International Journal of Multimedia and Ubiquitous Engineering
Vol.10, No.10 (2015), pp.197-210
http://dx.doi.org/10.14257/ijmue.2015.10.10.21

An Extended Diagonal Mesh Topology for Network-on-Chip Architectures

Md. Hasan Furhad and Jong-Myon Kim

Department of Electrical, Electronic and Computer Engineering, University of Ulsan, De 93 Daehak –ro, Nam-gu, Ulsan 680749, Korea
{hfurhad, jongmyon.kim}@gmail.com

Abstract

This paper proposes an extended diagonal mesh (XDMesh) topology for network-on-chip (NoC) architectures to reduce latency and energy consumption for fast and low-power communication among remote nodes by including diagonal links in the network. In addition, we compare the performance of the proposed XDMesh with conventional state-of-the art topologies, including mesh, extended-butterfly fat tree (EFTI), and diametrical mesh, in terms of throughput, latency, energy consumption, and area overhead. Experimental results indicate that XDMesh outperforms the conventional topologies in terms of throughput and latency by varying the number of virtual channels and injection rate. Moreover, XDMesh achieves 46.28%, 35.29% *and* 19.37% *lower energy consumption than EFTI, mesh, and diametrical mesh topologies, respectively, and* 9.29%, 31.28%, *and* 15.23% *lower silicon area, respectively.*

Keywords: *Network-on-chip, system-on-chip, extended butterfly fat tree, diametrical mesh*

1. Introduction

Advances in semiconductor technology introduced the notion of the system-on-chip (SoC) paradigm, which is composed of many types of processors, on-chip memory, and I/O devices on a single chip. System-on-chip provides high flexibility and better performance than systems on separate chips, allowing designers to incorporate the functionality of a system onto a single chip by integrating tens or hundreds of intellectual property (IP) cores [1]. Whether homogenous, heterogeneous or hybrid, this sort of integration of several components into a single system gives rise to new challenges. With the impressive improvements involved, it is essential to have a flexible communication facility that can cope with the multipurpose programming of the cores [2]. Such systems should process data in real time, perform data transfers at high rates, support multiple functions and protocols for communications with standard wired and wireless interfaces, provide security and privacy, and handle time-to-market (TTM) pressures [3]. The emerging technology that targets such connections between heterogeneous cores is called an on-chip interconnection network, also known as a network-on-chip (NoC). Modularity and explicit parallelism can be incorporated into NoCs on future multiprocessor system-on-chip (MP-SoC) platforms [4].

NoC is motivated by conventional computer network systems, which provide an efficient communication model for MP-SoCs. This seeks to overcome the performance and scalability problems of shared buses and dedicated peer-to-peer links in multi-core SoCs by exploiting the concept of a packet-based communication network [5]. NoC connects a network of IP cores, memory, and programmable I/O through a network of routers or switches that communicate via addressed data packets using wormhole or virtual-cut-through switching. High scalability and versatility, low latency, high

ISSN: 1975-0080 IJMUE

throughput, high bandwidth, reusability, and lower power consumption for communication can be achieved with NoC-based architectures [6].

A number of topologies have been proposed in previous studies of NoCs, primarily derived from traditional data communication network topologies. Some examples of NoC topologies include mesh [7], extended-butterfly fat tree (EFTI) [8], and diametrical mesh (DiaMesh) [9]. Among these, the mesh topology is preferred by NoC designers due to its modularity and scalability. Since the two-dimensional mesh perfectly matches the 2D silicon surface, it is currently the most popular topology used for NoCs in tile-based architectures. Moreover, a 2D mesh topology can be partitioned into smaller meshes, which is a desirable feature for many parallel applications [10]. However, the conventional mesh topology of NoC architecture increases the number of hops when distant nodes communicate with each other, thereby increasing the system size. It requires higher energy consumption, and longer clock cycles to accomplish the communication. To solve this problem inherent in the mesh topology, we propose an extended diagonal mesh (XDMesh) topology that considers the location of nodes situated in remote regions by including diagonal links among these remote regions, which reduces the hop count, latency, energy consumption, and area cost.

The rest of this paper is organized as follows. Section 2 discusses related studies of NoC. Section 3 describes the proposed XDMesh topology and its routing algorithm. Section 4 presents the experimental results and analysis. Finally, Section 5 provides conclusions for the paper.

2. Background Information

2.1. State-of-the-Art NoC Topologies

In this section, we discuss some improved topologies, as shown in Figures 1 - 3, where the functional IP blocks (PEs) are denoted by circles, the nodes (or routers) are denoted by black squares, and links are bi-directional.

Kumar *et al*. proposed a 2-dimensional ($m \times n$) mesh network topology for NoC known as chip-level integration of communicating heterogeneous elements (CLICHE) [7]; it is depicted in Figure 1. In this topology, every router is connected to a specific resource, and the number of routers is equal to the number of resources, *i.e*., nodes. Each router is further connected to four neighboring routers.

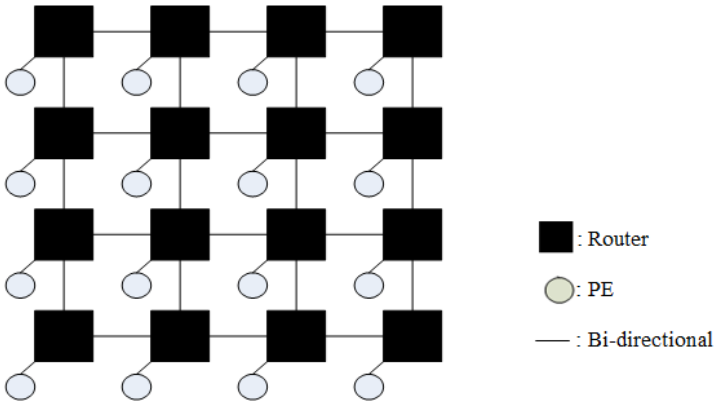

Figure 1. The Mesh Topology

Midia *et al*. [9] proposed a new mesh-like topology known as DiaMesh topology, which is depicted in Figure 2. This architecture consists of an $m \times n$ mesh of routers and 8 extra bypass channels. In addition, this architecture includes four sub-networks, in which

the opposite corners of each sub-network can be connected by a diametrical link. According to [9], the number of extra links in DiaMesh is not increased by an increase in IP cores. The number of links is restricted to 8. The main objectives of adding 8 extra links are to reduce the network diameter and enhance the network performance.

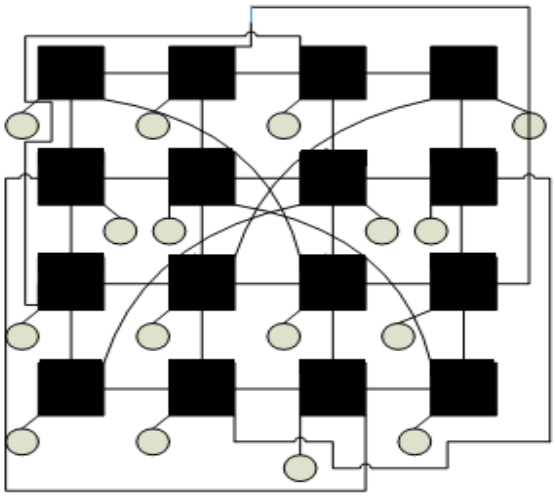

Figure 2. The Diametrical Mesh (DiaMesh) Topology

Hossain *et al*. [8] proposed an interconnection architecture by deriving the butterfly fat tree known as EFTI architecture, as shown in Figure 3. In this architecture, the PEs are placed at the leaves, while routers are placed at the internal nodes. Each node in the network is identified by a pair of coordinates (l, p), where l denotes the node's level and p denotes its position within that level. Each PE has one resource network interface (RNI) to connect with one of the routers. The RNI has one port, which consists of unidirectional links. Each router has four child ports, two sibling ports, and two parent ports.

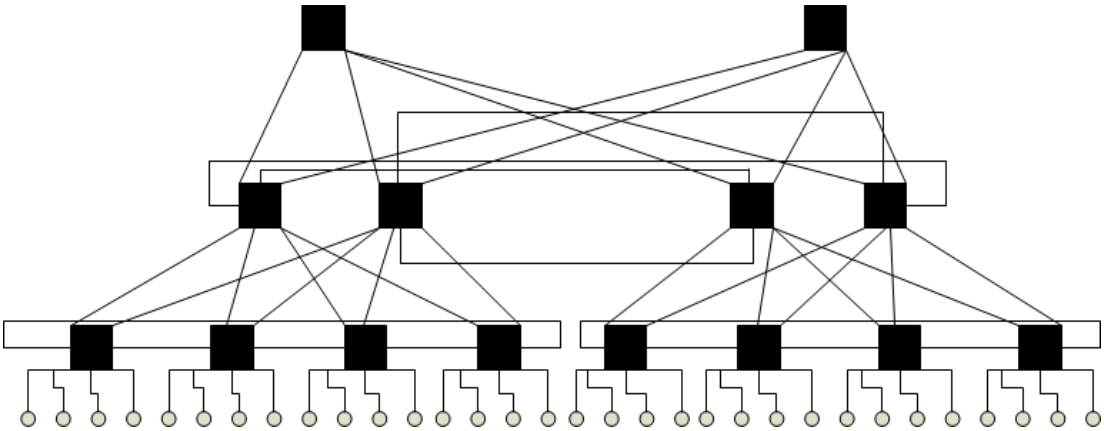

Figure 3. The Extended Butterfly Fat Tree (EFTI) Topology

2.2. Switching Strategies in NoC

Switching techniques are responsible for moving data from the input channel to the output channel of a router through the network. There are different types of switching techniques; these can be classified as packet switching, circuit switching, virtual cut-through switching (VCT), and wormhole switching (WH). Packet switching incurs communication delays and utilization of large buffer sizes. Circuit switching exhibits poor

resource utilization due to the need to establish a dedicated electrical path. Virtual cut-through switching exhibits packet blocking problems on busy output channels [11].

A packet is divided into flits in WH switching. A flit is a flow control digit which includes a constant number of bits that fits the storage resources in the network routers. There are three types of flits: head flit, data flit, and tail flit. A head flit carries the control and routing information of a packet; data flits carry the payload or data; and a tail flit contains data as well as the end of the packet information. To send data from a source to a destination, a head flit is transmitted first, followed by the data and tail flit. Routers check the head flit and lock the path for the rest of the flits of each packet. The path is unlocked by the tail flit. Thus, the flits in WH switching travel through the network in a pipelined fashion. The flits appear to move through the network like worms, and, therefore, the process is referred to as WH switching. This process can significantly reduce the average packet latency and overall buffer space requirements. Thus, WH switching is the most popular switching technique used in NoC.

2.3. Research Motivations

Through our intensive study of the mesh topology of NoC interconnection architecture, we observe that the main problem with conventional mesh topology is the large communication delay between remote arbitrary nodes, which increases the hop count. This increases the possibility of network congestion. In addition, it doesn't ensure routing flexibility due to the limitation of resources. A number of studies have attempted to address this shortcoming by adding application-specific links between distant nodes in the mesh [12], by leaving out intermediate nodes for express channels to reduce hop counts [13], by changing the network diameter in topologies related to the mesh, by adding 2D digraph families to increase network performance [14], or by inserting additional links into the mesh topology [15].

In the 2D mesh topology, the *network diameter* is the longest distance between two nodes; these are situated at diagonal corners of the mesh. In mesh-based topologies, the maximum packet delay solely depends on the diameter. It is observed that data packets suffer from long packet latencies due to the traversal of longer distances, which requires longer hops. Addressing this issue, we propose an XDMesh topology to reduce the communication delay and hop counts by including diagonal links between the nodes which are aligned in the diagonal direction. This also results in energy reduction, which is of prime importance for the interconnection architectures. To evaluate and analyze the performance of XDMesh, this study compares the proposed XDMesh with three conventional NoC topologies in terms of throughput, latency, energy consumption, and area overhead.

3. Proposed Topology and Routing Algorithm

3.1. Proposed Topology

Figure 4 shows the proposed XDMesh topology. The XDMesh is derived from a traditional mesh network by inserting diagonal links within the nodes situated in the diagonal direction. The formal definition of XDMesh is as follows:

XDMesh can be denoted by XD*mxn*. It consists of a set of nodes, $N = \{(x,y) \mid 0 \le x \le m-1, 0 \le y \le n-1\}$, where two nodes, $N_1 = (x_1, y_1)$ and $N_2 = (x_2, y_2)$, are connected to each other by a horizontal and vertical link if their distance is equal to one, according to the Euclidean distance. In addition, there will be a diagonal connection between them if the following conditions are satisfied:

If x_1 and y_1 are both even or odd simultaneously and the nodes lie in the diagonal direction between $(0,0)$ and $(m-1,n-1)$, then $\{(x_1 = x_2 + 1), (y_1 = y_2 + 1) \}$ or $\{(x_1 = x_2 - 1), (y_1 = y_2 - 1)\}$. If x_1 or y_1 is even, the other is odd, and the node lies in the diagonal

direction between *(m-1,0)* and *(0,n-1)*, then *{($x_1 = x_2 + 1$), ($y_1 = y_2 - 1$) }* or *{($x_1 = x_2 - 1$), ($y_1 = y_2 + 1$)}*.

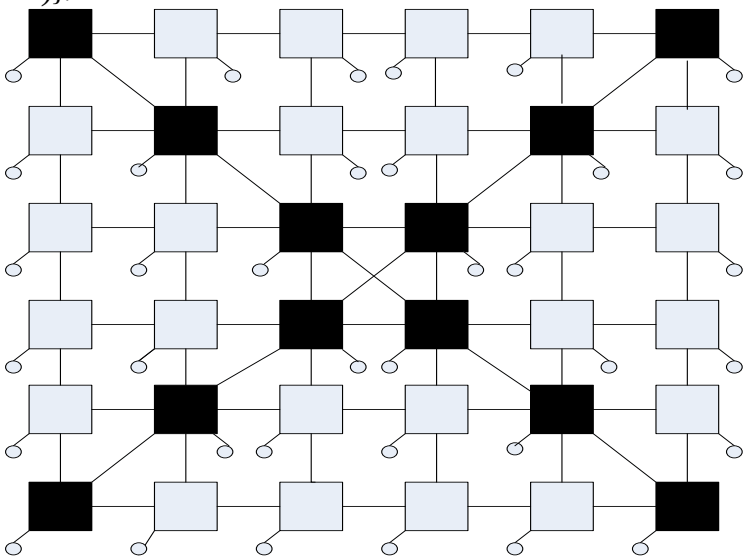

Figure 4. An XDMesh Topology

The nodes situated at the corner positions are the farthest nodes, and they are known as *remote nodes*. The XDMesh topology provides faster data communication between these *remote nodes* by its router architecture. Each router in the network has a maximum of six ports, comprising a DEMUX instead of a crossbar switch. This approach reduces the cost on a good scale, compared to other topologies. The block diagram of the XDMesh router architecture is shown in Figure 5.

From Figure 5, it is observed that the virtual channel (VC) allocator allots the VC to the corresponding physical channel. The flits with routing information are then passed through the DEMUX. The input link selection to the DEMUX is made by the routing algorithm inside the router arbiter, and it is embedded into the header flit of the packet. Hence, the flits are transferred to the next hop through the desired channel. In addition, we employ a flow control mechanism within the controller to ensure faster communication. This results in the reduction of network congestion. In the flow control mechanism, the upstream node maintains a counter for the number of flits available in each relevant FIFO in the downstream node. The counter for the relevant FIFO is decremented by one whenever any flit is output by the upstream node. When a flit leaves a downstream node, a signal is sent back to the upstream node from the downstream node to increment the associated counter. This not only reduces the packet delay, but also provides more flexibility for routing by exploiting the diagonal ports. Therefore, we can conclude that the diagonal links in the proposed topology can help reduce the average delay of packets that have been routed between the remote nodes.

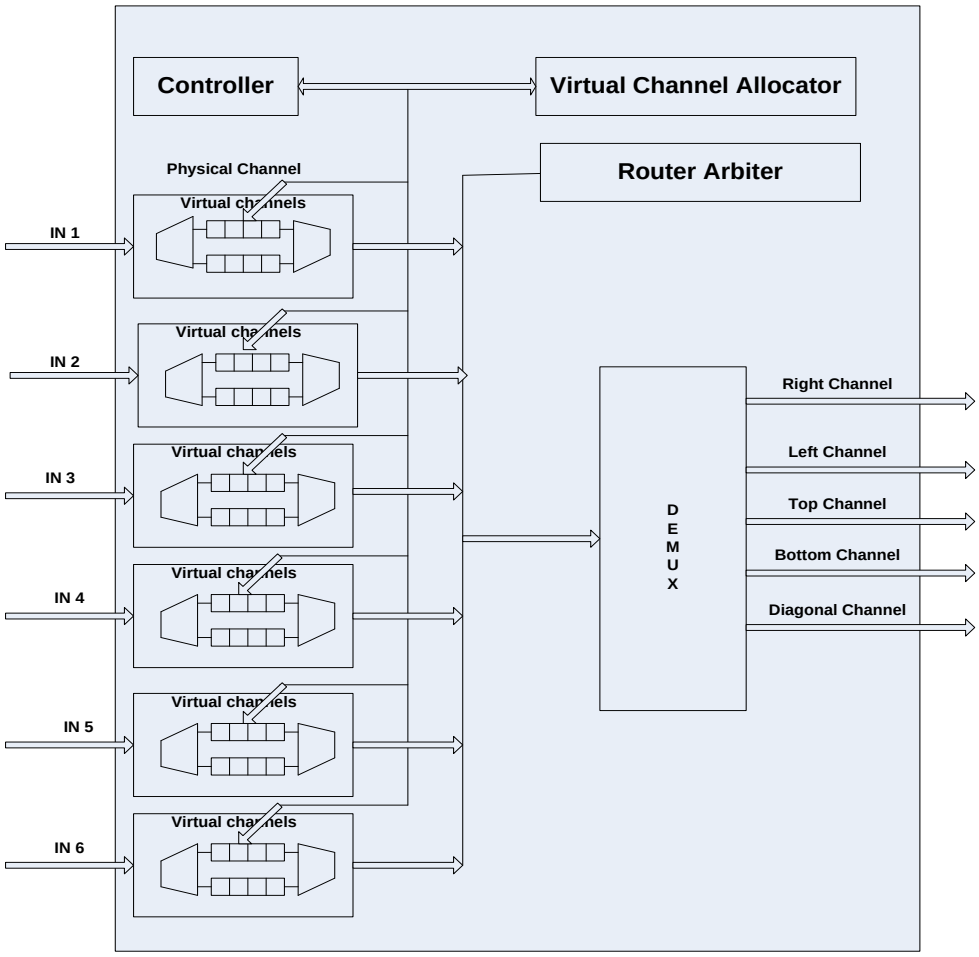

Figure 5. Router Architecture for the XDMesh Topology

3.2. The Proposed Routing Algorithm

The routing algorithm is responsible for determining the path that each packet traverses from a source node to a destination node. The basic routing algorithm used in the mesh topology is known as an XY routing algorithm [7]. To address the problems inherent in the XY routing algorithm, we propose a routing algorithm for XDMesh known as XDMeshR. Pseudocode for the proposed XDMeshR routing algorithm is shown in Figure 6. The algorithm first calculates the shortest path between the source and destination by exploiting Dijkstra's shortest path first (DSPF) routing algorithm. Dijkstra's shortest path first calculates the shortest path by effectively using the diagonal links of the XDMesh network.

XDMeshR(Source,Destination){
Input:
Coordinates of source router (X_s,Y_s)
Coordinates of destination router (X_d,Y_d)
Coordinates of destination PE (X_{pe},Y_{pe})
Output:
Selected output link
Process:
Dijkstra(XDMesh,(X_s,Y_s)) // to calculate shortest path first between source and destination//
$X_{off} = X_d - X_s$

$Y_{off} = Y_d - Y_s$
$X_{offpe} = X_{pe} - X_d$
$Y_{offpe} = Y_{pe} - Y_d$
if (X_{off} >0) select bottom link
else if (X_{off} <0) select top link
else if (Y_{off} >0) select right link
else if (Y_{off} <0) select left link
else {
if ((X_{offpe} == 0) && (Y_{offpe} == 0))
select top-left diagonal link
if ((X_{offpe} == 0) && (Y_{offpe} == 1))
select top-right diagonal link
if ((X_{offpe} == 1) && (Y_{offpe} == 1))
select bottom-right diagonal link
if ((X_{offpe} == 1) && (Y_{offpe} == 0))
select bottom-left diagonal link
}
End

}

Figure 6. Pseudocode for the XDMeshR Routing Algorithm

Once the shortest path is determined, packets can be transferred in the network using the XDMeshR routing algorithm. The XDMeshR routing algorithm is the modified version of the XY routing algorithm. Decoding of the routing information stored in the head flit of the packet is as follows: an extra bit is added to the router address to validate the destination router's address. The bit is set to '0' when the destination router's address is not determined after a packet is sent to the source router from the source PE. The bit is set to '1' for the opposite situation. Hence, the flits can be transferred easily within the network after the determination of the destination router's address through the pre-calculated DSPF route. An acknowledgement is received from the destination PE after the flits reach the destination router. By doing this, the diagonal links help to reduce the average packet delay.

4. Experimental Results and Analysis

4.1. Experimental Environment

A number of simulators have been developed to implement and analyze NoC architectures. However, most of them are topology-specific simulators. Of these, we use a general-purpose NoC simulator++ (Gpnocsim++) to evaluate our proposed topology because it supports the evaluation of different NoC topologies. It is a Java-based simulator built upon an object-oriented modular design of NoC architectural components for facilitating the evaluation of new architectures [17]. Different synthetic traffic models have been used for evaluating interconnection networks. The most widely used traffic models for the analysis of interconnection networks are uniform, transpose, bit-complement, hot-spot, and self-similar. The uniform traffic model is a standard model used in on-chip network-based routing studies. This model can be considered the traffic model for well-balanced shared memory computations [18]. Thus, we employ the uniform traffic model for evaluating state-of-the-art topologies. We also consider the Orion 2.0 [19] simulator to investigate the energy consumption of XDMesh with respect to other topologies.

The throughput of a network is the rate at which the network sends or receives data. For a packet-passing system, throughput can be defined as follows [20]:

$$Throughput = \frac{(total\ packets\ received) \times (packet\ length)}{(number\ of\ IP\ blocks) \times (total\ time)}, \quad (1)$$

Where *total packets received* refers to the number of packets successfully arriving at the destination, *packet length* is measured in flit units, *number of IP blocks* is the number of functional IPs involved in communication, and *total time* is the simulation time in clock cycles. Packet latency is defined as the time (in clock cycles) taken for a packet to go from the source to the destination. In other words, it is the time elapsed between the occurrence of a head flit injection at the source node and the occurrence of the tail flit reception at the destination node [20]. The packet latency is defined as follows:

$$Latency_{avg} = \frac{\sum_{i=1}^{N} Packet\ Latency_i}{N}, \quad (2)$$

Where N is the number of packets that have arrived at the destination node. *Packet latency*$_i$ refers to the latency of the i^{th} packet within the range *[1:N]*. In addition, the injection rate is used to investigate the effect on the overall network behavior. The injection rate is defined as the number of flits injected into the network during each clock cycle. In the simulations, it is converted to the time interval (in clock cycles) between two adjacent packets.

The average number of hops traveled by a packet from the source to the destination can be calculated using the following equation:

$$AvgHop = \frac{\sum_{i=1}^{N} PacketHop_i}{N}, \quad (3)$$

Where N is the number of packets received at the destination node. *PacketHop*$_i$ refers to the number of hops required for the packets while traveling from the source to the destination.

When flits travel in the interconnection network, both the inter-switch wires and the logic gates in the switch toggle result in energy consumption. The flits from the source node need to traverse multiple hops consisting of routers and wires to reach destinations. Thus, energy consumption increases with the number of hops required by the flits to traverse from the source to the destination [21]. The proposed XDMesh reduces the average number of hops and energy consumption in the network by inserting diagonal links. In [22], Ye *et al*. proposed a new model for evaluating the energy consumption of switch fabrics in network routers. The bit energy, Ebit, metric is defined as the energy consumed when one bit of data is transported through the router. Ebit can be calculated with the following equation:

The bit energy metric, E_{bit}, is defined as the energy consumed when one bit of data is transported through the router. E_{bit} can be calculated with the following equation:

$$E_{bit} = E_{Sbit} + E_{Lbit}, \quad (4)$$

Where E_{Sbit} and E_{Lbit} represent the energy consumed by the switching and the interconnection wires, respectively. E_{Sbit} is calculated with the following equation:

$$E_{Sbit} = \alpha_{switch} \times C_{switch} \times V^2, \quad (5)$$

Where α_{switch} is the switching activity of the router, C_{switch} is the total capacitance of the switch, and V is the supply voltage. E_{Lbit} is calculated with the following equation:

$$E_{Lbit} = \alpha_l \times C_l \times V^2 \times f_{clk}, \quad (6)$$

Where α_l is the activity factor, C_l is the load capacitance, V is the supply voltage, and f_{clk} is the frequency. Consequently, the average energy consumption for sending one bit of data from one node to another node is calculated as:

$$E_{bit}^{n_s,n_d} = \eta_{hops} \times E_{Sbit} + (\eta_{hops} - 1) \times E_{Lbit}, \quad (7)$$

Where η_{hops} is the number of hops required by the flit on its way along a path from the source node to the destination node. In the experiment, a number of network parameters are defined in order to generate different network scenarios.

The overall area overhead for different topologies is calculated with the following equation:

$$Area_{overhead} = N_1 \times A_{IP} + N_2 \times A_{I^2P} + A_{link}, \quad (8)$$

Where A_{IP} denotes the area occupied by the PEs, A_{I^2P} denotes the area occupied by the *routers*, and A_{link} denotes the area occupied by the interlink wires. N_1 and N_2 are the total numbers of *PEs* and *routers*, respectively. To maintain consistency with the previous research work, a die size of $20 \times 20\ \mu m$ is used. In addition, each *PE* is assumed to consist of the equivalent of 100K two-input NAND gates for the evaluation. Under these assumptions, the silicon area consumed by the different topologies is determined. In the experiment, a number of network parameters are defined to generate different network scenarios. Tables 1 and 2 list the key parameters and their associated values using Gpnocsim++ and Orion 2.0, respectively.

Table 1. Parameters for Gpnocsim++ Used in this Study

Parameters	Values
EFTI(N_1,N_2)	(16, 6)
DiaMesh(N_1,N_2)	(16, 16)
Mesh(N_1,N_2)	(16, 16)
XDMesh(N_1,N_2)	(16, 16)
Total simulation cycle	10000
Packet length (flit units)	150
Flit length	32 bits
Number of virtual channels	2–12
Input buffer size	6 flits

Table 2. Parameters for Orion 2.0 Used in this Study

Parameters	Values
Router name	Alpha 21364
Technology node	65 *nm*
Transistor type	LVT
Supply voltage	1.2 V
Activity factor	0.31

Operating frequency	5.1e9
Number of virtual channels	2–12
Input buffer size	6 flits
Output buffer size	6 flits

4.2. Analysis of Performance Results

Figure 7 shows throughput characteristics of the proposed XDMesh and other topologies when varying the number of virtual channels and injection rate. From Figure 7(a), we observe that the throughput increases linearly with the number of virtual channels due to the functional behavior of the IP blocks under uniform traffic distribution. The throughput of EFTI is better than the conventional mesh because EFTI has more available paths than the mesh. On the other hand, DiaMesh achieves better throughput than both EFTI and the mesh due to its diametric channels, which are used to bypass the flits between the intermediate nodes. Overall, the proposed XDMesh outperforms all of the other topologies due to its diagonal links and routing algorithm. In addition, the topological advantage of XDMesh facilitates fast communication by alleviating the congestion that can take place in the horizontal and vertical directions. Figure 7(b) illustrates the effect that varying the injection rate of different topologies has on the throughput. The throughput increases in a linear fashion due to the functional behavior of the IP blocks. However, when the injection rate is more than *0.5*, the throughput is saturated because many flit conflicts occur; this limits increases in throughput due to limitations in routing resources, including routers and interconnect wires. Therefore, to maintain a balanced throughput according to the injection rate, we set the injection rate to 0.5.

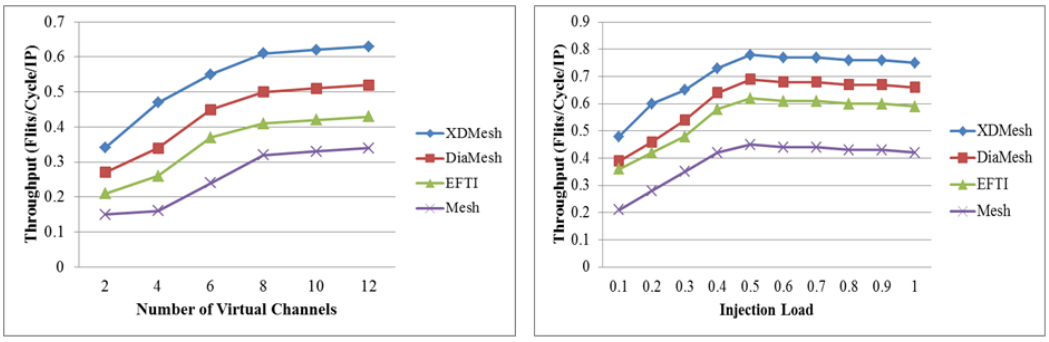

Figure 7. Throughputs of different Topologies when Varying (a) the Number of Virtual Channels and (b) Injection Load

Figure 8(a) presents additional data which show the effect that varying the number of virtual channels has on the latency for different topologies. The average packet latency generally increases with an increase in the number of virtual channels, because increasing the number of virtual channels increases the switching complexity and requires a significant amount of buffer storage within the channel. According to the experimental results, 6–8 virtual channels are adequate for a network to maintain an appropriate balance between high throughput and low latency. Figure 8 (b) shows the effect that varying the injection rate has on the latency. We observe that the injection rate directly affects the average packet latency in an increasing order. The rate of latency increase is much higher when the injection rate is higher than 0.5, because of the high traffic. We

observe that the mesh-based topology exhibits the highest latency compared to other topologies because the traffic is constrained at the four destinations within the shortest Manhattan distance.

EFTI exhibits a higher latency than DiaMesh and XDMesh because it has fewer routers than these topologies and the traffic in the EFTI is limited within a single sub-tree. Consequently, the network performs well when the injection rate is kept below 0.5. Overall, the diagonal links in the proposed XDMesh are used to bypass the horizontal and vertical nodes and to choose the minimum distance, exhibiting lower latency compared to other topologies.

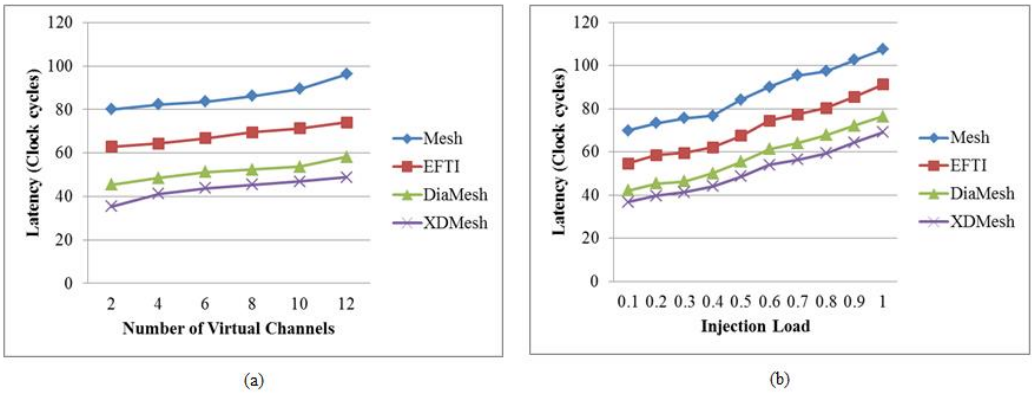

Figure 8. Latency with a Varied (a) Number of Virtual Channels and (b) Injection Load

4.3. Analysis of Energy Consumption Results

Energy consumption is as important as improved performance for NoC architectures. We evaluate the energy consumption of different topologies by varying the number of virtual channels and the injection load. Figure 9(a) shows the variation of the average energy consumption per flit as a function of the number of virtual channels, assuming the networks are operated at the peak sustainable data rate. We observe that the energy consumption increases linearly with the number of virtual channels for all of the topologies. In addition, the energy consumption with EFTI is much higher than with other topologies because of EFTI's irregular structure, which leads to paths of different sizes, and consequently, many hops being required for the packets to transfer from the source to the destination node. For example, the routing inside a level-one EFTI router entails the use of a buffer and a switch, which consume twice the energy of the other topologies because the switches and buffers of routers from level one must provide a path for packets coming from the upper level as well as from the same router. This increases the number of hops compared to other topologies. The average energy consumption in the proposed XDMesh is less than that of other state-of-the-art topologies because of its diagonal links which allow the flits to bypass the horizontal and vertical routers and reduce the number of hops from the source to the destination node. Figure 9(b) shows the average energy consumption with respect to the injection rate. As with the virtual channels, the energy consumption increases linearly as the injection rate increases. However, when the injection rate is greater than 0.5, the energy consumption is saturated because a high amount of traffic occurs. Thus, we infer that a topology with a higher degree of connectivity like EFTI has greater energy consumption than other topologies. The mesh energy consumption is higher than that of DiaMesh because of the absence of additional paths. Overall, XDMesh requires less energy consumption than other topologies because of its diagonal links which reduce the number of hops from the source to the destination node.

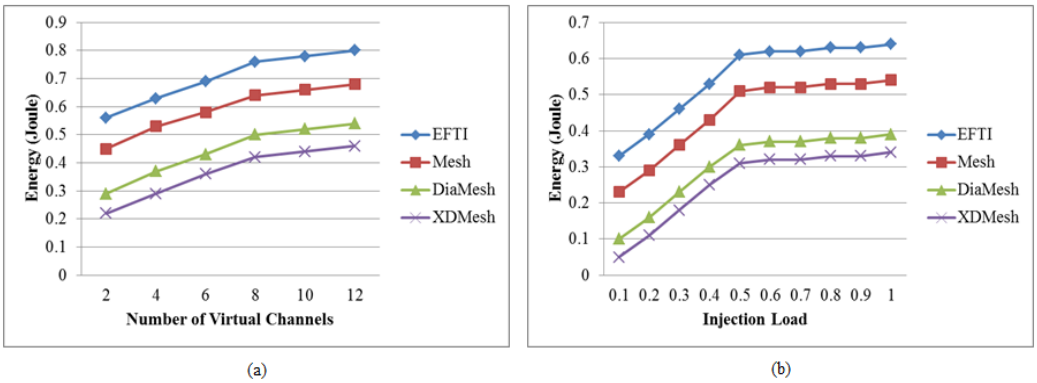

Figure 9. Energy Consumption with Varied (a) Number of Virtual Channels and (b) Injection Load

Figure 10 and Table 3 depict the silicon area of different interconnect architectures, which are categorized into interlink wires, a router including the buffer size, and functional IPs. The buffer space within the router is the main area consumer in the NoC architectures. Furthermore, the buffer space depends on the number of virtual channels, and the communication load implies the amount of flits for handling the performance of the network. We observe that EFTI has a considerably higher area overhead compared to other topologies because it requires more links to connect between the nodes. In addition, EFTI requires more virtual channels than other topologies to achieve better performance. Moreover, a greater number of blocks (PEs) are used than in the other topologies. Thus, EFTI consumes more silicon area. In contrast, of all of the topologies, XDMesh consumes the least silicon area because it allows flits to bypass buffering and arbitration by the proposed router. The DEMUX used in the router passes the flits to the output link very quickly. Thus, the area overhead of the proposed XDMesh can be reduced by reducing the buffer size of the router.

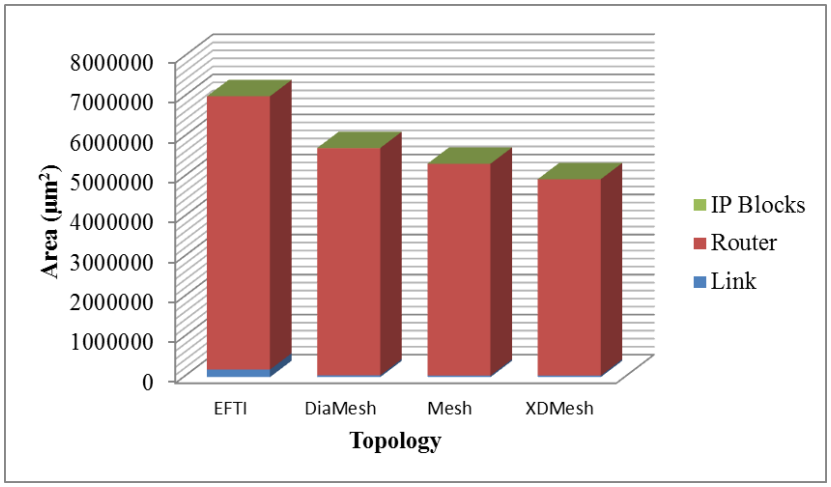

Figure 10. Silicon Area Overhead for different Topologies

Table 3. Component-Wise Silicon Area Overhead

Components	EFTI	DiaMesh	Mesh	XDMesh
Interlink wires	187,616 μm^2	43,706 μm^2	38,376 μm^2	41,042 μm^2
Router including buffer size	6,824,371 μm^2	5,676,782 μm^2	5,289,090 μm^2	4,902,223 μm^2
Functional IPs	2,560 μm^2	1,440 μm^2	1,440 μm^2	1,011 μm^2
Total Area	7,014,547 μm^2	5,721,928 μm^2	5,328,906 μm^2	4,944,276 μm^2

5. Conclusions

In this paper, we proposed an extended XDMesh topology by closely examining a routing algorithm to enhance the network performance and reduce the energy consumption of the NoC. We analyzed the performance of XDMesh in terms of throughput, latency, and energy consumption. In addition, we compared the performance of XDMesh with existing state-of-the-art NoC topologies. Experimental results indicated that XDMesh increases throughput by *44.47%*, *23.39%*, and *17.85%* and reduces latency by *40.10%*, *25.71%*, and *16.56%* over mesh, extended-butterfly fat tree, and diametrical mesh topologies, respectively. In addition, XDMesh outperformed other topologies in terms of energy consumption and silicon area.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2013R1A2A2A05004566).

References

[1] M. Bakhouya, S. Suboh, J. Gaber, E. Ghazawi and S. Niar, "Performance Evaluation and Design Tradeoffs of On-chip Interconnect Architectures", Simul. Modelling Pract. Theory, vol. 19, no. 6, **(2011)**, pp. 1496-1505.
[2] C. C. Ling and R. Marculescu, "Designing Heterogeneous Embedded Network-on-Chip Platforms with Users in Mind", IEEE Trans. Compt. Aided Desg. Intg. Circuits and Syst., vol. 29, no. 9, **(2010)**, pp. 1301-1314.
[3] E. Strohmaier, J. J. Dongarra, H. W. Meuer and H. D. Simon, "Recent Trends in the Marketplace of High performance Computing", Parallel Compt., vol. 31, no. 4, **(2005)**, pp. 261-273.
[4] S. Suboh, M. Bakhouya, J. Gaber and E. Ghazawi, "An Interconnection Architecture for Network-on-Chip Systems", Telecomm. Syst., vol. 31, **(2008)**, pp. 137-144.
[5] A. Balakrishnan and A. Naeemi, "Interconnect Network Analysis of Many-Core Chips", IEEE Trans. Electron Dev., vol. 58, no. 9, **(2011)**, pp. 2831-2837.
[6] G. David, "Networks on Processors Improve On-Chip Communications", IEEE Trans. Compt., vol. 42, no. 3, **(2009)**, pp. 17-20.
[7] S. Kumar, A. Jantsch, J. P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja and A. Hemani, "A Network on Chip Architecture and Design Methodology", In Proc. IEEE Annual Symp. Compt. Society on VLSI, Pittsburgh, PA, USA, **(2002)**, pp. 105-112.
[8] H. Hossain, M. M. Akbar and M. M. Islam, "Extended-Butterfly Fat Tree Interconnection (EFTI) Architecture for Network-on-Chip", In Proc. IEEE PACRIM on Communications, Computers and Signal Processing, Victoria, Canada, **(2005)**, pp. 613-616.
[9] P. Ghosal and T. S. Das, "A Novel Routing Algorithm for On-Chip Communication in NoC on Diametrical 2D Mesh Interconnection Architecture", Advances in Intell. Syst. Compt., vol. 178, **(2013)**, pp. 667-676.
[10] M. R. Seifi and M. Eshghi, "Clustered NoC, A Suitable Design for Group Communications in Network-on-Chip", J. Compt. Elect. Eng., vol. 38, no. 1, **(2012)**, pp. 82-95.

[11] A. Leroy, J. Picalausa and D. Milojevic, "Quantitative Comparison of Switching Strategies for Networks on Chip", In Proc. IEEE Intl. Conf. Programmable Logic, Mar del Plata, Argentina, **(2007)**, pp. 57-62.
[12] U. Y. Orgas and R. Marculescu, "It's a Small World After All: NoC Performance Optimization via Long-Range Link Insertion", IEEE Trans. VLSI Syst., vol. 14, no. 7, **(2006)**, pp. 693-706.
[13] W. J. Dally W. J, "Express Cubes: Improving the Performance of K-Ary N-Cube Interconnection Networks", IEEE Trans. Compt., vol. 40, no. 9, **(1991)**, pp. 1016-1023.
[14] S. R. Nadooshan, M. Modarressi and S. H. Azad, "The 2D Digraph-based NoCs: Attractive Alternatives to the 2D Mesh NoCs", J. Supercompt., vol. 59, no. 1, **(2012)**, pp. 1-21.
[15] J. Camacho, J. Flich, J. Duato, H. Eberle and W. Olesinski, "A Power-Efficient Network-on-Chip Topology", In Proc. 5th Intl. Workshop on Interconnection Network Architecture, New York, NY, USA, **(2011)**, pp. 23-26.
[16] A. Deorio, D. Fick, V. Bertacco, D. Sylvester, D. Blaauw, H. Jin and G. Chen, "A Reliable Routing Architecture and Algorithm for NoCs", IEEE Trans. Computer Aided Des. of Integ. Circuits and Syst., vol. 31, no. 5, **(2012)**, pp. 726-739.
[17] A. Azimi, A. Ahrabi, H. Bahrbegi and M. Bahrbegi, "gpnocsim++: A network-on-chip simulator", http://www.sourceforge.net/projects/gpnocsimpp, **(2009)**.
[18] G. V. Varatkar and R. Marculescu, "On-chip Traffic Modeling and Synthesis for MPEG-2 Video Applications", IEEE Trans. VLSI Syst., vol. 12, no. 1, **(2004)**, pp. 108-119.
[19] A. B. Kahng, L. Bin, P. L. Shiuan and K. Samadi, "ORION 2.0: A Power-Area Simulator for Interconnection Networks", IEEE Trans. VLSI Syst., vol. 20, no. 1, **(2012)**, pp. 191-196.
[20] 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. Compt., vol. 54, no. 8, **(2005)**, pp. 1025-1040.
[21] T. Ye, L. Benini and G. D. Micheli, "Analysis of Power Consumption on Switch Fabrics in Network Routers", In Proc. IEEE Design Automat. Conf., **(2002)**, pp. 524-529.
[22] H. Jingcao and R. Marculescu, "Energy-aware communication and task scheduling for network-on-chip architectures under real-time constraints", In Proc. IEEE Design, Automation and Test in Europe Conference and Exhibition, **(2004)**, pp. 234-239.

Authors

Md Hasan Furhad, received a BS degree in Computer Science & Engineering from Rajshahi University of Engineering & Technology, Bangladesh, in 2006, and an MS at Computer and Information Technology in University of Ulsan, South Korea. His research interests include network on chip, computer architecture, and embedded systems.

Jong-Myon Kim received a BS in electrical engineering from Myongji University, Yongin, Korea, in 1995, an MS in electrical and computer engineering from the University of Florida, Gainesville, in 2000, and a PhD in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, in 2005. He is an associate professor of Electrical Engineering at the University of Ulsan, Korea. His research interests include multimedia processing, multimedia-specific processor architecture, parallel processing, and embedded systems. He is a member of IEEE and the IEEE Computer Society.

