol. 5, pp. V–217, 2003

[25] J. Duato, S. Yalamanchili, and L. M. Ni, *Interconnection networks: an engineering approach*, Morgan Kaufmann, 2003

[26] S. Skiena, Dijkstra’s algorithm, *Implement. Discrete Math. Comb. Graph Theory Math. Read*. MA Addison-Wesley, pp. 225–227, 1990.