IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 15, 2021, accepted December 1, 2021, date of publication December 27, 2021, date of current version January 10, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3138828

HFBN: An Energy Efficient High Performance Hierarchical Interconnection Network for Exascale Supercomputer

FAIZ AL FAISAL[1], M. M. HAFIZUR RAHMAN[2], AND YASUSHI INOGUCHI[3], (Member, IEEE)
[1]Department of Computer Science and Engineering, Green University of Bangladesh, Dhaka 1207, Bangladesh
[2]Department of Computer Networks and Communications, CCSIT, King Faisal University, Al Ahsa 31982, Saudi Arabia
[3]Research Center for Advanced Computing Infrastructure, Japan Advanced Institute of Science and Technology, Nomi, Ishikawa 923-1211, Japan

Corresponding author: Faiz Al Faisal (faisal@cse.green.edu.bd)

This work was supported by the Japan Society for the Promotion of Science (JSPS) KAKENHI Grant Number 20K11731.

ABSTRACT Supercomputers are trying to be eco-friendly using their main components like- interconnection networks, processors, and shared memory through high power efficiency (GFlops/watts). The reason for this is the exascale systems, are on the horizon, require a 1000 times performance improvement over the petascale computers and energy efficiency has attracted as the key factor for to achieve exascale system. The main contribution of this paper is to introduce a new hierarchical interconnection network; considering simulations at the million processing cores especially for the exascale system. Moreover, our designed network claims the supremacy of high network performance and low power consumption over the conventional networks and ensuring the utmost preferability for exaFLOPS system. On the other hand, the performance per watt metric, used for the TOP500 list, doesn't reflect the overall performance of a given system. Hence, one of the possible solutions to reach the next generation exascale performance is to redesign the ''Interconnection Network'', which holds the main responsibility for the intercommunication between the CPUs and also the power consumption for the supercomputers. This paper focuses on a redesigned new energy efficient interconnection network that mitigates the problems of high power consumption, longer wiring length, and low bandwidth issues. Our designed network (HFBN) has been compared against the Tofu network and in the case of 1M cores, HFBN can obtain about 87.26% better energy efficiency with uniform traffic, about 86.32% with perfect shuffle traffic, and about 92.98% with the bit-compliment traffic at the zero load latency.

INDEX TERMS Interconnection network, HFBN, routing, dynamic performance, power analysis.

I. INTRODUCTION

Recent supercomputers require to be eco-friendly, and to ensure the eco-friendly use of their resources, requires to be redesigning, manufacturing/engineering, using and disposing of computing equipment in a way to reduce their environmental impact. Lower energy consumption allows reducing the operational cost as well as the environmental impact of powering the computer [8]. Moreover, the requirement of exascale computing power is obvious to combat future generation challenges. To fight against COVID-19, the world's fastest supercomputer Fugaku is being used to determine the effectiveness of various drugs [2]. Fugaku supercomputer used Tofu interconnect with 7,299,072 cores and can achieve about 415PFLOPS requiring about 28,335kW power usage [3]. On the other hand, 5Dtorus network used in Blue Gene/Q supercomputer requires 6.6MW of electrical power in achieving 20PF/s performance with 1.57M processor core. However, this supercomputer will require about 330MW of electrical power for building the exascale system. Therefore, the energy efficiency of the supercomputers is the most important issue with continuing the other constraints like- inadequate static network performance, low scalability, low throughput & large network latency [4], [15], [19].

Network performances and power consumption of supercomputers are heavily affected by the interconnection networks and their processing nodes. Consequently, every supercomputer requires an interconnection network as an obvious choice. Since every computer chip has limited

The associate editor coordinating the review of this manuscript and approving it for publication was Vlad Diaconita.

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

processing power, sequential processors can't be a suitable choice. For example, an Intel Core i7-3630QM processor contained with 4 cores, 22nm fabrication process can achieve about 76.8GFlops through the requirement of 45W electrical power usage. However, the requirement for exascale computing will require about 13 million connected such processors. On the other hand, in MPC systems, the wiring complexity of the network for on-chip as well as the off-chip connections is the most considerable issue due to power usages and high network latency. The K-Computer requires a total cable length of about 1,000 kilometers [25]. Moreover, as per analysis on Infiniband QDR 40Gbps switch requires typically about 1W of electrical power for its per link [26]. Friedman shows 3D NoC requires less power usage than the 2D NoCs with shorter vertical links [7]. However, the on-chip networks consume about 50% of the total chip power and off-chip bandwidth is limited to the total number of outgoing physical links [36]. Hence, the requirement of the interconnecting network has a huge impact on building exascale systems.

Energy consumption is completely dominated by the costs of data movement. The most critical problem for 3D networks is the massive heat generation. On the other hand, 3D networks require a much higher number of off-chip connections than 2D networks (50% higher). Even the cost and the power usage for 3D networks are much higher than 2D networks. This consideration leads to 2D NoC architecture is an obvious choice for exascale supercomputing. Even the Sunway supercomputer used the 2DMesh network for considering the exascale system [6], [11]. However, hierarchical networks are preferable over the flat networks due to the hierarchical design for the modern MPC systems, and the dynamic communication performance for many hierarchical interconnection networks are not capable enough to support exascale MPC network. Hence, in this research, we are considering a 2D NoC based hierarchical network (HFBN) as the interconnect for next-generation exascale system.

The rest of the paper describes the architecture of HFBN, reviews the routing algorithm, shows the static performance analysis, then evaluates the dynamic communication performance of HFBN, and finally, power estimation along with the energy usage for HFBN.

II. RELATED WORKS

Chip Multiprocessors (CMPs) usually adopt flat interconnects like- Mesh and Torus, which consume an increasing fraction of the chip power [11], [12]. Moreover, as technology advances and voltage continues to scale down, static power consumes a large fraction of the total power. Hence, reducing the total power usage is increasingly important for energy proportional computing. Energy efficiency ensures the reduced amount of energy required to provide suitable performance. The power usage effectiveness (PUE) of the Swiss Supercomputer (CSCS) datacenter was 1.8 in 2012 [9], however, the current PUE is about 1.25; a factor of $\tilde{1}.5$ improvements. With the modern advancements, the biggest

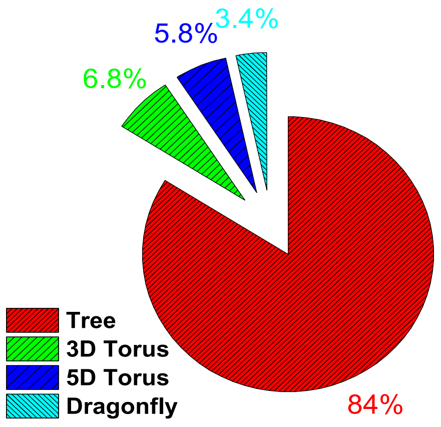


FIGURE 1. Existing network topology for system shares [5].

concern for supercomputers is power dissipation. Tianhe-1A supercomputer consumes 4.04 megawatts (MW) of electrical power, costing about \$ 0.10/kWh will require \$ 400 an hour and even about \$ 3.5 million per year [37].

System cost of MPC systems is highly correlated with an increased number of wiring interconnect [52]. High degree flat networks require a high number of wiring complexity, which increases the system performance and also increases the system cost. Modern supercomputers like- Blue Gene/Q considered high degree network (5DTorus) for its own interconnect [10]. However, the latest top-ranked supercomputer in 2021, supercomputer Fugaku achieving 442 petaFLOPS LINPACK benchmark performance considered the Tofu interconnect D for interconnecting 158,976 nodes having Fujitsu A64FX CPU (48+4 core) as per node [3]. Figure 1 shows the latest scenario of existing interconnects for massively parallel computer (MPC) systems [5]. This figure confirms that the vastly used network is the Tree network for the MPC systems, which has a big concern in the case of network performance. Figure 2 shows cost analysis for chip-chip links (level-2 links) and intra-rack links (level-3 links) (figure 2 is the link cost for the 4096 nodes). We considered the electrical links at the inter-chip level (x is the number of electric links) and optical links at the intra-rack level. This analysis explains that 2DMesh will require about 90.39% less amount of cost for designing level-2 and level-3 off-chip links than 4DTorus network. On the other hand, 2DMesh network also has low performance and high network congestion issues. Hence, the motivation for this research is not only to maximize network performance but also to minimize the network power usage.

In our latest paper, we had considered a three-dimensional on-chip network with two-dimension at the off-chip level called as 3D-TTN [16]. The main difference between HFBN and 3DTTN, HFBN which is based on a 2D structure whereas our latest paper of 3DTTN focuses on 3D structure at basic module (BM) level. And, from various observation, it is evident that 2D structures are less complex and more suitable for current computer systems even the Sunway supercomputer used 2Dmesh network [6]. Plus, figure 2

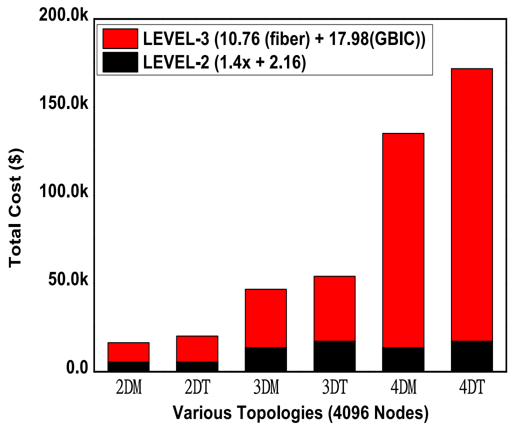


FIGURE 2. Link Cost analysis for various networks.

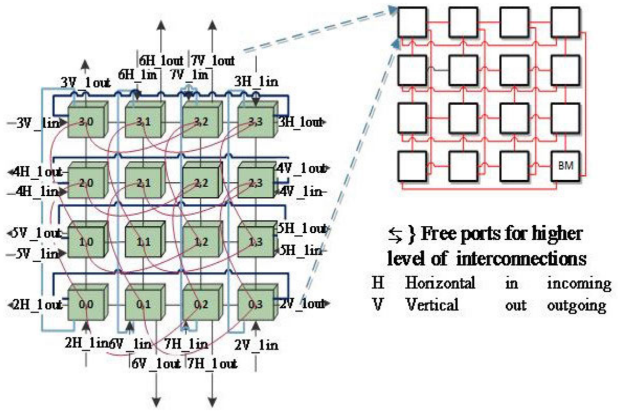


FIGURE 3. Interconnection of HFBN.

ensures the link cost for 2D networks is much lower than 3D networks, which motivates us to use a new 2D-based hierarchical network for this paper.

III. ARCHITECTURE OF HIERARCHICAL FLATTENED BUTTERFLY NETWORK (HFBN)

HFBN is a hierarchical network as it maintains different topological patterns at the different levels of network structure. The lowest network level is (level-1 network) is defined as the basic module for HFBN, where each core maintains a fixed number of radix and similar to 2D flattened butterfly architecture [32]. On the other hand, the upper level of HFBN is considered with 2DTorus network. Hence, we named our network the ''Hierarchical Flattened Butterfly Network (HFBN)''. This section defines the architectural pattern for HFBN, on-chip connections as well as off-chips. HFBN maintains particular higher-level link pattern along with the 2DTorus upper-level connectivity. On the other hand, the requirements for exascale system can be possible through interconnecting hundreds of millions of cores, which is certainly be possible by HFBN. However, HFBN requires pre-defined port assignments for its upper-level connectivity. Figure 3 illustrates the interconnection philosophy of HFBN. However, we defined HFBN through the definition of network structure at various network levels, and equations to obtain a fixed structure of HFBN is given as below-

Topological Definition: A HFBN(m, L, q) network, by definition, is built with constant radix (similar to 2D flattened butterfly network) at the lowest level of the network followed by the 2DTorus interconnection at the upper level (with the particular connectivity consideration); where L considered as the level of the hierarchy, q is the number of paired connectivity for each higher levels and m is any positive integer, which indicates the size of the basic module.

A HFBN(m, L, q) follows the exact definition for its certain level of connectivity-

Definition of HFBN Basic Module (BM): ($2^m \times 2^m$) is the lowest network level, (m is any positive integer)

Definition of HFBN Upper-level Connection: $L_{max} = \lceil 2 \cdot (2^m - 1)/q \rceil + 1$ is the maximum network size. Q_{max} = the maximum possible paired connectivity for any value of m; $Q_{max} = 2(2^m - 1)$, $1 \leq$ q $\leq Q_{max}$; depending on the value of q, HFBN(m, L, q) considers two set of configuration-

(a) IDEAL_HFBN(m, L, q) if $(2 \cdot (2^m - 1) mod\ q) = 0$

(b) PARTIAL_HFBN(m,L,q) if $(2 \cdot (2^m - 1) mod\ q)\ ! = 0$ (some exterior cores will be remained free).

A. LINK CONNECTIVITY

Basic module cores requires two digit for the formulation; the first is the Y-index, then the X-index. In general, in a Level-L HFBN, the core address can be represented by:

$$A^L = \left\{ (a_{yL}, a_{xL}) \quad \text{if } L_{max} \geq L \geq 1 \right.$$

More generally in a Level-L HFBN, the core address is represented by-

$$\begin{aligned} A &= A^L A^{L-1} A^{L-2} \ldots\ldots\ldots A^2 A^1 \\ &= (a_{2L-1}, a_{2L-2})(a_{2L-3}, a_{2L-4}) \ldots (a_3, a_2)(a_1, a_0) \quad (1) \end{aligned}$$

Here, the Level-1 is defined by core address (a_1, a_0), where a_1 defines the core address for the Y-axis and then the X-axis with the a_0. Higher level networks are two dimensional networks, hence we consider the first digit as the row index and then the second one is the column index. Now, if the address of a core N^1 included in BM_1 is represented as N^1 = $(s_{2L-1}, s_{2L-2}) \ldots\ldots (s_3, s_2)\ (s_1, s_0)$ and the address of a core N^2 included in BM_2 is represented as $n^2 = (d_{2L-1}, d_{2L-2}) \ldots\ldots (d_3, d_2)\ (d_1, d_0)$. The core N^1 in BM_1 and N^2 in BM_2 are connected if the following connections are satisfied for N^2:

- Link for BM-

  $(s_{2L-1}, s_{2L-2}) \ldots\ldots (s_3, s_2)\ (s_1, s_0)$ to $(s_{2L-1}, s_{2L-2}) \ldots\ldots (s_3, s_2)\ (s_1, s_0 \pm 1 \mod 2^m)$ where $2^m > s_0 >= 0$,

  $(s_{2L-1}, s_{2L-2}) \ldots\ldots (s_3, s_2)\ (s_1, s_0)$ to $(s_{2L-1}, s_{2L-2}) \ldots\ldots (s_3, s_2)\ (s_1 \pm 1 \mod 2^m, s_0)$ where $2^m > s_1 \geq 0$

$(s_{2L-1},\ s_{2L-2})\ldots\ldots(s_3,\ s_2)\ (s_1,\ s_0)$ to $(s_{2L-1},\ s_{2L-2})\ldots\ldots(s_3, s_2)\ ((s_1 + 2^m/2) \bmod 2^m, s_0)$
$(s_{2L-1},\ s_{2L-2})\ldots\ldots(s_3,\ s_2)\ (s_1,\ s_0)$ to $(s_{2L-1},\ s_{2L-2})\ldots\ldots(s_3, s_2)\ (s_1, (s_0 + 2^m/2) \bmod 2^m)$

- Link for Higher Level Vertical Connections-

$$DV = ((BMN^{L-1} \times 2^m) \pm x) \bmod BMN^L$$

- Link for Higher Level Horizontal Connections-

$$DH = (BMN^{L-1} \pm x) \bmod (BMN^{L-1} \times 2^m)$$

In case of higher level links, BMN ($BMN = (2^m \times 2^m)$) defines the number of cores in a basic module and L defines the level number of corresponding levels. Considering m is 2, then the BMN = 16. On the other hand, x defines the source core number which is equal to $(2^m \cdot s_{2L-1}, s_{2L-2}) \times BMN^{L-1} + \ldots\ldots + (2^m \cdot s_3 + s_2) \times BMN + (2^m \cdot s_1 + s_0)$. The highest level of network, which can be obtained by a $(2^m \times 2^m)$ BM is defined by $L_{max} = \lceil 2 \cdot (2^m - 1)/q \rceil + 1$. Finally, DV and DH gives the core number that is vertically or horizontally connected with the source core number x in respective. Algorithm 2 shows the port assignment for upper-level connectivity for a particular basic module (the flowchart of this algorithm (Appendix A) is given in figure 22 in the Appendix B). This algorithm considers all of the exterior cores in the on-chip network to be interconnected with the other on-chip module. Algorithm 2 requires the input value of m and q. On the other hand, L_{max} can be calculated from m and q. Function HIGHERLevel_HFBN(m,q) allocates the particular core in the basic module for high-level port connectivity, which requires the value of m, the possible number of paired connectivity for each level. As the high-level ports are being allocated by the exterior cores of each basic module, this algorithm allocates each possible port position for the higher level connectivity. In the initialize function, L_{max} has been calculated and BMAX is the number of possible cores in each X or Y direction.

B. BASIC MODULE (BM) OF A HFBN

HFBN(m, L, q) used only six intra-chip links for the interconnection of the basic module. Hence, the on-chip design for HFBN and the flattened butterfly has a distinctive difference. However, the basic module design for HFBN(2, L, q) follows the same pattern as the flattened butterfly network. Since HFBN maintains a constant node degree, the HFBN link pattern varies (link connectivity has already been defined beforehand for the basic module) from flattened butterfly when m is greater than 2. The lowest level of HFBN(m, 1, q) has been considered as the "Basic Module" (BM). HFBN considers $(2^m \times 2^m)$ number of cores as his basic module size. Hence, m = 2 means the possible number of cores at the BM level will be sixteen. Figure 3 also shows the basic module of HFBN(2, L, q). This network is based on the two-dimensional architecture and hence, BM connectivity is considered with

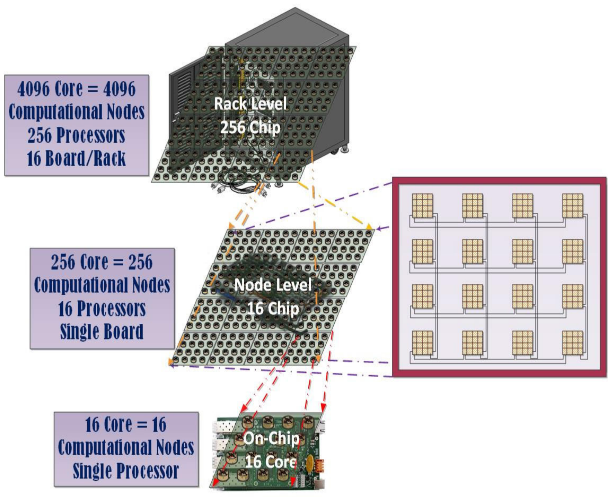


FIGURE 4. Higher-level Interconnection for HFBN(2, L, q).

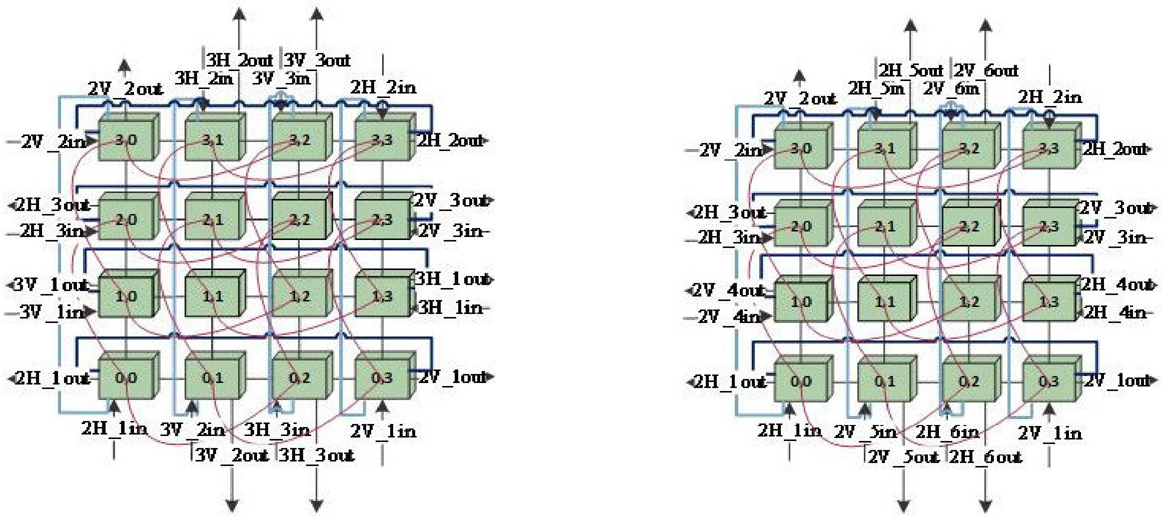
X and Y directions. However, we have already discussed the link connectivity for the basic module beforehand.

C. UPPER LEVEL OF A HFBN

Integrating a large number of on-chip links is useful for network performance as well as cost-effective. However, the scenario for the off-chip level is completely different, where per-link cost and power requirement increase simultaneously with the total system cost. Hence, we have considered the hierarchical design for the next generation supercomputer architecture, where particular level links are interconnected for each level. The higher level of HFBN considers 2DTorus interconnection considering recursive interconnect patterns of the immediate lowest level of subnetworks. Hence, a level-2 (node level) HFBN consists of a certain number (2^{2m}) of level-1 networks. This statement constitutes that an HFBN(2, L, q) will have 16 level-1 networks or basic modules for the complete level-2 network. Figure 4 shows the formation of a single level-3 network through the combination of 16 Level-2 networks and 256 Level-1 networks of HFBN(2, L, q). On the other hand, we have considered multiple lemmas to increase the readability of the network setup.

Lemma 3.1: A $(2^m \times 2^m)$ basic module of HFBN has $2^3 \cdot (2^m - 1)$ numbered of free ports for the higher-level interconnectivity.

One of the important points to achieve high performance, HFBN must use all its free ports. Hence, the large number of paired connectivity is highly effective. q is defined as the number of paired connectivity for each higher level and Q_{max} is the max paired value for any m; Maximum paired value for any m is defined as the $Q_{max} = 2(2^m - 1)$ and the q is all the possible divisors of Q_{max}, $Q_{max} = 2(2^m - 1) = 2(2^2 - 1) = 6$. In addition, the highest level network of HFBN can be defined as $L_{max} = ceil(2 \cdot (2^m - 1)/q) + 1$. Hence, HFBN(2, L, 1) can be constructed as the maximum seven network-level $L_{max} = (2 \cdot (2^2 - 1)/1) + 1 = 7$. The number of paired connectivity, q is responsible



(a) a 4 x 4 BM customised for HFBN(2,3,3)

(b) a 4 x 4 BM customised for HFBN(2,2,6)

FIGURE 5. Increased paired connectivity for each level.

TABLE 1. Generalization of HFBN.

Basic Module	Number of Paired Connectivity	Max Levels, L_{max}	Number of Cores
$(2^m \times 2^m)$	$1 \leq q \leq Q_{max}$	$\lceil 2 \cdot (2^m - 1)/q \rceil + 1$	$N_L = 2^{2mL}$

TABLE 2. Possible number of cores with various levels of HFBN.

m	q	L	Number of Cores
2	1	3	$N_3 = (2^{2 \cdot 2 \cdot 3}) = 4{,}096$
2	2	3	$N_3 = (2^{2 \cdot 2 \cdot 3}) = 4{,}096$
2	1	4	$N_4 = (2^{2 \cdot 2 \cdot 4}) = 65{,}536$
2	2	4	$N_4 = (2^{2 \cdot 2 \cdot 4}) = 65{,}536$
2	1	5	$N_5 = (2^{2 \cdot 2 \cdot 5}) = 1{,}048{,}576$
2	1	6	$N_6 = (2^{2 \cdot 2 \cdot 6}) = 16{,}777{,}216$
2	1	7	$N_7 = (2^{2 \cdot 2 \cdot 7}) = 268{,}435{,}456$
3	1	5	$N_5 = (2^{2 \cdot 3 \cdot 5}) = 1073{,}741{,}824$ (> 1073 millions)

for the increase of the number of outgoing and incoming connections at each off-chip level. An increased value of q, increases the number of in/out connections and decreases the maximum network level. Figure 5 shows the architectural design of HFBN(2, 3, 3) and HFBN(2, 2, 6). Those figures also show that the choice of off-chip connectivity of a particular core with the paired connectivity number (starting 1 to 3 for Figure 5(a) and 1 to 6 for Figure 5(b)).

Lemma 3.2: The total number of the cores at each level of HFBN can be defined as $N = 2^{2mL}$

HFBN maintains a fixed number of cores at the basic module level ($2^m \times 2^m$) and builds the upper level with having (2^{2m}) immediate lower level of subnetworks, which finally constitutes the network size of a particular number of interconnected cores. A HFBN(2, 3, q) has a network size of 4096 cores. Table 1 generalizes the architectural parameter for HFBN(m, L, q). Table 2 compares the various levels of HFBN for m = 2 with different q values. Moreover, in order to simplify the result analysis on HFBN, we have considered the HFBN(2, L, 1) class for this paper.

D. NUMBER OF LINKS AT VARIOUS NETWORK LEVELS OF A HFBN

Interconnection network has one of the greatest concerns for its own router layout [25], [34]. Hence, the number of on-chip, as well as the off-chip connection, will be a major concern in designing the exascale system. Figure 4 shows the hierarchical structure for HFBN(2, L, q), where level-1 network constructs at the chip level, the level-2 network will be used for node level, and level-3 network can be used at the rack level. The number of interconnecting links at various layers of HFBN can be defined by Equation 2.

$$\mathbf{L_L} = \mathbf{N_{BM}} \cdot \mathbf{IL_1}\ \text{Links} + \sum_{\mathbf{i=2}}^{\mathbf{N}} \mathbf{N_{BM}} \cdot \mathbf{OL_i}\ \text{Links} \qquad (2)$$

Here, N_{BM} is the number of basic module in current level, IL_1 links considers for number of inner level-1 links and OL_i considers the number of outer i-th level links. Here, the level-1 HFBN(2, 1, 1) network requires about 48 links for its BM. We have also generalized this equation in Table 3. On the other hand, Table 4 shows the total number of links for HFBN, in which the required number of links for HFBN is above only 800 at the node layer. Table 4 also shows the link comparison on various networks like- 2DMesh, 2DTorus against the HFBN. And, from this table, we can also find that 2DMesh and 2DTorus networks require a much higher number of links than the HFBN at the higher levels. HFBN(2, 3, 1) will require about 18.75% fewer interconnected links than the 2DTorus and about 17.46% fewer than 2DMesh.

IV. ROUTING ALGORITHM FOR HFBN

Modern supercomputer *BlueGene*/*L* uses deterministic routing along with the adaptive routing. Hence, in our performance analysis, we also considers a simple deterministic

TABLE 3. Generalization on total connected links at various levels of HFBN.

Topology	Level-1 Network	Higher Level Network
HFBN	(# of X-dir. Links) + (# of Y-dir. Links)	(# of BM in current level) × (# of inner links in level-1 network) + $\sum_{i=2}^{L}$ { (# of BM at current level, L_L) × (# of outgoing links at each higher-level, L_i)} [N ≥ 2]

TABLE 4. Total number of links at various levels networks.

Topology	Level-1 Net. (16 Cores)	Level-2 Net. (256 Cores)	Level-3 Net. (4096 Cores)
2DMesh	24 links	480	16128
2DTorus	32 links	512	16384
HFBN(2,L,1)	48 links	$16 \cdot 48 + 32 = 800$	13312

routing (*dimension−order* routing (DOR)) for HFBN(2, L, 1) class. *Dimension − order* routing continues to route the packet to the same dimension until the distance of that dimension become zero. Now considering HFBN routing, routing Algorithm 1 can be subdivided into two parts; one part considers the *BM_routing* and another part considers higher-level routing (*Routing_HFBN*). If the packet is destined for the other BM, the source core will send the packet to the *outlet_core* of the next interconnected BM of the current network level. On the other hand, *receiving_core* is used to track down the new source core address after the BM transfer has been completed. If the packet is destined to another BM, the source core sends the packet to the outlet core. Suppose, source core address is s = (s_{2L-1}, s_{2L-2}) (s_{2L-3}, s_{2L-4}) … … (s_3, s_2) (s_1, s_0) and destination core $d = (d_{2L-1}, d_{2L-2})$ (d_{2L-3}, d_{2L-4}).. … (d_3, d_2) (d_1, d_0) considering the routing at the Y,X direction for the higher levels as well as for the level-1 networks. Similarly, routing tag can be defined as t = (t_{2L-1}, t_{2L-2}) (t_{2L-3}, t_{2L-4}). . . . . . (t_3, t_2) (t_1, t_0), where t_i = $d_i - s_i$. In Routing_HFBN function, outlet_x and outlet_y are the function to get x coordinate s_0 and y coordinate s_1 of the core that link (s, d, l, dα) exists, where level l(2 ≤ l ≤ L), dimension d(d ∈ {V,H}) and direction $\alpha(\alpha \in \{+, -\})$.

We considered the DOR routing for HFBN, Figure 6 shows the routing flowchart for HFBN. For example- considering the routing for the source core (1,2),(1,2),(1,2) to destination core (2,1),(2,1),(2,1). Hierarchical routing for HFBN send the packet highest network level at first and condition for level-3 will be check. Hence, the packet will be routed to Level-3 network, the source core (1,2),(1,2),(1,2) will send the packet to the basic module outlet core (1,2),(1,2),(3,0) of Level-3 network and will reach Level-3(2,2) from Level-3(1,2) network, considering the route direction and updating the tag value. Similarly, after completing the level-2 routing, Level-1 routing will be started from (2,1),(2,1),(0,0) core and will reach destination core through Basic Module Routing.

Deadlock-free Routing for HFBN: Routing of packets requires to be deadlock-free, otherwise the packet will not be sent to the destination core ever and will delay the delivery of other packets, which in turn, drastically reduces dynamic communication performance. In this section, we studied the deadlock-freedom for HFBN. HFBN(2, 1, 1) network

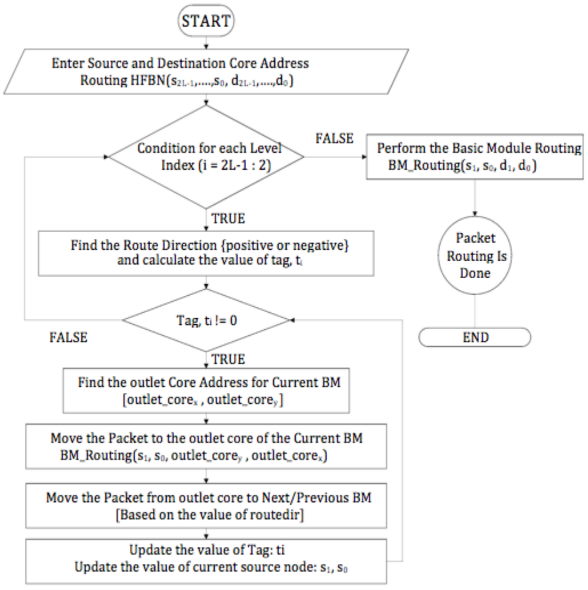

START
Enter Source and Destination Core Address
Routing HFBN(s2L-1,....,s0, d2L-1,....,d0)
Condition for each Level Index (i = 2L-1 : 2)
FALSE
Perform the Basic Module Routing BM_Routing(s1, s0, d1, d0)
TRUE
Find the Route Direction {positive or negative} and calculate the value of tag, ti
Packet Routing Is Done
Tag, ti != 0
FALSE
END
TRUE
Find the outlet Core Address for Current BM [outlet_corex , outlet_corey]
Move the Packet to the outlet core of the Current BM BM_Routing(s1, s0, outlet_corey , outlet_corex)
Move the Packet from outlet core to Next/Previous BM [Based on the value of routedir]
Update the value of Tag: ti
Update the value of current source node: s1, s0

FIGURE 6. Routing flowchart for HFBN.

maintains the core-core connections from each x or y (row/column) directional cores. Hence, there is no wrap-around routing is required for HFBN(2, 1, 1), which leads to a similar routing for the Mesh network (which requires only one VC for routing). This conclusion leads to only a single VC is required for the HFBN(2, 1, 1). On other hand, off-chip HFBN is constructed with the 2DTorus network arrangement. Hence, it requires 2 VCs for its deadlock-free. In summary, HFBN requires 2 VCs for its deadlock-free routing. However, using the required number of VCs, we could like to consider the proof for the HFBN deadlock-freeness based on the routing paths, which are divided into multiple states.

- **State 1:** Transfer of packet from the source core to outlet core of the Intra-BM.
- **State 2:** Transfer of packet for higher level.
  - **State 2.i.1:** Transfer of packet to the outlet core of Level (L - i) through the y-link for the Intra-BM.
  - **State 2.i.2:** Transfer of packet of Level (L - i) through the y-link for the Inter-BM.
  - **State 2.i.3:** Transfer of packet to the outlet core of Level (L - i) through the x-link for the Intra-BM.
  - **State 2.i.4:** Transfer of packet of Level (L - i) through the x-link for the Inter-BM.
- **State 3:** Transfer of packet from the receiving core to the destination core of the Intra-BM.

Lemma 4.1: if a message is routed in the order $y \rightarrow x$ in a 2DMesh network, then the network is deadlock free with 1 virtual Channel (VC) [33].

Lemma 4.2: if a message is routed in the order $y \rightarrow x$ in a 2DTorus network, then the network is deadlock free with 2 virtual channels (VC) [33].

Theorem 4.1: A HFBN is deadlock-free with 2 VCs.

Proof: The BM for HFBN(2, L, q) follows the flattened butterfly connection. Hence, this network level requires

Algorithm 1 Routing Algorithm for HFBN(2, L, 1)

```
Routing_HFBN($s_{2L-1}, s_{2L-2}, s_{2L-3}, \ldots, s_1, s_0, d_{2L-1}, d_{2L-2}, d_{2L-3}, \ldots, d_1, d_0$);
    tag: $t_{2L-1}, t_{2L-2}, t_{2L-3}, \ldots, t_1, t_0$;
      for i = 2L−1: 2;
          if $(((d_i - s_i + 2^m) \bmod 2^m) <= 2^m/2)$ then routedir = positive; $t_i = ((d_i - s_i + 2^m) \bmod 2^m)$;
          else routedir = negative; $t_i = (2^m - (d_i - s_i + 2^m) \bmod 2^m)$; endif;
          while $(t_i\ != 0)$ do
              if (i mod 2) = 1, then
                  $outlet\_core_x = outlet\_x(s, d, \lfloor(i)/2 + 1\rfloor, H, routedir)$;
                  $outlet\_core_y = outlet\_y(s, d, \lfloor(i)/2 + 1\rfloor, H, routedir)$;
              else
                  $outlet\_core_x = outlet\_x(s, d, \lfloor(i)/2 + 1\rfloor, V, routedir)$;
                  $outlet\_core_y = outlet\_y(s, d, \lfloor(i)/2 + 1\rfloor, V, routedir)$; endif;
              $BM\_routing(s_1, s_0, outlet\_core_y, outlet\_core_x)$;
              if (routedir = positive) then send the packet to the next BM;
              else move the packet to previous BM; endif;
              if $(t_i > 0)$ then $t_i = t_i - 1$; endif;
              if $(t_i < 0)$ then $t_i = t_i + 1$; endif;
              if (i mod 2) = 1,
                  $s_0 = receiving\_core_x(s, d, \lfloor(i)/2 + 1\rfloor, H, routedir)$;
                  $s_1 = receiving\_core_y(s, d, \lfloor(i)/2 + 1\rfloor, H, routedir)$;
               else $s_0 = receiving\_core_x(s, d, \lfloor(i)/2 + 1\rfloor, V, routedir)$;
                  $s_1 = receiving\_core_y(s, d, \lfloor(i)/2 + 1\rfloor, V, routedir)$; endif;
          endwhile;
      endfor;
      $BM\_routing(s_1, s_0, d_1, d_0)$;
end:

BM_routing($s_1, s_0, d_1, d_0$);
    source: $s_1, s_0$; destination: $d_1, d_0$;
      $BM\_tag$: $t_1, t_0$ = destination address$(d_1, d_0)$ − source address$(s_1, s_0)$;
      for i = 0: 1;
        if $(t_i > 0)$, movedir = positive;
        if $(t_i < 0)$, movedir = negative;
        if (movedir = positive & $t_i = 2^m/2$) then $t_i = t_i - 1$;
        if (movedir = positive & $t_i = -2^m/2$) then $t_i = t_i + 1$;
        if (movedir = negative & $t_i = 2^m/2$) then $t_i = t_i - 1$;
        if (movedir = negative & $t_i = -2^m/2$) then $t_i = t_i + 1$;
        if (movedir = positive & $t_i > 2^m/2$) then $t_i = t_i - 2^m$;
        if (movedir = negative & $t_i < -2^m/2$) then $t_i = t_i + 2^m$;
      endfor;
        while($t_0\ != 0$) do
          if($t_0 > 0$) move packet to +x direction w.r.t. destination; $t_0 = t_0 - 1$; endif;
          if($t_0 < 0$) move packet to -x direction w.r.t. destination; $t_0 = t_0 + 1$; endif;
        endwhile;
        while($t_1\ != 0$) do
          if($t_1 > 0$) move packet to +y direction w.r.t. destination; $t_1 = t_1 - 1$; endif;
          if($t_1 < 0$) move packet to -y direction w.r.t. destination; $t_1 = t_1 + 1$; endif;
        endwhile;
end;
```

TABLE 5. Static simulation environment.

	SGI UV3000
OS	SUSE Linux Enterprise Server 12
CPU	Intel Xeon E5-465v3
Core	6 core
Number of Threads	16
Compiler	Intel C++ Compiler 17.0.1

TABLE 6. Node degree of various networks.

2DMesh	2DTorus	TESH	TTN	RTTM	HFBN
4	4	4	6	4	8

only 1 VC, which is also proofed by the lemma 4.1. However, the higher level network is designed with a toroidal connection. Hence, it requires 2 VCs for the upper-level deadlock-free routing, and even if we considered the routing phase phase-1 and phase-3 require 1 VC for HFBN(2, L, q). However, phase-2 with the sub-phases considered the toroidal connectivity. Hence, the HFBN requires 2 VCs for this case. In summary, HFBN is deadlock-free with the two virtual channels.

V. STATIC NETWORK PERFORMANCE

Static network performance ensures the network capability without considering the packet movement. Hence, static network performance is useful in the initial choice of the network. A good network ensures low cost, low degree, low congestion, high connectivity, and high fault-tolerant rate than the others [45], [46]. The node degree is defined as the maximum number of physical outgoing links from a core. Since each core of the HFBN network has a maximum of eight outgoing links, the degree of HFBN is eight. Table 6 shows node degrees for the various networks. In this section, we compare the parameters, such as diameter, average distance, and cost analysis. We consider up to the level-5 network performance for HFBN(2, 5, 1). Hence, we use an SGI supercomputer with the OpenMP parallel programs run with the 6 core with 16 threads. Table 5 shows the simulation environment for static network performance.

A. DIAMETER PERFORMANCE

The diameter (Max. Hop Count) ensures the maximum number of channels is required for a packet to be sent from each source core to the destination core along its shortest path. However, static diameter doesn't consider the channel faults. Low diameter ensures low communication delay [29]. Hence, the low diameter is preferable for any interconnection network. Equation 3 shows the diameter evaluation for HFBN(m, L, q), and Table 7 shows the calculated formulation considering Equation 3. On the other hand, Figure 7 shows the diameter analysis of HFBN(2, L, 1) comparing with the various networks. This simulation ensures that the diameter performance of HFBN(2, L, 1) is much better than various hierarchical networks (Like- TTN [17], TESH [13]). In comparing the conventional networks, such as 2DMesh and 2DTorus, HFBN(2, 5, 1) shows much better results. Even HFBN(2, 5, 1) can achieve about 34.15% better than the TTN network. The diameter for HFBN can also be evaluated using

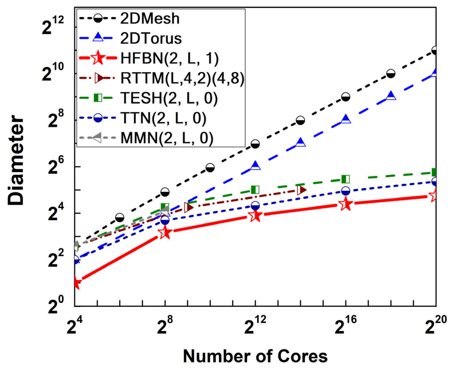


FIGURE 7. Diameter (Max. Hop Count) performance for various networks.

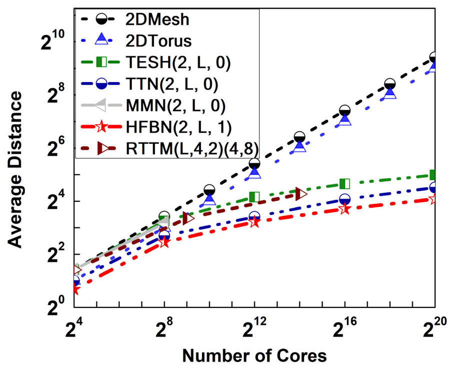


FIGURE 8. Average distance (Avg. Hop Count) for various networks.

Equation 3. For the HFBN, an upper bound for the diameter is given by-

$$\mathbf{Diameter} = \mathbf{max}(\mathbf{D_s} + (\sum_{\mathbf{i=2}}^{\mathbf{L}}(\mathbf{D_{si}} + \mathbf{D_i})) + \mathbf{D_d}) \qquad (3)$$

Here, where D_s = distance for highest level of outgoing core. D_{si} = distance for the next level of routing and D_i = distance for corresponding level of routing. D_d = distance from last level-2 core to destination core i.e. routing required at destination on-chip network. Table 7 shows the calculated formulation for HFBN up to level-4 network and in case of level-4 network, we need value of D_s (this value is for the routing at starting on-chip network), then the values of D_{si} and D_i (those values will depend on each inter-level routing i.e. routing distance for moving packets at lower network level to higher network level or vice versa) and finally, D_d is required for routing at destination on-chip network.

TABLE 7. Calculated formulation of diameter for HFBN.

Parameter HFBN(m,L,q)	D_s	Result of D_{si} and D_i	D_d	Each Level Diameter
HFBN(2, 1, 1)	2	$D_{si}=0$, $D_i=0$	0	2
HFBN(2, 2, 1)	2	for i = 2: D_{si}=0, D_i=5;	2	9

HFBN(2, 3, 1)	2	for i = 2: D_{si}=1, D_i=5;
		for i = 3: D_{si}=0, D_i=5;
			2	15

HFBN(2, 4, 1)	2	for i = 2: D_{si}=1, D_i=5;
		for i = 3: D_{si}=1, D_i=5;
		for i = 4: D_{si}=0, D_i=5
			2	21

B. AVERAGE DISTANCE

Diameter analysis considers the routing for a single packet with a maximum path required to traverse along its shortest distance [31], [44]. On the other hand, average distance (avg. hop count) considers the broadcasting of packets from each core to every other core. Hence, an average shorter path is preferable over the low diameter. The average distance is the mean distance between each distinct pair of cores. The small average distance allows small communication latency. The average distance of graph G can be defined by Equation 4, where n is the total number of cores in the network and d is defined as the diameter between all distinct pairs x and y. However, Figure 8 shows the average distance of various networks, which ensures that HFBN is superior over TTN, 2DMesh, 2DTorus, RTTM and TESH. On the other hand, for message transfer between nodes of a higher-level RTTM, a lot of packets need to pass through the 2DMesh basic module. The high hop count of a 2D mesh will result high hop distance for that source-destination pair. And, there will have many distinct pair nodes of the higher-level RTTM will traverse through the 2DMesh basic modules. Therefore, this high hop distances of many distinct source destination pair will incur high average distance. Apart from this, based on the other static parameters like- diameter and dynamic network performances it is evident that HFBN is more suitable than RTTM network. Table 8 shows the static parameter analysis for HFBN comparing with RTTM and other on-chip networks with 16 cores.

$$\mu(\mathbf{G}) = \frac{\sum_{\mathbf{x,y}\,\epsilon\,\mathbf{v}} \mathbf{d(G;\,x,\,y)}}{\mathbf{n(n-1)}} \qquad (4)$$

TABLE 8. Static analysis for on-chip Network(16 Cores).

Static Parameter	RTTM(1,4,2)	HFBN(2,1,1)	2DTorus	TTN
Diameter	6	2	4	4
Average Distance	2.66	1.6	2	2

C. STATIC COST PERFORMANCE

Cost performance analysis is vital for interconnection networks for its considerations of the product of node degree and diameter. The cost performance is inter-related due to inter-node distance, message traffic density, and fault tolerance. The node degree of a network can be considered as the maximum number of physical outgoing channels from a single core. The node degree of HFBN is 8. On the other hand, network radix can be defined by the number of channels for inter-router and the number of cores is connected to a single router. Hence, the network radix for HFBN is 9 (8 links are used to connect other routers, and a single link will be used for connecting the single-core). However, our network is flexible for connecting multiple cores from a single router. This feature allows high network scalability for

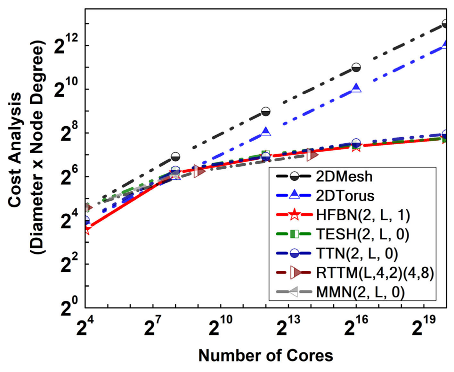


FIGURE 9. **Cost analysis for various networks.**

HFBN. Figure 9 shows the cost analysis for HFBN(2, L, 1), which shows that this network outperformed the 2DTorus and is even much better than the TTN network. On the other hand, RTTM [14] illustrates the low-cost performance and almost similar to HFBN due to its low node degree.

VI. EVALUATION ON EFFICIENT ENERGY USAGE

Efficient energy usage is an important consideration for MPC systems. As the modern MPCs are highly affected by power consumption, efficient energy usage can able to trace the system performance with respect to power usage which will be a key feature in the field of interconnection networks. In addition, in an Alpha 21364 microprocessor, the integrated routers and links consume about 20% of the total chip power (about 25W of total chip power 125W) [23]. Efficient energy usage is treated as the goal for reducing the amount of power usage which is required to maintain the suitable network performance. Regarding the definition of performance, it can be considered as the dynamic communication performance (DCP) of the corresponding network and the power usage leads to network power usage. In this section, all the DCP graphs are considered with the data flits only as of the accepted throughput. Equation 5 shows the consideration to obtain the network energy usage. Here, we have considered the single clock cycles as 1ns (as the system clock is 1GHz). Hence, network energy usage leads to average flits transfer time and the total power usage for transmitting the flits. On the other hand, efficient energy usage is the reduction of obtained network energy usage in comparing between two networks with the relative request-probability (r). In DCP analysis, packets are transmitted by the request-probability (r) during the simulation clock cycles. We have considered a wormhole simulator for the DCP analysis, which is specially designed for the hierarchical networks [35]. On the other hand, electrical power is considered up to inter-chip level (256 cores) and we have used the Orion energy model for this analysis [27]. Electrical power analysis considered with the various traffic patterns (used the Garnet 1.0 simulator for traffic generation [28] (used the default table-based routing)). Now, this analysis gives the required power usage for a single electrical module. Hence, to obtain the total electrical power, we have multiplied the single electrical module power with a total number of electrical modules. And then, to obtain the optical power, we have considered the fixed data-driven power of intra-rack link (0.0101 watts) [41], [50] and inter-rack link (0.035 watts) [41] along with the per gigabit interface converter (GBIC) module power (FG-TRAN-SFP28-SR (1.2 watts) [38]) for the optical off-chip connectivity. Total optical power usage can be obtained from the multiplication of the required number of optic links (intra-rack and the inter-racks) with its power usage and the required number of gigabit interface converter (GBIC) modules with its power usage. In this section, we have considered three traffic patterns analyses with various simulation conditions with respect to the number of computing cores.

$$\mathbf{N}EU = \mathbf{A}TT \cdot \mathbf{N}TPU \tag{5}$$

Here, NEU is defined as network energy usage, ATT as the average transfer time, and NTPU as the network total power usage. As the high degree networks require a large number of off-chip interconnect, it's not suitable for large MPC systems mainly due to the required power usage. A large-scale analysis is considered with 2 possible cases of 65K cores analysis and the 1M core analysis. In both the analysis, we have considered various traffic patterns for the result evaluation. This paper also shows the energy analysis of the Tofu network and the basic module (on-chip) for the Tofu network considers 12 cores. And hence, we can consider 240 cores for the inter-chip level and the total number of cores for the Tofu is considered with 4080 cores for the single rack level (for 65K the total number of cores for Tofu network is 65,280 cores and 1M analysis is consist of 1,044,480 cores), whereas rest of the networks considers 256 cores for there inter-chip level and 4096 cores as the total number of simulating cores for each rack. In the case of the RTTM network, we have considered ($a = 4$) and each upper-level network is constructed with (4×4) 16 lower level sub-networks with twisted torus connectivity to have the same number of cores for each possible case.

A. DYNAMIC COMMUNICATION PERFORMANCE

Network performance heavily depends on the variable traffic patterns. And, even the running applications on an MPC system are heavily affected by the traffic patterns. Dynamic Communication Performance for networks considers various traffic patterns and is characterized by latency and throughput. Latency refers to the time of a single packet to reach the destination core from its source core. On the other hand, Network throughput is the rate at which packets can be delivered by the network. It refers to the maximum amount of information delivered per unit of time through the network. Latency can be defined by the below equation-

$$\mathbf{T} = \mathbf{T_h} + \mathbf{L}/\mathbf{b}, \quad \mathbf{T_s} = \mathbf{L}/\mathbf{b} \tag{6}$$

Here, T_h is the header latency requires time to transmit the header message to traverse the network and T_s is the serialization latency, is the time for a packet of length L to cross a channel with bandwidth b.

B. DEFINITION OF VARIOUS TRAFFIC PATTERNS

Network load has a high effective influence on performance. Traffic patterns are responsible for the choice of a particular source and destination core in any network. Traffic patterns can be random or non-random selection. This paper considers the following non-uniform traffic patterns along with the uniform traffic patterns for dynamic communication performance (DCP) analysis.

Uniform- Here, every core sends message to every other core with equal probability, i.e., source and destination are randomly selected for each generated message.

Perfect Shuffle- This pattern is defined as the fixed source-destination pair for every message. The core with binary value $a_{n-1}, a_{n-2}, \ldots.., a_1, a_0$ communicate with $a_{n-2}, a_{n-3}, \ldots \ldots., a_0, a_{n-1}$ (rotate left 1 bit).

Bit-compliment- Fixed source-destination pair for every message. This case each core sends message to a such core with one's complement of its own address.

C. CONSIDERATION ON DYNAMIC COMMUNICATION PERFORMANCE

Dynamic communication performance (DCP) considers the latency and throughput of each network. Hence, dynamic communication performance shows the network zero load latency, saturation load, and maximum amount of packet can be delivered per unit of time through the network. Accepted throughput(Flits/Cycle/Core) is the number of flits that have been received at the destination cores with respect to the total number of cores and the total simulation cycle. Here, DCP graphs are considered with the data flits only as of the accepted throughput. On the other hand, the average transfer time (measured in clock cycles) is the average delivery period for all the delivered packets within the given simulation time. To evaluate the dynamic communication performance of HFBN, we considered a specially designed simulator [49]. This simulator is specially designed for hierarchical networks with the facility of explicitly changing the packet ID with the change of source router. Hence, the choice of a particular virtual channel is possible for different links in making the network deadlock-free. HFBN(2, 1, q) requires only 1 VC for its deadlock-free routing. However, the off-chip level of the HFBN network requires 2 VCs for its deadlock-free due to its torus connections. HFBN network considers the DOR routing for its dynamic performance. And, even we also considered simple dimension-order routing for the rest of the networks with wormhole flow control. We have considered the intel compiler with mcmodel(=medium) for the 1M cores and for 65K cores analysis, DCP simulation results are obtained from Visual C++ 2017 compilation.

Flow control is responsible for the allocation of resources in packets. In DCP analysis, a packet is a key component to ensure the network capability, who follows a certain route for reaching the destination core. The key resources in networks are the channels and buffers. Channels make sure the network connectivity and buffers are used to holding the packets temporarily at the cores. In the DCP analysis, we have considered the wormhole flow control. Wormhole flow control requires low buffering and most importantly, ensures latency independence over the message distance. In wormhole routing, each message is divided into packets, which are later divided into flits. Flits have consisted of two parts as header flits and data flits. Header flit holds the routing information and data flit follows the header flit through the network. On the other hand, DCP analysis considers only the deterministic routing for each network. Deterministic routing is also called oblivious routing. In deterministic routing, the same routing path will always be considered between the same source and destination pair even though multiple routing paths exist.

D. ESTIMATION OF POWER CONSUMPTION

Power consumption is the major concern for the exascale systems. Modern supercomputers are heavily affected by the on-chip as well as off-chip power usages. One of the powerful supercomputers Tianhe-2 requires 24MW of electrical power in achieving 33.86 petaFlops performance with more than 3 million of core [37]. Hence, the required power at the exascale level will be similar to a single nuclear power plant, which is unrealistic.

1) ASSUMPTION FOR THE POWER MODEL

The power consumption for the MPC system depends on various components such as network system, processor, memory module, and the cooling system. On the other hand, the network has a high impact on total power usage. The 16-tile MIT RAW on-chip network consumes about 36% of the total chip power [39]. Hence, on-chip power is the most important factor for estimating the total power usage. On the other hand, the required power at the rack level for each link is typically about 1W, bandwidth is over GB/s and cost is very high [26]. H. Wang and et. al. show the power comparison for high-speed electrical and optical interconnects for interchip communication [40]. Hence, off-chip power estimation is also required for the analysis of total power usage. In this paper, we have considered the fixed data-driven power of intra-rack link (0.0101 watts) [41] and inter-rack link (0.035 watts) [41] along with the per gigabit interface converter (GBIC) module power (FG-TRAN-SFP28-SR (1.2 watts) [38]) for the optical off-chip connectivity. Hence, our power model is considered with electric power at the inter-chip level and optic power at the intra-rack or the inter-rack level.

$$\begin{aligned} \mathbf{P}_{\text{total}} &= \mathbf{P}_{\text{electrical}} + \mathbf{P}_{\text{optical}} \\ \mathbf{P}_{\text{electrical}} &= \mathbf{P}_{\text{router}} + \mathbf{P}_{\text{link}} + \mathbf{P}_{\text{clock power}} \\ \mathbf{P}_{\text{optical}} &= \mathbf{P}_{\text{optical link}} + \mathbf{P}_{\text{GBIC module power}} \end{aligned} \tag{7}$$

2) ELECTRICAL POWER MODEL

Our power model is based on the Orion energy model [27] using 65nm fabrication process. We have used the GARNET

TABLE 9. Simulation environment for traffic analysis.

Parameter	Value	Units
Cores	1,048,576	-
Flow Control	wormhole switching	-
Packet Size	6 (+ 6 Header flits)	flits
Channel Buffer Size	4	-
Simulation Cycle	5,000	Clock cycles
Virtual Channels	6	-
link latency	1	clocks
Router Pipeline Cycle	1	clocks

1.0 NoC simulator [28] for analyzing dynamic power consumption. The power usage for the inter-chip network depends on the dynamic and leakage power usage. The router power is based on the router buffers, its local and global arbiter, and the crossbar traverse. The dynamic energy model of the router is being considered with Equation 8, where C is the capacitance, V is the supply voltage and finally α is the switching factor. Here, we have considered the default buffered input capacitance as 7.8e-15F for 65NM fabrication process and HVT transistor [27]. On the other hand, channels dynamic power are caused by the charging and discharging of capacitive loads, which is formulated as $P_{dy\_link} = \alpha C_1 V_{dd} f_{clk}$, where C_1 is the load capacitance, V_{dd} is the supply. We have considered 6 header flits along with the 6 data flits for the electrical power analysis. The header flits are merged with the data flits (8 bits for each data flits and 8 bits for each header flits). Hence, the total number of flits for the electrical power usage is considered as 6 flits per packet. And, to obtain the total electrical power, we have multiplied the obtained simulated power usage with the total number of inter-chip modules. The routing for this inter-chip analysis is considered with the default "Table-based" routing of the garnet 1.0 simulator.

$$\mathbf{E = 1/2\alpha CV^2} \quad [27] \tag{8}$$

E. EFFICIENT ENERGY USAGE ANALYSIS (1M Cores)

1M cores are considered up-to-the Level-5 network of HFBN(2,5,1). In this case starting with the Table 9 parameters for the 1M traffic analysis. This case we evaluated the power simulation with the same traffic condition concerning Table 10 and Table 11. Power analysis is obtained from parameters shown in Table 10 with the same accepted throughput (used as the injection rate in Garnet 1.0 simulator [28]), which is considered as the parameter for dynamic communication performance. In this section, we have also considered 256 cores for their inter-chip level and 1M cores as the total number of simulating cores.

1) UNIFORM TRAFFIC

The energy usage evaluation for the 1M cores, Table 9 shows the traffic parameters for the 1M cores evaluation, and Table 10 shows the power parameter for electrical analysis (Table 11 is for the optical link power usage). And finally, Figure 11 shows the energy analysis for MMN, RTTM, 2DMesh, 2DTorus, HFBN, TTN, and TESH networks considering the uniform traffic analysis for 1M cores showed

TABLE 10. Simulation environment for inter-chip model (electrical interconnect).

Parameter	Value	Units
Fabrication process	65	nm
Average link length	25	[mm]
Operating frequency	1×10^9	Hz
Transistor type	HVT	-
Supply voltage	1.0	V
Packet Size	6	flits
Simulation cycle	5,000	-
Virtual Channel	6	-
VA Model	VC_select	-
VA Buffer Model	SRAM	-
Buffer Size	4	-
CLOCK PIPELINE STAGE	2	-

TABLE 11. Consideration for optical interconnect.

Optical Power Evaluation	Value	Units
GBIC Module [FG-TRAN-SFP28-SR [38]]	1.2	watt
Intra-rack Link Power [41]	0.0101	watt
Inter-rack Link Power [41]	0.035	watt

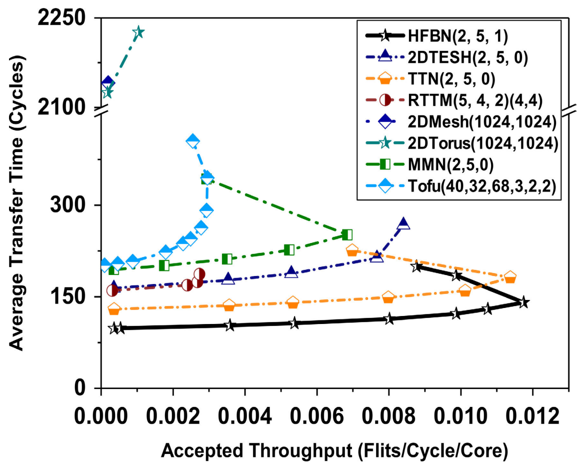


FIGURE 10. Uniform traffic analysis (1M Cores).

in Figure 10. Figure 11 explains that HFBN can obtain about 23.49% efficiency over the TTN, and compare with the Tofu(40,32,68,3,2,2) network, HFBN can obtain about 87.26% at the zero load latency. This analysis also ensures that the zero load latency and the network saturation rate for HFBN are superior to any other network.

2) PERFECT SHUFFLE TRAFFIC

Figure 13 shows the energy usage for perfect shuffle traffic, which also ensures the superiority of HFBN over any other network. At the zero load latency, 2DTorus network shows the worst energy usage among all the networks due to its high network latency (showed in figure 12) and the high number of off-chip interconnect compared to the 2DMesh network. In this case, HFBN can sure about 47.07% better efficiency before the network saturation in comparing with the TTN. However, the Tofu(40,32,68,3,2,2) network shows an efficiency difference of about 86.32% in comparison with the HFBN at the zero load latency.

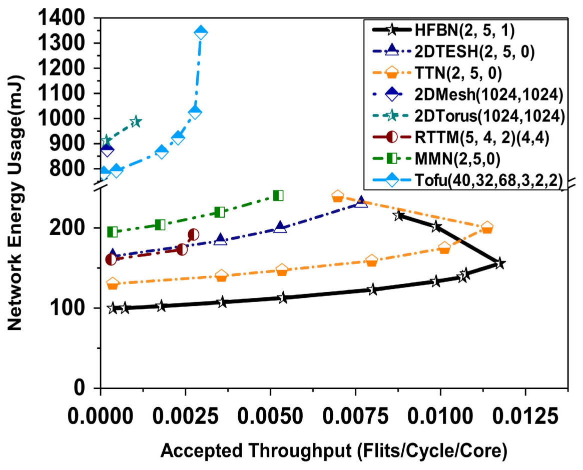


FIGURE 11. Energy usage with uniform traffic (1M Cores).

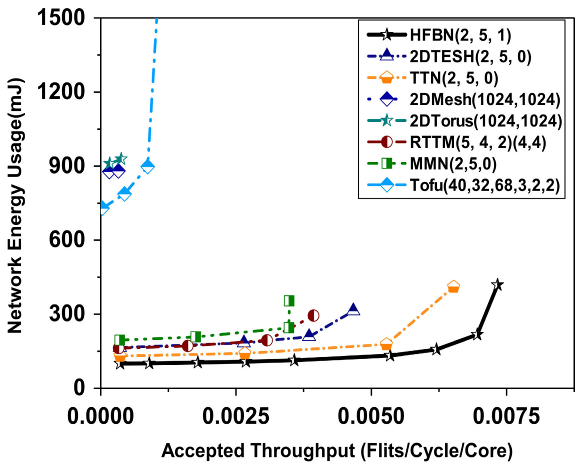


FIGURE 13. Energy usage with perfect shuffle (1M Cores).

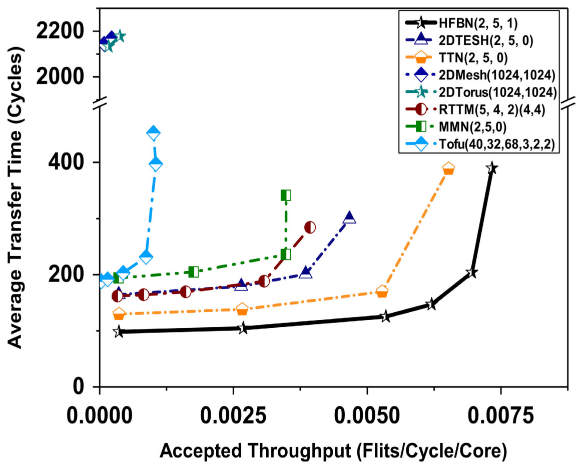


FIGURE 12. Perfect shuffle traffic analysis (1M Cores).

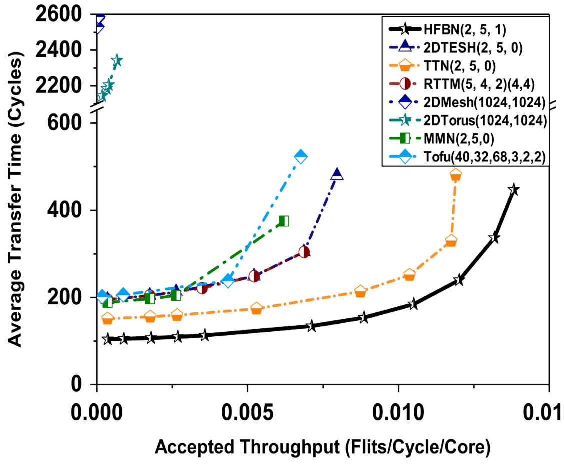


FIGURE 14. Bit-compliment traffic analysis (1M cores).

3) BIT-COMPLIMENT TRAFFIC

As the full system simulator (Gem5) has limited system scalability [42], [43], we have also considered the NAS parallel benchmarks [48] communication characteristics with the Message Passing Interface (MPI) implementation [47]. In static communication, compiled communication technique considers the compiler knowledge on application communication requirements and the provided network structure and allows to significantly optimize the performance of communications at the compile-time [48]. The communication pattern of MPI programs can be sub-divided into three types: static communications, dynamic communications, and dynamically analyzable communications. Static communications are those communications whose source and destination core are determined at the compile time. Dynamically analyzable selects source and destination core at runtime without incurring excessive overheads. Dynamic communication selects source and destination at only the runtime. However, the majority of communications in scientific programs maintain static communication. Hence, in this part of traffic analysis, we have considered the static communication pattern of MPI_Send, where the selection of all source-destination pairs needs to be determined at compile time. We have considered the bit-compliment traffic pattern with fixed source and destination for this analysis. Table 9 shows the parameter consideration for the traffic analysis. Figure 14 reviled the performance analysis with various networks along with the Figure 15 shows that HFBN can obtain about 30.76% better efficiency over the TTN in considering the bit-compliment traffic pattern.

F. EFFICIENT ENERGY USAGE ANALYSIS (65K Cores)

65K cores are considered up-to-the Level-4 network of HFBN(2, 4, 1). This case starting with the Table 12 parameters for the 65K traffic analysis. In this case, we evaluated the power simulation with the same traffic condition with respect to Table 13 and Table 11. Power analysis is obtained from parameters shown in Table 13 with the same accepted throughput (used as the injection rate in Garnet 1.0 simulator [28]), which is considered as the parameter for dynamic communication performance. In this section, we have also considered 256 cores for their inter-chip level

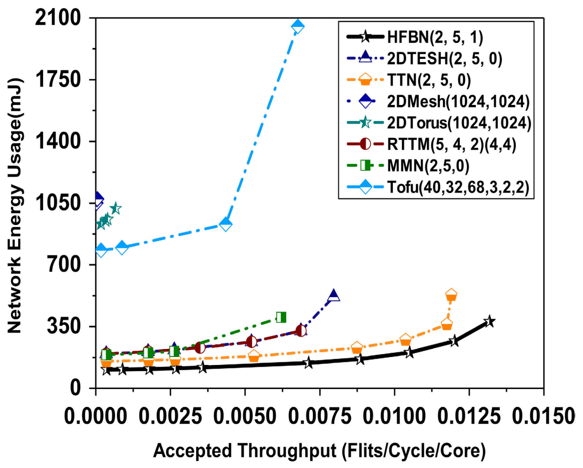


FIGURE 15. Energy usage with bit-compliment (1M Cores).

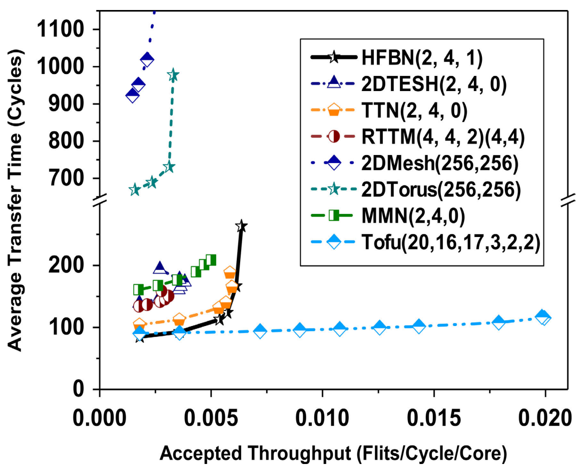


FIGURE 16. Uniform traffic analysis (65K Cores).

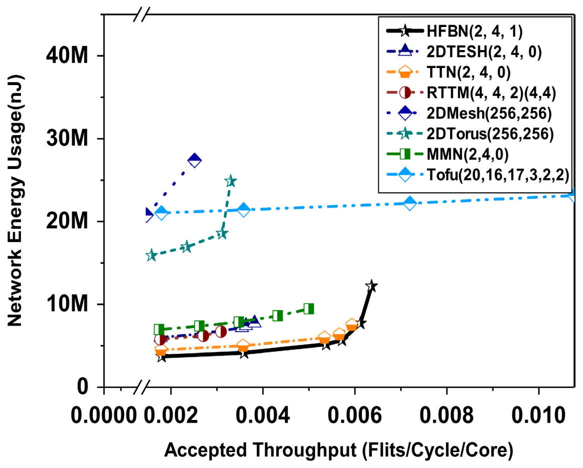


FIGURE 17. Energy usage with uniform traffic (65K Cores).

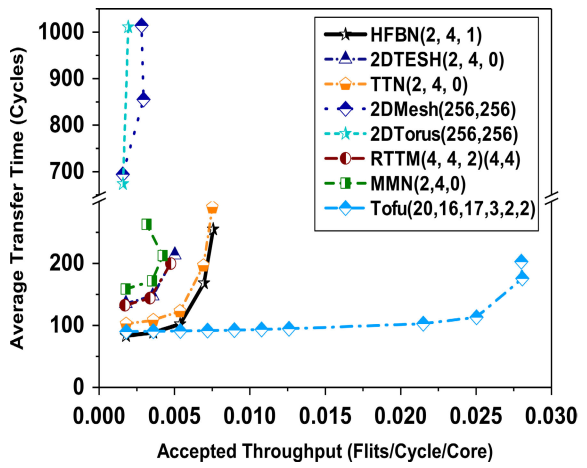


FIGURE 18. Perfect shuffle traffic analysis(65K Cores).

TABLE 12. Simulation environment for traffic analysis.

Parameter	Value	Units
Cores	65,536	-
Flow Control	wormhole switching	-
Packet Size	6 (+ 6 Header flits)	flits
Channel Buffer Size	2	-
Simulation Cycle	5000	Clock cycles
Virtual Channels	4	-
link latency	1	clocks
Router Pipeline Cycle	1	clocks

TABLE 13. Simulation environment for inter-chip model (electrical interconnect).

Parameter	Value	Units
Fabrication process	65	nm
Average link length	25	[mm]
Operating frequency	1×10^9	Hz
Transistor type	HVT	-
Supply voltage	1.0	V
Packet Size	6	flits
Simulation cycle	5000	-
Virtual Channel	4	-
VA Model	VC_select	-
VA Buffer Model	SRAM	-
Buffer Size	2	-
CLOCK PIPELINE STAGE	2	-

and 65,536 cores as the total number of simulating cores (other than the Tofu network).

1) UNIFORM TRAFFIC

In the case of 65K analysis, we have considered electrical power analysis up to inter-chip level, and we have considered the optical power for intra-rack level and inter-rack level. Figure 16 shows the traffic analysis for this case, and using Table 11 (optical connectivity) and Table 13 (electrical connectivity), we could obtain the power usage. Figure 17 shows the energy analysis for 65K uniform traffic, where HFBN could achieve about 22.24% better efficiency over the TTN. Comparing with the Tofu(20,16,17,3,2,2) network, even with the different network size of 65,280 cores, HFBN can also achieve much better efficiency (80.56%) with zero load latency.

2) PERFECT SHUFFLE TRAFFIC

The zero load latency for HINs always ensures the lowest latency over the conventional networks. Here, perfect shuffle traffic analysis with the 65K cores considers the same traffic

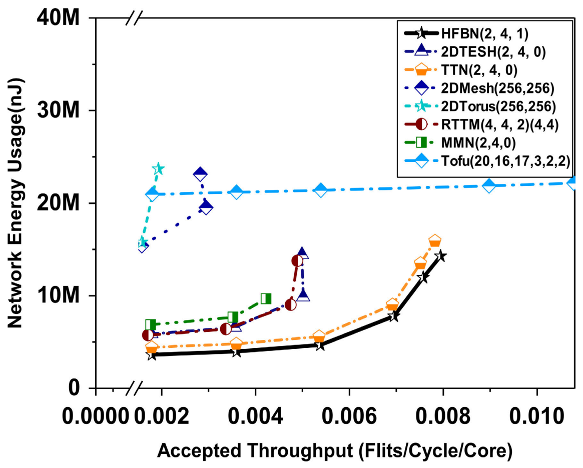


FIGURE 19. Energy usage with perfect shuffle (65K Cores).

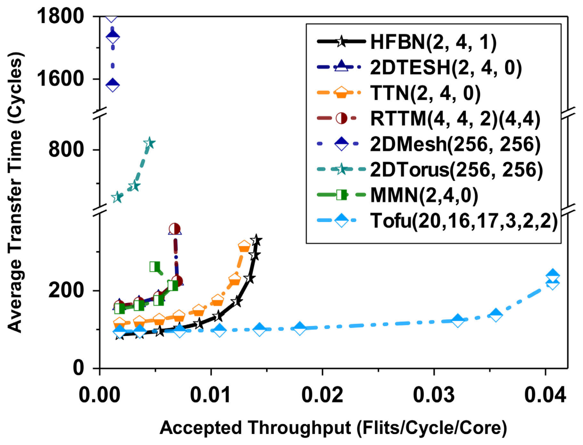


FIGURE 20. Bit-compliment traffic analysis (65K Cores).

parameters as Table 12 for 65K analysis. In addition to traffic analysis parameters, the parameter used (Table 13 and 11) for the power analysis is also the same as to analyze the network energy usage. Figure 19 shows the energy usage for Perfect Shuffle traffic, which also ensures the superiority of HFBN over any other network. 2DMesh and 2DTorus networks show the worst efficiency among all the 2D networks due to their high network latency showed in Figure 18. This analysis ensures that HFBN has superiority over TTN up to 18.39% with efficiency (Figure 19). Now, comparing with the Tofu(20,16,17,3,2,2) network, even with the different network sizes (65,280 cores), HFBN can achieve about 78.13% better efficiency with zero load latency.

3) BIT-COMPLIMENT TRAFFIC

Figure 21 shows the energy usage for Bit-compliment traffic, which also ensures the superiority of HFBN over any other network. 2DMesh and 2DTorus networks have provided the worst efficiency among all the 2D networks due to their high network latency showed in Fig. 20. This analysis ensures that HFBN has superiority over TTN up to 22.36% with energy efficiency (Figure 21). Comparing to Tofu network, even with the different network sizes, HFBN can achieve much better energy efficiency (72.09%) at zero load latency.

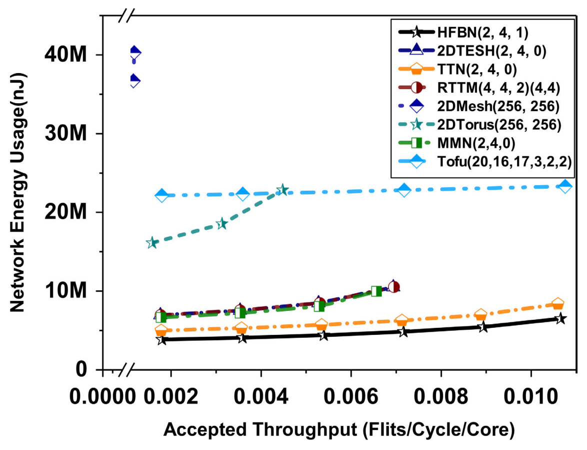


FIGURE 21. Energy usage for bit-compliment (65K Cores).

VII. CONCLUSION

In this research plan, our main objective was to find an interconnection network that requires low power usage and can achieve high network performance for many-core processors. High degree interconnection networks are useful for performance considerations but are not useful for low power usage. Hence, the target network was focused on efficient energy usage and achieving suitable network performance. Our new interconnection network is based on the 2D NoCs, entitled the Hierarchical Flattened Butterfly Network (HFBN). Considering the static performance analysis, HFBN can show much better performance than conventional networks. Even comparing with 5DTorus network, HFBN network can also achieve about 32.5% better diameter performance (over 1M cores) and about 66.67% with over 200M cores, almost similar average distance performance and even shows much better cost analysis at 1M cores. On the other hand, in considering the most recent high-performed hierarchical networks such as TTN [17], MMN [20], TESH [13], RTTM [14]; this network confirms itself as an obvious choice. In the case of diameter, HFBN can achieve about 34.15% better performance than TTN (with 1M cores); whereas in the case of average distance, HFBN can achieve about 26.14% better performance. Our energy evaluation considers various configurations starting from 65K cores to 1M cores, HFBN shows its uttermost superiority over any other network. Comparing with the high degree network like-Tofu; HFBN also confirms its superiority. In the case of 1M cores analysis of Tofu(40,32,68,3,2,2) network, HFBN can obtain about 87.26% better energy efficiency at the zero load latency with uniform traffic, about 86.32% with perfect shuffle traffic, and about 92.98% with the bit-compliment traffic. On the other hand, with the 65K core analysis of Tofu(20,16,17,3,2,2) network, HFBN can obtain about 80.56% better energy efficiency with uniform traffic, 78.13% better with the perfect shuffle traffic, and about 72.09% better with bit-compliment traffic at the zero load latency.

APPENDIX A
PORT ASSIGNMENT ALGORITHM

Algorithm 2 Higher Level Port Assignment for HFBN

```
HIGHERLevel_HFBN(m,q);
/* Main function to call with m, possible number of paired
connectivity for each level */
    initialize(q); /* Initializing the pre-defined ports */
    for lv = 2: $L_{max}$; /* lv defines the network level */
      for $i$ = 1: q;
      /* loop for making sure about all paired connectivity */
        $min\_dist = 65536$;
/* Now, assign the previous level Horizontal or Vertical port
address */
        if (lv != 2 and $i$ = 1)
          lvmh = $port\_levelQ(lv - 1,'H', q)$;
          lvmv = $port\_levelQ(lv - 1,'V', q)$;
        else
          lvmh = $port\_levelQ(lv - 1,'H', i - 1)$;
          lvmv = $port\_levelQ(lv - 1,'V', i - 1)$;
        endif;
        if (!availableQ(lv, 'V', i))
          for (x = 0, y = 0; x $\leq$ BMAX; x++, y++)
            SaveXY(x, y, &$save\_pv$, &$save\_ph$, lvmv,
lvmh, dist, &$min\_dist$);
          endfor;
          setHVLQ($save\_ph$.y, $save\_ph$.x, lv, 'H', i);
setHVLQ($save\_pv$.y, $save\_pv$.x, lv, 'V', i);
        endif;
      endfor;
    endfor;
end;

SaveXY(x,y,*$save\_pv$,*$save\_ph$, lvmv, lvmh, dist,
*$min\_dist$);
    if(LV[0][x] = 0)
      pv = $set\_xy(0, x)$; $ph$=$set\_xy(BMAX, x)$;
      dist = distance(pv, lvmh) + distance(ph, lvmv);
      if(dist < $min\_dist$)
        $save\_pv$ = pv; $save\_ph$ = ph; $min\_dist$ = dist; /*
save the position */
      endif;
      ph = $set\_xy(0, x)$; $pv$=$set\_xy(BMAX, x)$;
      dist = distance(pv, lvmh) + distance(ph, lvmv);
      if(dist < $min\_dist$)
        $save\_pv$ = pv; $save\_ph$ = ph; $min\_dist$ = dist; /*
save the position */
      endif;
    endif;
    if(LV[y][0] = 0)
      pv = $set\_xy(y, 0)$; $ph$=$set\_xy(y, BMAX)$;
      dist = distance(pv, lvmh) + distance(ph, lvmv);
      if(dist < $min\_dist$)
        $save\_pv$ = pv; $save\_ph$ = ph; $min\_dist$ = dist; /*
save the position */
      endif;
      ph = $set\_xy(y, 0)$; $pv$=$set\_xy(y, BMAX)$;
      dist = distance(pv, lvmh) + distance(ph, lvmv);
      if(dist < $min\_dist$)
        $save\_pv$ = pv; $save\_ph$ = ph; $min\_dist$ = dist; / *
save the position */
      endif;
    endif;
end;

typedef struct
    position x, y;
end;

distance(src, dest)
    dist.x = abs(src.x − dest.x);
    dist.y = abs(src.y − dest.y);
    return(dist.x + dist.y);
end;

availableQ(pl, phv, connect)
/* check the current level port is already assigned or not (pl
is the level number and phv is for the choice of vertical or
Horizontal selection and connect is to defined the current core
number) */
    for(p.y = 0; p.y $\leq$ BMAX; p.y++)
      for(p.x = 0; p.x $\leq$ BMAX; p.x++)
        if(pl = LV[p.y][p.x] && phv = HV[p.y][p.x] &&
connect = IN[p.y][p.x]) return true;
    return false;
end;

initialize(q)
/* Initializing the pre-defined ports */
    BMAX = $2^m - 1$;
    $L_{max}$ = ceil(2*($2^m$ − 1)/q) + 1;
    for(x = 0; x <= BMAX; x++)
      for(y = 0; y <= BMAX; y++)
        LV[y][x] = 0; HV[y][x] = '*'; IN[y][x] = 0;
if (q < 2)
      setHVLQ(0, 0, 2, 'H', 1); setHVLQ( 0, BMAX,
2, 'V',1); setHVLQ(BMAX, 0, 3, 'V',1); setHVLQ(BMAX,
BMAX, 3, 'H',1);
    else
      setHVLQ(0, 0, 2, 'H', 1); setHVLQ(0, BMAX, 2,
'V', 1);
      setHVLQ(BMAX, 0, 2, 'V', 2); setHVLQ(BMAX,
BMAX, 2, 'H', 2);
    endif;
end;

port_levelQ(pl, phv, connect)
/* check the previous level which is already assigned */
    for(p.y = 0; p.y $\leq$ BMAX; p.y++)
      for(p.x = 0; p.x $\leq$ BMAX; p.x++)
        if(pl = LV[p.y][p.x] && phv = HV[p.y][p.x] &&
connect = IN[p.y][p.x]) return(p);
end;

setHVLQ(y, x, pl, phv, connect) /* initialize */
    LV[y][x] = pl;
    HV[y][x] = phv;
    IN[y][x] = connect;
end;

set_xy(y, x)
    p.x = x; p.y = y;
    return(p);
end;
```

APPENDIX B
FLOWCHART FOR PORT ASSIGNMENT

Figure 22 shows the flowchart for the port assignment algorithm.

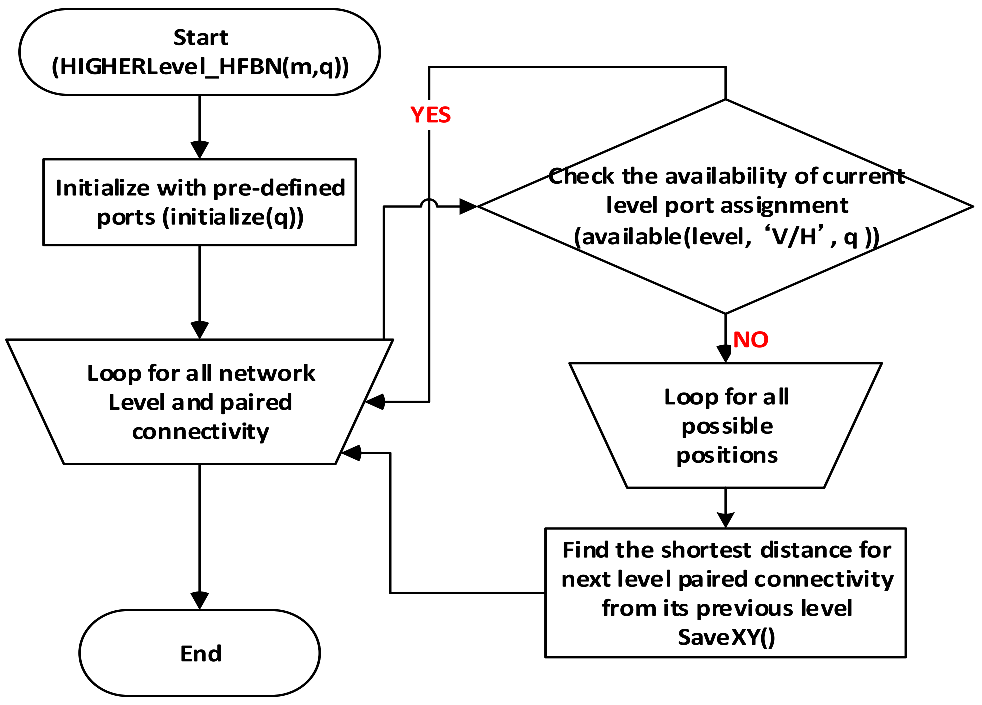

Start (HIGHERLevel_HFBN(m,q))
Initialize with pre-defined ports (initialize(q))
Loop for all network Level and paired connectivity
End
YES
Check the availability of current level port assignment (available(level, 'V/H', q))
NO
Loop for all possible positions
Find the shortest distance for next level paired connectivity from its previous level SaveXY()

FIGURE 22. **Flowchart for port assignment algorithm.**

ACKNOWLEDGMENT

The authors are grateful to the anonymous reviewers for their valuable and constructive suggestions and comments that guided them to enhance the quality and clarity of this manuscript.

REFERENCES

[1] Y. Ajima, T. Inoue, S. Hiramoto, Y. Takagi, and T. Shimizu, "The tofu interconnect," *IEEE Micro*, vol. 32, no. 1, pp. 21–31, Jan. 2012, doi: 10.1109/mm.2011.98.

[2] S. Kudo, K. Nitadori, T. Ina, and T. Imamura, "Implementation and numerical techniques for one EFlop/s HPL-AI benchmark on Fugaku," in *Proc. IEEE/ACM 11th Workshop Latest Adv. Scalable Algorithms Large-Scale Syst. (ScalA)*, Nov. 2020, pp. 69–76.

[3] Y. Kodama, T. Odajima, E. Arima, and M. Sato, "Evaluation of power management control on the supercomputer Fugaku," in *Proc. IEEE Int. Conf. Cluster Comput. (CLUSTER)*, Sep. 2020, pp. 484–493, doi: 10.1109/CLUSTER49012.2020.00069.

[4] D. Sanchez, G. Michelogiannakis, and C. Kozyrakis, "An analysis of on-chip interconnection networks for large-scale chip multiprocessors," *ACM Trans. Archit. Code Optim.*, vol. 7, no. 1, pp. 1–28, Apr. 2010.

[5] C. Minkenberg, "Interconnection network architectures for high-performance computing," in *Advanced Computer Networks—Guest Lecture*. Rüschlikon, Switzerland: IBM Research Zurich, May 2013.

[6] H. Fu, J. Liao, J. Yang, L. Wang, Z. Song, X. Huang, C. Yang, W. Xue, F. Liu, F. Qiao, W. Zhao, X. Yin, C. Hou, C. Zhang, W. Ge, J. Zhang, Y. Wang, C. Zhou, and G. Yang, "The sunway TaihuLight supercomputer: System and applications," *Sci. China Inf. Sci.*, vol. 59, no. 7, pp. 1–16, Jul. 2016.

[7] V. F. Pavlidis and E. G. Friedman, "3-D topologies for networks-on-chip," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 15, no. 10, pp. 1081–1090, Oct. 2007.

[8] *The Green 500 List, Energy-Efficient Supercomputers*. Accessed: Jul. 28, 2021. [Online]. Available: https://www.top500.org/lists/green500/

[9] *Swiss National Supercomputing Centre*. Accessed: Jul. 22, 2021. [Online]. Available: https://www.cscs.ch/

[10] N. R. Adiga *et al.*, "An overview of the BlueGene/L supercomputer," in *Proc. ACM/IEEE SC Conf. (SC)*, Nov. 2002, p. 60, doi: 10.1109/SC.2002.10017.

[11] A. Punhani, Nitin, and P. Kumar, "A modified diagonal mesh interconnection network," in *Proc. INDICON*, Dec. 2014, pp. 1–6.

[12] S. Konstantinidou and L. Snyder, "The chaos router," *IEEE Trans. Comput.*, vol. 43, no. 12, pp. 1386–1397, Dec. 1994.

[13] V. K. Jain, T. Ghirmai, and S. Horiguchi, "TESH: A new hierarchical interconnection network for massively parallel computing," *IEICE Trans. Inf. Syst.*, vol. E80-D, no. 9, pp. 837–846, 1997.

[14] Y. Liu, C. Li, and J. Han, "RTTM: A new hierarchical interconnection network for massively parallel computing," in *High Performance Computing and Applications* (Lecture Notes in Computer Science), vol. 5938. Berlin, Germany: Springer, 2010, pp. 264–271.

[15] F. A. Faisal, M. M. H. Rahman, and Y. Inoguchi, "A new power efficient high performance interconnection network for many-core processors," *J. Parallel Distrib. Comput.*, vol. 101, pp. 92–102, Mar. 2017.

[16] F. A. Faisal, M. M. H. Rahman, and Y. Inoguchi, "3D-TTN: A power efficient cost effective high performance hierarchical interconnection network for next generation green supercomputer," *Cluster Comput.*, vol. 24, no. 4, pp. 2897–2908, May 2021.

[17] M. M. H. Rahman, Y. Sato, and Y. Inoguchi, "High and stable performance under adverse traffic patterns of tori-connected torus network," *Comput. Electr. Eng.*, vol. 39, no. 3, pp. 973–983, Apr. 2013.

[18] M. M. H. Rahman and S. Horiguchi, "HTN: A new hierarchical interconnection network for massively parallel computers," *IEICE Trans. Inf. Syst.*, vol. E86-D, pp. 1479–1486, Sep. 2003.

[19] M. M. H. Rahman, Y. Inoguchi, F. A. Faisal, and M. K. Kundu, "Symmetric and folded tori connected torus network," *J. Netw.*, vol. 6, no. 1, p. 26, Jan. 2011.

[20] M. R. Awal, M. M. H. Rahman, and M. A. H. Akhand, "A new hierarchical interconnection network for future generation parallel computer," in *Proc. 16th Int. Conf. Comput. Inf. Technol.*, Khulna, Bangladesh, Mar. 2014, pp. 314–319.

[21] M. Yokokawa, F. Shoji, A. Uno, M. Kurokawa, and T. Watanabe, "The k computer: Japanese next-generation supercomputer development project," in *Proc. IEEE/ACM Int. Symp. Low Power Electron. Design*, Aug. 2011, pp. 371–372, doi: 10.1109/ISLPED.2011.5993668.

[22] IEEE Spectrum. *Power Problems Threaten to Strangle Exascale Computing*. Accessed: Mar. 24, 2017. [Online]. Available: https://spectrum.ieee.org/power-problems-threaten-to-strangle-exascale-computing

[23] S. S. Mukherjee, P. Bannon, S. Lang, A. Spink, and D. Webb, "The alpha 21364 network architecture," in *Proc. HOT Interconnects. Symp. High Perform. Interconnects*, 2001, pp. 113–117.

[24] *Blue Gene Supercomputer*. Accessed: May 28, 2021. [Online]. Available: https://www.ibm.com/ibm/history/ibm100/us/en/icons/bluegene/

[25] I. Fujiwara, M. Koibuchi, and H. Casanova, "Cabinet layout optimization of supercomputer topologies for shorter cable length," in *Proc. 13th Int. Conf. Parallel Distrib. Comput., Appl. Technol.*, Dec. 2012, pp. 227–232.

[26] M. J. Koop, W. Huang, K. Gopalakrishnan, and D. K. Panda, "Performance analysis and evaluation of PCIe 2.0 and quad-data rate InfiniBand," in *Proc. 16th IEEE Symp. High Perform. Interconnects*, Aug. 2008, pp. 85–92.

[27] A. B. Kahng, B. Li, L.-S. Peh, and K. Samadi, "Orion 2.0: A power-area simulator for interconnection networks," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 1, pp. 191–196, Jan. 2012.

[28] N. Agarwal, T. Krishna, L.-S. Peh, and N. K. Jha, "GARNET: A detailed on-chip network model inside a full-system simulator," in *Proc. IEEE Int. Symp. Perform. Anal. Syst. Softw.*, Apr. 2009, pp. 33–42.

[29] M. Moudi, M. Othman, K. Y. Lun, and A. R. A. Rahiman, "X-folded TM: An efficient topology for interconnection networks," *J. Netw. Comput. Appl.*, vol. 73, pp. 27–34, Sep. 2016.

[30] J. Kim, J. Balfour, and W. Dally, "Flattened butterfly topology for on-chip networks," in *Proc. 40th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO)*, Dec. 2007, pp. 1–5.

[31] M. Besta and T. Hoefler, "Slim fly: A cost effective low-diameter network topology," in *Proc. SC*, New Orleans, LA, USA, Nov. 2014, pp. 348–359.

[32] J. Kim, W. J. Dally, and D. Abts, "Flattened butterfly: A cost-efficient topology for high-radix networks," *ACM SIGARCH Comput. Archit. News*, vol. 35, no. 2, pp. 126–137, May 2007.

[33] *Virtual Channels and Deadlocks, Topics in Parallel Computing*. Accessed: Dec. 28, 2020. [Online]. Available: http://pages.cs.wisc.edu/~tvrdik/8/html/Section8.html

[34] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 63–74, Oct. 2008.

[35] Y. Miura, K. Shimozono, S. Watanabe, and K. Matoyama, "An adaptive routing of the 2-D torus network based on turn model," in *Proc. Int. Symp. Comput. Netw.*, Dec. 2013, pp. 587–591.

[36] E. John and J. Rubio, *Unique Chips and Systems*. Boca Raton, FL, USA: CRC Press, 2007.

[37] R. Wang, K. Lu, J. Chen, W. Zhang, J. Li, Y. Yuan, P. Lu, L. Huang, S. Li, and X. Fan, "Brief introduction of TianHe exascale prototype system," *Tsinghua Sci. Technol.*, vol. 26, no. 3, pp. 361–369, Jun. 2021, doi: 10.26599/TST.2020.9010009.

[38] *Fortinet Transceivers, GBic Module, FG-TRAN-SFP28-SR*. Accessed: Feb. 20, 2021. [Online]. Available: https://www.fortinet.com/

[39] H. Wang and L. Peh, "Power-driven design of router microarchitectures in on-chip networks," in *Proc. Int. Symp. Microarchitecture (MICRO)*, Dec. 2003, pp. 105–116.

[40] H. Cho, P. Kapur, and K. C. Saraswat, "Power comparison between high-speed electrical and optical interconnects for interchip communication," *J. Lightw. Technol.*, vol. 22, no. 9, pp. 2021–2033, Sep. 2004, doi: 10.1109/jlt.2004.833531.

[41] Y. Shen *et al.*, "Silicon photonics for extreme scale systems," *J. Lightw. Technol.*, vol. 37, no. 2, pp. 245–259, Jan. 2019, doi: 10.1109/JLT.2019.2897365.

[42] A. Gutierrez, R. G. Dreslinski, T. F. Wenisch, T. Mudge, A. Saidi, C. Emmons, and N. Paver, "Full-system analysis and characterization of interactive smartphone applications," in *Proc. IEEE Int. Symp. Workload Characterization (IISWC)*, Austin, TX, USA, Nov. 2011, pp. 81–90.

[43] *The Gem5 Simulator*. Accessed: Jan. 20, 2020. [Online]. Available: https://www.gem5.org/

[44] J. W. Dally and B. P. Towles, *Principles and Practices of Interconnection Networks*. San Mateo, CA, USA: Morgan Kaufmann, Mar. 2004.

[45] M. Abd-El-Barr and T. F. Al-Somani, "Topological properties of hierarchical interconnection networks: A review and comparison," *J. Electr. Comput. Eng.*, vol. 2011, pp. 1–12, Feb. 2011.

[46] S. P. Mohanty, B. N. B. Ray, S. N. Patro, and A. R. Tripathy, "Topological properties of a new fault tolerant interconnection network for parallel computer," in *Proc. Int. Conf. Inf. Technol.*, Dec. 2008, pp. 36–40.

[47] (1997). *MPI Forum, MPI-2: Extensions to the Message Passing Interface*. [Online]. Available: http://www.mpi-forum.org/docs/mpi-20-html/mpi2-report.html

[48] G. A. D. Araujo, D. Griebler, M. Danelutto, and L. G. Fernandes, "Efficient NAS parallel benchmark kernels with CUDA," in *Proc. 28th Euromicro Int. Conf. Parallel, Distrib. Network-Based Process. (PDP)*, Mar. 2020, pp. 9–16, doi: 10.1109/PDP50117.2020.00009.

[49] M. M. H. Rahman, Y. Inoguchi, Y. Sato, and S. Horiguchi, "TTN: A high performance hierarchical interconnection network for massively parallel computers," *IEICE Trans. Inf. Syst.*, vol. E92-D, no. 5, pp. 1062–1078, 2009.

[50] E. Agrell, M. Karlsson, A. R. Chraplyvy, D. J. Richardson, P. M. Krummrich, P. Winzer, K. Roberts, J. K. Fischer, S. J. Savory, and B. J. Eggleton, "Roadmap of optical communications," *J. Opt.*, vol. 18, no. 6, May 2016, Art. no. 063002.

[51] Y. Ajima, T. Inoue, S. Hiramoto, and T. Shimizu, "Tofu: Interconnect for the k computer," *Fujitsu Sci. Tech. J*, vol. 48, no. 3, pp. 280–285, Jul. 2012.

[52] M. Moudi and M. Othman, "A GreedyZero algorithm to minimise the conflicts in an optical multistage interconnection network," *J. Netw. Comput. Appl.*, vol. 41, pp. 312–318, May 2014.

FAIZ AL FAISAL received the B.Sc. degree in computer science and engineering from the Khulna University of Engineering and Technology (KUET), Khulna, Bangladesh, in 2009, and the M.Sc. and Ph.D. degrees in information science from the Japan Advanced Institute of Science and Technology (JAIST), Japan, in 2015 and 2018, respectively. He is currently working as an Assistant Professor with the Department of Computer Science and Engineering (CSE), Green University of Bangladesh (GUB), Dhaka, Bangladesh. Before joining GUB, he worked as an Assistant Professor with the Canadian University of Bangladesh (CUB) and North South University (NSU), Dhaka. Prior to his M.Sc. graduation, he worked as an IN-VAS Associate Engineer at Orascom Telecom Bangladesh Ltd., during 2011–2013. He was also provisioned as a Software Engineer at Leads Corporation Ltd., Dhaka, in 2010. His current research interests include parallel and distributed computer systems, interconnection networks, machine learning algorithms, and deep learning algorithms.

M. M. HAFIZUR RAHMAN received the B.Sc. degree in EEE from the KUET, Khulna, Bangladesh, in 1996, and the M.Sc. and Ph.D. degrees in information science from the JAIST, in 2003 and 2006, respectively. He is currently working as an Assistant Professor with the Department of CN, CCSIT, KFU, Saudi Arabia. Prior to joining the KFU, he was an Assistant Professor with Xiamen University, Malaysia; and IIUM, Malaysia; and an Associate Professor with the CSE, KUET. He was also a Visiting Researcher with the School of Information Science, JAIST. He was a JSPS Postdoctoral Research Fellow with the Graduate School of Information Science (GSIS), Tohoku University, Japan, in 2008 and 2009, and the Center for Information Science, JAIST, Japan, from 2010 to 2011. His current research interests include hierarchical interconnection networks, optical switching networks, and software defined networks.

YASUSHI INOGUCHI (Member, IEEE) received the B.E. degree from the Department of Mechanical Engineering, Tohoku University, in 1991, and the M.S. and Ph.D. degrees from the Japan Advanced Institute of Science and Technology (JAIST), in 1994 and 1997, respectively. From 1994 to 1997, he was a Research Fellow of the Japan Society for the Promotion of Science. From 2002 to 2006, he was a Researcher of the PRESTO Program, Japan Science and Technology Agency. From 2008 to 2009, he was a Courtesy Senior Research Scholar at the University of South Florida. He is currently a Professor with the Research Center for Advanced Computing Infrastructure, JAIST. His research interests include parallel computer architecture, interconnection networks, GRID/Cloud architecture, and high-performance computing on parallel machines. He is a member of IEICE and IPS of Japan.