DMesh: a Diagonally-Linked Mesh Network-on-Chip Architecture

Wen-Hsiang Hu, Seung Eun Lee, and Nader Bagherzadeh
Department of Electrical Engineering and Computer Science
University of California, Irvine
Irvine, CA 92697 USA
{wenhsiah, seunglee, nader} @uci.edu

Abstract— In this paper, we propose a new mesh-typed NoC architecture which aims at enhancing network performance while keeping implementation cost feasible. The result is a diagonally-linked mesh (DMesh) NoC that uses wormhole packet switching technique. Together with the proposed adaptive quasi-minimal routing algorithm, DMesh improves average latency and saturation traffic load. In addition, logic synthesis results show that adding diagonal links is a more area-efficient way for increasing network performance than using large buffers.

I. INTRODUCTION

As semiconductor technology continues its phenomenal growth and follows the Moore's Law, the amount of computation power and storage that can be integrated on a chip increases. There have been previous articles reporting a single chip that incorporated 64 cores [1] and another one with 80 cores [2]. While the computation logic grows, the performance of on-chip interconnections does not scale as well. Starting with 0.25μm CMOS technology, wire delay dominates gate delay and the gap between wire delay and gate delay becomes wider as process technology improves. In addition, human design productivity can not keep up with the growth rate of available circuits on a single chip. These issues call for a well-structured design approach, modularized design methodology, clear programming model and predictable behavior of the system [5]. There is a need for a new on-chip interconnection architecture to solve these design challenges.

Network-on-chip (NoC) interconnection scheme is proposed as a unified solution for the design problems faced in advanced process technology [3][4]. With NoC, we can apply wire segmentation and wire sharing design techniques to resolve the performance bottleneck due to wire delay. NoC uses a distributed control mechanism, resulting in a scalable interconnection network. The use of standardized sockets enables modular design and intellectual property (IP) reuse and the system predictability can be obtained by using guaranteed service provided by NoC. Therefore, there is growing interest in NoC research [5][6] and NoC is considered as a practical approach for the next-generation on-chip interconnection.

We have recently developed a multi-processor system platform called Network-based Processor Array (NePA) [7] in which the processors are interconnected by using an on-chip two-dimensional (2D) mesh network. The NePA NoC is a deadlock-free and livelock-free network that implements the wormhole packet switching technique and utilizes an adaptive minimal routing algorithm. To further improve the performance of NePA NoC, we propose in this paper to add diagonal links to the 2D mesh network, because of the emergence of X-architecture routing technique in chip manufacturing [8][9]. The diagonal links not only reduce the distance between a source node and a destination node but alleviate traffic congestion in the network so that the network performance is enhanced. Our proposed NoC architecture is referred to as DMesh: Diagonally-linked Mesh. Simulation results from self-similar traffic show that DMesh improves the average latency and the saturation traffic load on both 4x4 and 8x8 mesh networks. In addition, logic synthesis results in TSMC 65nm CMOS process show that adding diagonal links is a more area-efficient way to improve network performance than increasing buffer size.

The rest of this paper is organized as follows: Section 2 presents the background knowledge of NoC architecture and related researches. The proposed DMesh NoC architecture is discussed in Section 3. Section 4 presents experimental results. Finally, brief statements conclude this paper in the last section.

II. BACKGROUND

In this section, we discuss the background of NoC architecture and provide a review of some related works in this field, as well as an overview of the NePA platform.

A. NOC Architecture

The function of an on-chip network is to deliver messages from source node to destination node and there exist many design alternatives to accomplish this job. Depending on the application requirements, how to choose suitable network architecture remains an open problem in this field of research. Here we discuss the network properties that need to be considered when devising an NoC architecture for specific application needs.

1) Switching policy

There are two major switching techniques: circuit switching and packet switching. Circuit switching establishes a link between source node and destination node either

virtually or physically before a message is being transferred. The link is held until all the data is transmitted. The major advantages of circuit switching are that there is no contention delay during message transmission and its behavior is more predictable, so circuit switching is usually employed when Quality of Service (QoS) is considered. Examples of using this technique are [15] and [16].

On the other hand, packet switching transfers messages on a per-hop basis. With packet switching, messages are divided into packets at the source node and then sent into a network. Packets move along a route determined by the routing algorithm and traverse through a series of network nodes and finally arrive at the destination node. Packet switching is utilized in most of NoCs because of its potential for providing simultaneous data communication between many source-destination pairs. Readers are referred to [6] for a list of NoCs utilizing packet switching techniques. Packet switching can be further classified into three classes: store and forward (SAF), virtual cut through (VCT), and wormhole switching. The most popular one for NoC based architectures is wormhole switching because it only requires a buffer size of one flit (flow control unit) so that the area cost of a router can be kept low. In contrast, SAF and VCT require a buffer size of the whole packet which prohibits their adoption.

2) Topology

Topology defines how nodes are placed and connected, affecting the bandwidth and latency of a network. Many different topologies have been proposed, [6], such as mesh, torus, binary tree, Octagon, mixed and custom topology, as shown in Fig. 1. Some researchers have proposed the application-specific topology that can offer superior performance while minimizing area and energy consumption [17][18]. The most common topologies are 2D mesh and torus due to their grid-type shapes and regular structure which are the most appropriate for the two dimensional layout on a chip.

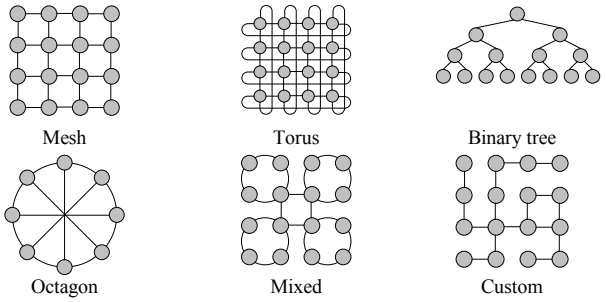


Figure 1. NoC topologies.

3) Routing

Routing is the mechanism responsible for determining the path that a packet traverses from the source node to the destination node. Routing algorithms such as deterministic and adaptive ones have been proposed. With deterministic routing, the path between source-destination pair is fixed, regardless of the current state of the network. On the other hand, an adaptive routing algorithm takes the network state into account when deciding a route, resulting in variation of the routing path with time. For example, it may choose an alternative path if a certain link is congested, therefore, an adaptive routing algorithm has the potential of supporting more traffic for the same network topology. However, most of the proposed packet-switched NoCs use deterministic routing because of its simplicity and the low area overhead in router design.

B. *NePA*

We provide an overview of the NePA architecture in this section. NePA implements the wormhole packet switching technique and the topology of NePA is based on a 2D mesh as shown in Fig. 2. Each node in NePA consists of a router and a local IP which can be a CPU, DSP, memory block, or application-specific logic. The router connects with its four neighboring routers via six bidirectional links. A key feature of the NePA architecture is the use of two separate vertical links which are employed to construct a deadlock-free network [19]. The NePA network is actually composed of two disjoint sub-networks. One sub-network is responsible for delivering east-bounded packets while the other one is for west-bounded packets. Therefore, cycles in the resource dependence graph [14] and prevent deadlocks from happening. This design technique reduces the design complexity of the router because there is no need for a deadlock aware routing algorithm. To increase network performance, NePA utilizes an adaptive XY routing algorithm. When an output port is congested, or the output buffer is full, the router selects an alternative output port for packets. Therefore, the link utilization is balanced and network performance improves.

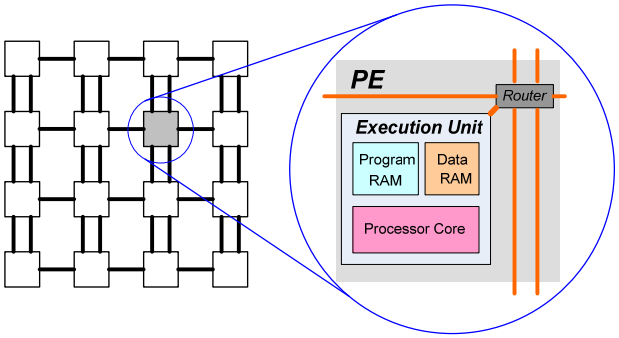


Figure 2. A 4x4 NePA network and its node composition.

III. DMesh ARCHITECTURE

A. *Topology*

The DMesh network is constructed by integrating diagonal links to NePA, as presented in Fig. 3. Each node has 10 64-bit bidirectional links connecting with its neighbors so the DMesh router has 10 output ports (*N1/N2/S1/S2/E/W/NE/NW/SE/SW-out*) and 10 input ports (*N1/N2/S1/S2/E/W/NE/NW/SE/SW-in*). Additionally, there are three ports for connection with local PEs: *IntR*, *IntL* and *Int*. Fig. 4 depicts the input and output ports of NePA router and DMesh router. The DMesh network is composed of two sub-networks: *E-subnet* and *W-subnet*, represented in dashed arrows and solid arrows in Fig. 3, respectively. The E-subnet is responsible for transferring

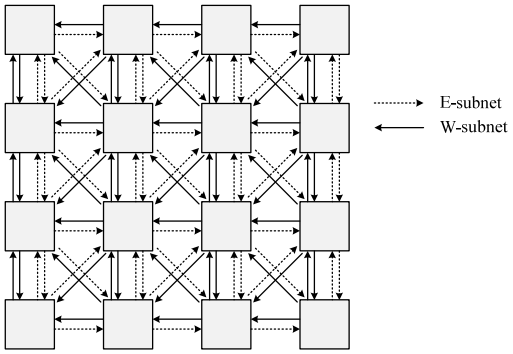


Figure 3. Topology and links of DMesh.

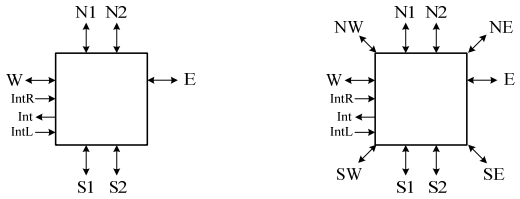


Figure 4. Ports of NePA router and DMesh router.

packets eastward while the W-subnet is for transmitting westward traffic. When source PE starts packet transmission, it injects packets into the network via IntR or IntL port, depending on the direction of destination PE. The IntR port is in charge of injecting packets into the E-subnet and the IntL port is for the W-subnet. Then the packets traverse in one of the sub-networks to their destinations. When packets arrive at the destination node, they are ejected from the Int port.

B. Packet format

With wormhole packet switching, DMesh packets are composed of 64-bit flits. We utilized the same packet format defined for NePA in [19]. There are four types of packets defined. The single data transfer packet consists of one flit and is used for transferring 32-bit data. The single command packet is for building control specific protocols between processor elements (PEs) or between a PE and a router.

DMesh also supports multiple data packets, which are used for transmitting more than one 32-bit words at a time, because multiple data transmission has better performance in terms of communication overhead than the single data transmission. Two different block transfers are defined. One is block program transfer packet which is used for programming each PE. The other is block data transfer packet used for transferring multiple data words between PEs. The block program/data transfer packet consists of a header flit and a series of body flits which contains the actual program/data to be transmitted. The number of body flits is encoded in the header flit.

The address of destination PE is represented in the X-dir field and Y-dir field in a relative distance format. For instance, if the destination node is on the east side of the source node the X-dir field has a positive value. The X-dir field has a negative value if the destination node is on the west side of the source node. This technique of relative address representation helps reduce router design effort because a same router can be applied to all network nodes without any modification. We also incorporated the seq_num field in the packet for reordering out-of-order delivery. The single/block data transfer packet has the sourcePE_address field and the application-dependent data_ID field in order for the destination PE to identify received data.

C. Routing

We devised a distributed adaptive routing algorithm for DMesh. With distributed routing, the selection of the next hop is decided at the current node and the path selection is based on a quasi-minimal routing technique. Take Fig. 5 for example, if there is a packet being transferred from node S to node D, there are three alternative paths: *a*, *b*, and *c*. Clearly, path *a* is the shortest path. However, from our preliminary simulation, if a minimal routing algorithm is adopted and we always choose the shortest path there will be severe congestion on diagonal links and low utilization on vertical and horizontal links. Thus, the network performance is impacted. Our quasi-minimal routing relaxes the output port selection. In this example, it allows packets to take path *b* or path *c* depending on the network state, if path *a* is congested. Although the packet may traverse a longer path, this approach helps balance link load and relieve congestion. In order to solve contention at an output port, we employed a fixed-priority scheme for arbitration. In general, the diagonal input ports are given the highest priority, then the horizontal and vertical input ports, and IntR and IntL have the lowest priority.

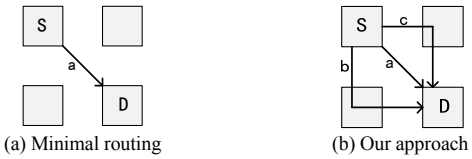


Figure 5. Example of route selection

IV. PERFORMANCE AND COST EVALUATION

Here we describe the methodology used to analyze the performance and area cost of DMesh architecture and present the results.

A. Performance evaluation

To evaluate DMesh performance and compare it with NePA, we constructed a SystemC based cycle accurate simulator called eNoC. In eNoC, we can change various network configurations, such as network size, topology, buffer size, routing algorithm, priority scheme for router arbitration, and traffic pattern. There are four different traffic patterns used for measuring the performance: uniform random, bit complement, bit reverse and matrix transpose traffic patterns. These patterns define the spatial distribution of packets.

As for the temporal distribution of packets, we adopted the self-similar traffic generation techniques. Self-similar traffic has been found in the traffic between on-chip modules in MPEG-2 video applications [10] and conventional computer networks [11]. Researchers [12] have shown that self-similar traffic can be generated by aggregating a large number of packet sources which exhibit a long-range dependence property. We used the modeling method proposed in [13] to produce the self-similar traffic. During simulation, each source node is either in the ON or OFF state. A source node generates packets when it is in the ON state and it does not generate any packets when in the OFF state. The length of time a node spends in the ON or OFF states is determined by the Pareto distribution ($F(x) = 1 - x^{-\alpha}$, $1<\alpha<2$). The equations for calculating ON and OFF times are

$$T_{ON} = U^{-1/\alpha_{ON}} \quad (1)$$

$$T_{OFF} = U^{-1/\alpha_{OFF}} \quad (2)$$

U is a uniformly distributed value in the range of (0, 1], α_{ON} = 1.9 and α_{OFF} = 1.25.

We used a standard interconnection network measurement setup described in [14]. After a packet is generated, it is stored in an infinite queue at the source node and waits for being injected into the network. This mechanism referred to as the open-loop measurement configuration isolates the packet generation from the network behavior, i.e. the packet generation is independent of the network condition. Each simulation executes 10,000 clock cycles for warm-up and then continues for 100,000 cycles during which performance measurements are conducted.

Two performance metrics are of importance to us: latency and throughput. In order to compute latency information, each flit in eNoC is declared as a SystemC object that carries four latency related private variables: generation time (T_g), injection time (T_i), arrival time (T_a) and inter-node distance (D). Inter-node distance is represented in terms of the number of hops between source-destination pairs. With this information, the latency, queuing delay, and blocking time of each flit can be calculated by the following equations:

$$\text{Latency} = T_a - T_g \quad (3)$$

$$\text{Queuing delay} = T_i - T_g \quad (4)$$

$$\text{Blocking time} = T_a - T_i - (D * \text{clock cycle time}) \quad (5)$$

The average inter-node distance is shown in Table I. We can see that our routing algorithm makes efficient use of diagonal links so that the inter-node distance is reduced in all traffic patterns. Matrix transpose traffic has the largest improvement and makes the most of the diagonal links because the source-destination pairs are all symmetric to the diagonal in a matrix.

The comparison of average latency in 4x4 and 8x8 networks under four different traffic patterns is shown in Fig. 8. In both 4x4 and 8x8 network size, DMesh outperforms NePA. In particular, the 4x4 network under bit reverse and matrix transpose traffic, the latency in DMesh is a constant because the network is capable of resolving all routing resource contentions. That is, each source-destination pair can obtain an alternative path that is not occupied by other packets. In Fig. 6, we compare the queuing delay, traverse latency, and blocking time for 4x4 and 8x8 networks under random traffic. For different traffic loads, all of these delays are decreased in DMesh. The saturation load (the point where throughput no longer grows linearly with traffic load) in various configurations are summarized in Table III. It can be observed that DMesh is able to sustain higher load than the NePA. For random traffic, the improvement in the 8x8 network is more than the 4x4 network which implies that DMesh has a greater impact on systems with more nodes. From Table III, it can be seen that the increase in FIFO sizes does not help much with the saturation load.

B. Area cost and power consumption evaluation

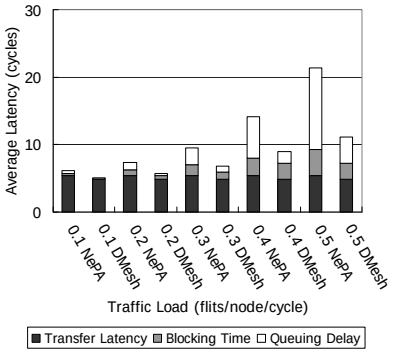
In order to estimate hardware cost, we implemented the NePA router, the DMesh router and the FIFO buffer in Verilog and performed logic synthesis by using the Synopsys Design Compiler to get gate count information. Various buffer sizes were also evaluated. For the NePA router, we followed the architecture described in [7]. The block diagram of DMesh router is presented in Fig. 7. The DMesh router has three sub-routers for processing traffic in the E-subnet, W-subnet and Int output port. There is a FIFO associated with each input port. Header processing unit (HPU) extracts destination information from the header flit and routing logic (RL) is used to decide routing path, perform arbitration and control the crossbar switch.

We used TSMC 65nm CMOS generic process technology in logic synthesis. The target clock rate is set to be 800 MHz and is met in all configurations. The results are listed in Table II. From the table, we can observe that the gate count and power consumption of the DMesh router with a FIFO depth of 4/8 is roughly equal to or less than those of the NePA router with a FIFO depth of 8/16. Performance comparisons of these four configurations are shown in Fig. 9. It is clear that DMesh has a shorter latency than NePA with similar hardware cost. For example, for a 8x8 mesh in random traffic, DMesh with a FIFO depth of 4 has a shorter latency than NePA with a FIFO depth of 8 and 16. All other configurations have similar results. Fig. 9 also shows that the improvements from diagonal links are more than those from larger buffers. Therefore, DMesh is a more area-efficient architecture.

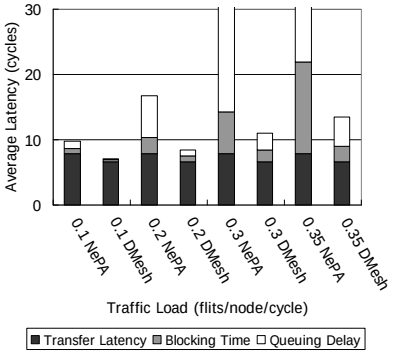
TABLE I. COMPARISON OF AVERAGE INTER-NODE DISTANCE

Network	4x4 network			
	Random	***Bit complement***	***Bit reverse***	***Matrix transpose***
NePA	2.38	4.33	3.32	3.38
DMesh	1.83	2.55	1.99	1.79
Reduction	23.1%	41.1%	40.0%	47.0%

Network	8x8 network			
	Random	***Bit complement***	***Bit reverse***	***Matrix transpose***
NePA	4.89	9.75	6.19	7.33
DMesh	3.66	5.80	4.04	3.86
Reduction	25.1%	40.5%	34.7	47.3



(a) 4x4 mesh network



(b) 8x8 mesh network

Figure 6. Comparison of various delays in random traffic

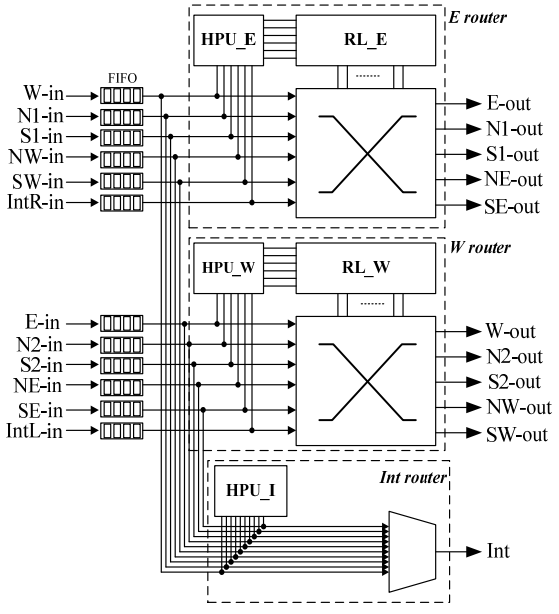


Figure 7. Block diagram of DMesh router.

TABLE II. GATE COUNT (EQUIVALENT 2-INPUT NAND GATE) OF NEPA AND DMESH ROUTER NODE (FIFOS INCLUDED)

FIFO depth (flits)	NePA		DMesh	
	Gate Count	Dynamic Power (mW)	Gate Count	Dynamic Power (mW)
2	18368	6.99	32750	11.53
4	28654	12.29	46598	20.40
8	47038	22.32	75382	36.88
16	85362	42.03	134479	69.45
32	163330	81.27	250559	134.36
64	316173	159.33	490820	261.95

V. CONCLUSION

We developed a novel DMesh NoC architecture and demonstrated its performance enhancement over the previous work. Hardware cost evaluation also shows that our approach is more area-efficient than previously reported results. With more links in the network, we anticipate that DMesh has the potential of supporting better QoS and fault tolerance capability.

REFERENCES

[1] S. Bell et al., "TILE64 Processor: A 64-Core SoC with Mesh Interconnect," Solid-State Circuits Conference, 2008. Digest of Technical Papers. IEEE International, pp. 88-598, 2008.

[2] S. Vangal et al., "An 80-Tile 1.28TFLOPS Network-on-Chip in 65nm CMOS," Solid-State Circuits Conference, 2007. Digest of Technical Papers. IEEE International, pp. 98-589, 2007.

[3] W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks," Design Automation Conference, 2001. Proceedings, pp. 684-689, 2001.

[4] L. Benini and G. De Micheli, "Networks on chip: a new paradigm for systems on chip design," Design, Automation and Test in Europe Conference and Exhibition, 2002. Proceedings, pp. 418-419, 2002.

[5] T. Bjerregaard and S. Mahadevan, "A survey of research and practices of Network-on-chip," ACM Computing Surveys, vol. 38, pp. 1, 2006.

[6] E. Salminen et al., "Survey of Network-on-Chip proposals," White Paper, OCP-IP, March 2008.

[7] J. H. Bahn, S. E. Lee, Y. S. Yang, J. Yang and N. Bagherzadeh, "On Design and Application Mapping of a Network-on-Chip(NoC) Architecture," Parallel Processing Letters, vol. 18, pp. 239-255, 2008.

[8] M. Igarashi, T. Mitsuhashi, A. Le, S. Kazi, Y. T. Lin, A. Fujimura and S. Teig, "A Diagonal-Interconnect Architecture and Its Application to RISC Core Design," IEIC Technical Report (Institute of Electronics, Information and Communication Engineers), vol. 102, pp. 19-23, 2002.

[9] S. L. Teig, "The X architecture: Not your father's diagonal wiring," Proceedings of the 2002 International Workshop on System-level Interconnect Prediction, pp. 33-37. 2002.

[10] G. Varatkar and R. Marculescu, "Traffic analysis for on-chip networks design of multimedia applications," in DAC '02: Proceedings of the 39th Conference on Design Automation, 2002, pp. 795-800.

[11] W. E. Leland, M. S. Taqqu, W. Willinger and D. V. Wilson, "On the self-similar nature of Ethernet traffic (extended version)," Networking, IEEE/ACM Transactions on, vol. 2, pp. 1-15, 1994.

[12] M. S. Taqqu, W. Willinger and R. Sherman, "Proof of a fundamental result in self-similar traffic modeling," SIGCOMM Comput. Commun. Rev., vol. 27, pp. 5-23, 1997.

[13] D. R. Avresky, "Performance evaluation of the ServerNet(R) SAN under self-similar traffic," Parallel and Distributed Processing, 1999. 13th International and 10th Symposium on Parallel and Distributed Processing, 1999. 1999 IPPS/SPDP. Proceedings, pp. 143-147, 1999.

[14] W. J. Dally, Principles and Practices of Interconnection Networks. Morgan Kaufmann, 2004.

[15] K. Chang, J. Shen and T. Chen, "Evaluation and design trade-offs between circuit-switched and packet-switched NOCs for application-specific SOCs," in DAC '06: Proceedings of the 43rd Annual Conference on Design Automation, 2006, pp. 143-148.

[16] P. Marchal, D. Verkest, A. Shickova, F. Catthoor, F. Robert and A. Leroy, "Spatial division multiplexing: a novel approach for guaranteed

throughput on NoCs," Hardware/Software Codesign and System Synthesis, 2005. CODES+ISSS '05. Third IEEE/ACM/IFIP International Conference on, pp. 81-86, 2005.

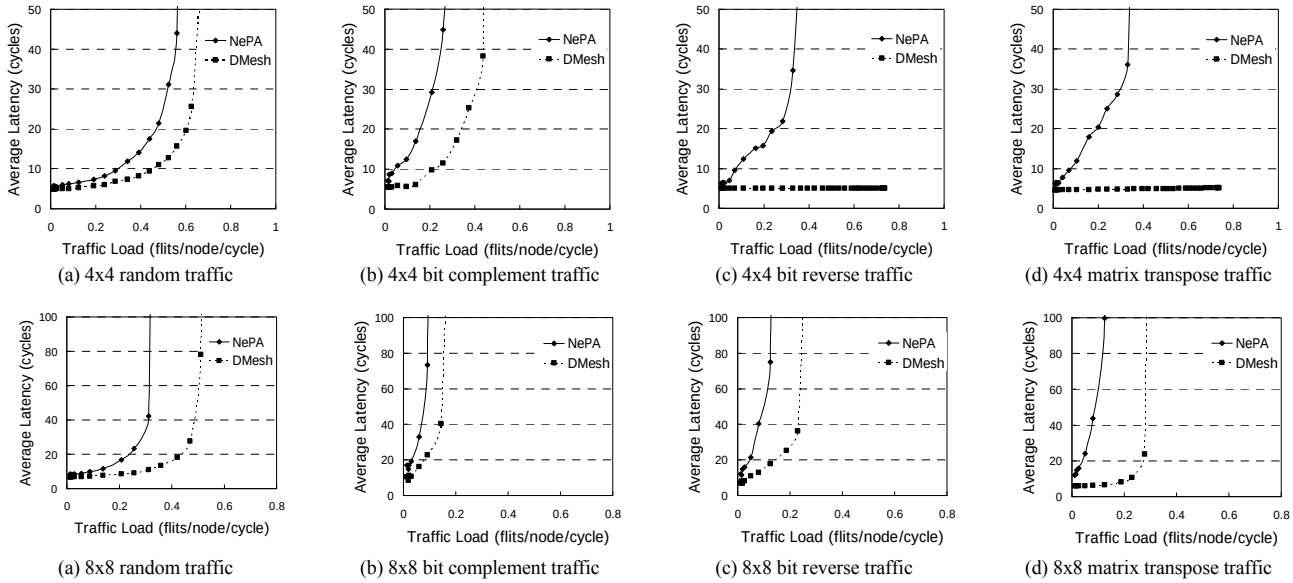
[17] J. Hu, Y. Deng and R. Marculescu, "System-level point-to-point communication synthesis using floorplanning information," in ASP-DAC '02: Proceedings of the 2002 Conference on Asia South Pacific Design automation/VLSI Design, 2002, pp. 573.

[18] D. Bertozzi, A. Jalabert, S. Murali, R. Tamhankar, S. Stergiou, L. Benini and G. D. Micheli, "NoC Synthesis Flow for Customized Domain Specific Multiprocessor Systems-on-Chip," vol. 16, pp. 113-129, 2005.

[19] J. H. Bahn, S. E. Lee and N. Bagherzadeh, "On Design and Analysis of a Feasible Network-on-Chip (NoC) Architecture," Information Technology, 2007.ITNG'07.Fourth International Conference on, pp. 1033-1038, 2007.

TABLE III. COMPARISON OF SATURATION LOAD OF NEPA AND DMESH

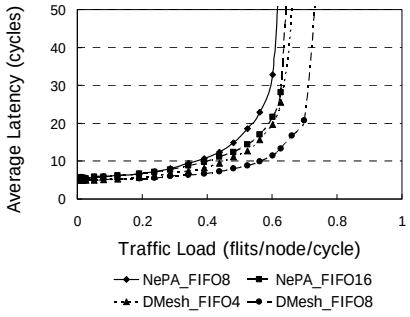
Network		FIFO depth = 2	FIFO depth =4	FIFO depth =8	FIFO depth =16	FIFO depth =32	FIFO depth =64
4x4 Random	NePA	0.519	0.595	0.626	0.659	0.686	0.695
	DMesh	0.597	0.688	0.752	0.803	0.828	0.855
	Improvement	15.0%	15.6%	20.1%	21.8%	20.6%	23.0%
4x4 Bit-complement	NePA	0.360	0.361	0.359	0.362	0.365	0.371
	DMesh	0.611	0.504	0.545	0.530	0.556	0.537
	Improvement	69.7%	39.6%	51.8%	46.4%	52.3%	45.1%
4x4 Bit-reverse	NePA	0.388	0.384	0.385	0.386	0.387	0.387
	DMesh	1.000	1.000	1.000	1.000	1.000	1.000
	Improvement	157.7%	160.4%	159.7%	159.0%	158.3%	158.3%
4x4 Matrix transpose	NePA	0.422	0.395	0.392	0.392	0.390	0.387
	DMesh	0.709	1.000	1.000	1.000	1.000	1.000
	Improvement	68.0%	153.1%	155.1%	155.1%	156.4%	158.3%
8x8 Random	NePA	0.298	0.353	0.394	0.423	0.436	0.441
	DMesh	0.459	0.509	0.550	0.596	0.627	0.650
	Improvement	54.0%	44.1%	39.5%	40.8%	43.8%	47.3%
8x8 Bit-complement	NePA	0.090	0.090	0.090	0.090	0.090	0.144
	DMesh	0.232	0.221	0.230	0.239	0.244	0.246
	Improvement	157.7%	145.5%	155.5%	165.5%	171.1%	70.8%
8x8 Bit-reverse	NePA	0.178	0.174	0.175	0.187	0.199	0.204
	DMesh	0.312	0.309	0.304	0.313	0.307	0.311
	Improvement	75.2%	77.5%	73.7%	67.3%	54.2%	52.4%
8x8 Matrix transpose	NePA	0.167	0.174	0.174	0.181	0.184	0.184
	DMesh	0.317	0.322	0.327	0.373	0.379	0.445
	Improvement	89.8%	85.0%	87.9%	106.0%	105.9%	141.8%



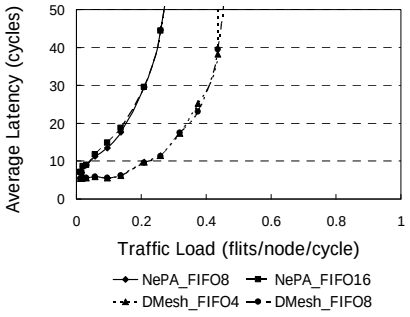
(a) 4x4 random traffic (b) 4x4 bit complement traffic (c) 4x4 bit reverse traffic (d) 4x4 matrix transpose traffic

(a) 8x8 random traffic (b) 8x8 bit complement traffic (c) 8x8 bit reverse traffic (d) 8x8 matrix transpose traffic

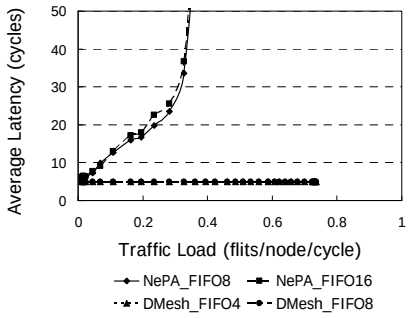
Figure 8. Comparisons of average latency in 4x4 and 8x8 mesh networks (FIFO depth = 4).



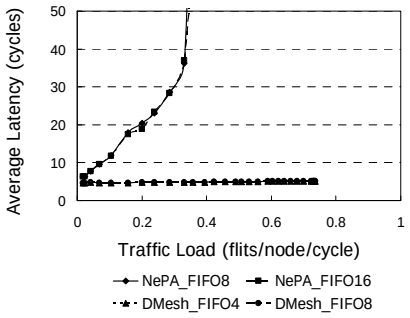
(a) 4x4 random traffic



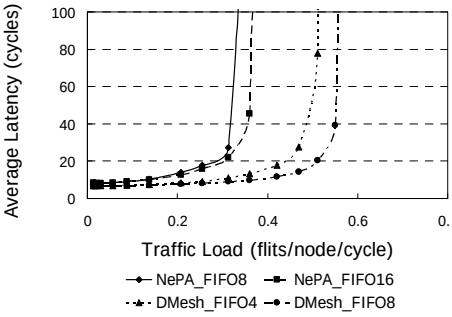
(b) 4x4 bit complement traffic



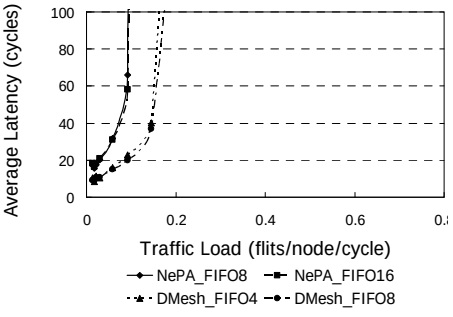
(c) 4x4 bit reverse traffic



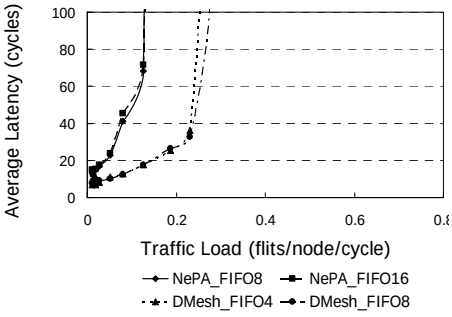
(d) 4x4 matrix transpose traffic



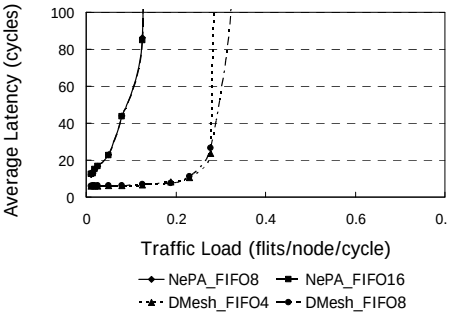
(a) 8x8 random traffic



(b) 8x8 bit complement traffic



(c) 8x8 bit reverse traffic



(d) 8x8 matrix transpose traffic

Figure 9. Comparisons of average latency in 4x4 and 8x8 mesh networks with similar router cost.