International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 5, No.2 (March-2016)

http://dx.doi.org/10.12785/ijcds/050205

Group based Shortest Path Routing Algorithm for Hierarchical Cross Connected Recursive Networks (HCCR)

Omair Inam[1], Sharifa Al Khanjari[2] and Wim Vanderbauwhede[2]

[1] *Electrical Engineering Department, COMSATS Institute of Information Technology, Islamabad, Pakistan*
[2]*School of Computing Science, University of Glasgow, Glasgow, UK*

Received 20 July 2015, Revised 28 December 2015, Accepted 17 January 2016, Published 1 March 2016

Abstract: Interconnection networks play a significant role in efficient on-chip communication for multicore systems. This paper introduces a new interconnection topology called the Hierarchical Cross Connected Recursive network (HCCR) and a Group-based Shortest Path Routing algorithm (GSR) for the HCCR. Network properties of HCCR are compared with Spidergon, THIN, 2-D Mesh and Hypercube. It is shown that the proposed topology offers a high degree of regularity, scalability, and symmetry with a reduced number of links, small diameter and low node degree. A unique address encoding scheme is proposed for hierarchical graphical representation of HCCR networks, based on which the GSR was developed. . The proposed addressing scheme divides the HCCR network into logical groups of same as well as different sizes. Packets move towards receiver using local or global routing. Simulations are performed to find all the possible shortest paths with GSR in HCCR networks (up to 1024 nodes). All the shortest paths produced by GSR are verified against Dijkstra's algorithm. The GSR for k-level HCCR (L_k) with $N = 4^{(2+k)}$ nodes, requires $5(k-1)$ time in the worst case to determine the next node along the shortest path. Average distance and frequency of hop counts of HCCR networks are investigated using GSR. The results are compared with average distance of 2-D Mesh. Experimental results show that with a network size of 1024 nodes, there is only a 7.7% increase in the average distance of L_3 HCCR in comparison to 2-D Mesh. However L_k have fewer paths with high hop count in comparison to 2-D Mesh.

Keywords: Shortest path routing Algorihtm, Network-on-Chip,Heirarchical Interconnects, manycores

1. INTRODUCTION

Over the last two decades, advancements in deep submicron technology have made it possible to incorporate multimillion transistors into complex, power efficient, reliable and low cost embedded systems. Nowadays, electronic consumer products can be realised as integrated set of electronic devices capable of performing variety of functions. In order to exploit parallel computing, chip manufacturers include multiple CPU cores on a single chip i.e. multi-core or chip multiprocessor (CMP) architectures [1]. CMP has become a popular choice compared to complex, monolithic, power hungry unicore CPUs. For example Intel core 2 duos, Intel core i7, AMD Opteron, IBM Cell processor are popular CMPs. Tilera Crop has unveiled Tile Gx100 with 100-core general purpose processor.

The evolution of CMP with many cores puts enormous strain on the underlying on-chip communication architecture. A single shared bus is the simplest on-chip communication architecture. Traditional bus-based and crossbar-based communication architectures such as ARM's AMBA, AHB, AXI, ST Microelectronics' STBus, IBM's Core-connect, and the Sonics Silicon Backplane, may provide efficient, cost effective solution and well-defined communication protocol to add new cores into the system. But increasing number of cores on a centralized interconnect architectures can worsen arbitration delay, latency, reachable clock frequency and power consumption due to the sharing of aggregate bandwidth, resulting in limited scalability. Network-on-Chip (NoC) has emerged as a promising candidate for efficient on-chip communication with distributed scheduling and routing. NoC provides high-throughput, low-latency, congestion-aware scheduling, better scalability and reusability in multicore systems. NoC is mainly characterised by its topology, routing algorithm, control flow techniques and router microarchitecture. The two fundamental design challenges for any NoC in terms of overall system cost and performance are topology and routing algorithm. Topology defines the physical layout between the nodes and communication links inside the network. The role of routing algorithm is to route a packet towards its receiver, using minimal or non-minimal paths inside the network.

Email address: omair_inam@comsats.edu.pk, s.al-khanjari.1@research.gla.ac.uk, wim.vanderbauwhede@glasgow.ac.uk

The specific route chosen by an algorithm determines the hop-count, which has profound impact on the system's performance in terms of throughput and latency. Routing algorithms for NoCs are classified as minimal and non-minimal routing. The former always chooses the shortest paths and latter may use longer paths for communication.

This paper presents a new on-chip interconnection architecture i.e. Hierarchical Cross Connected Recursive network (HCCR) along with a novel group-based shortest path routing algorithm (GSR). HCCR is derived from the WK-recursive topology [2], with small node degree and reduced number of links. The proposed network is simple, hierarchical, symmetrical and scalable interconnection architecture which makes it suitable for communication in parallel/distributed networks. Our main contributions are: 1) giving a unified representation of HCCR with unique address encoding scheme and provide a solution to find shortest path in HCCR; 2) validating the results of proposed algorithm with Dijkstra's algorithm [3]; 3) demonstrating that HCCR can provide low cost communication architecture due to reduced number of links and small node degree; 4) studying the average distance and frequency of hop count in HCCR (up to 1024 nodes) and assess its performance for many core systems. To perform all above tasks, a SystemC based communication framework is developed to implement HCCR (up to 1024 cores) along with the proposed routing algorithm. Average distance and frequency of hop counts in HCCR and 2-D Mesh are evaluated by using GSR and XY algorithm respectively. Simulation results show that for a network size of 1024 nodes, HCCR has a smaller average distance as compared to Spidergon and THIN networks. There is only a 7.7% increase in the average distance of HCCR in comparison to 2-D Mesh for a network size of 1024 nodes. However HCCR has fewer very long paths in comparison to 2-D Mesh and scales better with the size of network. These properties of HCCR make it a good choice for efficient communication in many core systems, at low cost. In the worst case, for k-level HCCR (L_k) with $n = 4^{(2+k)}$ nodes, proposed algorithm requires $5(k-1)$ time for $k > 0$ to determine the next node along the shortest path.

The remainder of this paper is organized as follows: In the next section related work on the NoC topologies and shortest path routing algorithms for hierarchical networks is discussed. In section III, architecture of HCCR networks is explained. In Section IV, network properties of HCCR are defined. In Section V, hierarchical address encoding scheme is presented for shortest path routing. Section VI discusses the address mapping scheme for shortest path routing in HCCR. Section VII explains the working of GSR for k-level HCCR (L_k). In Section VIII, simulation results are illustrated. Section XI draws the conclusion and discusses future recommendation.

2. RELATED WORK

A. NoC Toplogies

There have been many architectural and theoretical studies on NoCs topologies. Direct topologies such as rings [4] are cost-effective, but deliver relatively poor performance, especially as the number of connected cores increases. Reference [5] proposed Tornado NoC, a scalable ring based architecture for many core systems. Proteo [6] introduced hierarchical interconnect topology with global bidirectional ring, connecting several sub networks. Reference [7] presented a hierarchical ring topology, focusing on reducing energy consumption of interconnect using dynamic clock throttling. On the other hand, among higher connectivity topologies, 2-D mesh [8] has been the most popular topology. However in case of larger networks, 2-D meshes are susceptible to large hop counts over a long distance. Reference [9] chose hybrid network consisting of hierarchical rings in the mesh, with aim to reduce congestion and reduce latency. Torus has been introduced to further reduce the 2-D mesh diameter by adding wrap around links [8]. A hierarchical torus has been studied to get better performance than torus in case of localized traffic distribution [10]. CMESH has been introduced [11] with an aim to reduce the diameter and number of routers in 2-D mesh. However CMESH requires long links and higher degree routers than 2D-Mesh. A WK-recursive hierarchical network has been presented [2], offering high degree of regularity, scalability, and symmetry, which makes it suitable for manufacture using VLSI technology. There are very little efforts made for the low degree network such as Octagon [12] and Spidergon [13] proposed by ST Microelectronics. In [24] Spidergon-Donut (SD) network is proposed. SD extends Spidergon to the second dimension with 1024 nodes. Triple-based Hierarchical Interconnection Network (THIN) [14] focused on decreasing node degree, reducing links and shortening diameter. THIN is preferable to construct interconnection network when the network size is not too large.

B. Shortest Path Routing Algorithms for hierarchical interconnection network

An extensive research has been conducted on the shortest path routing algorithm for hierarchical interconnection networks. Reference [15] presented an algorithm for shortest path routing on crossed cube network with time complexity $O(n^2)$. Optimal dynamic two terminal message routing algorithm has been presented for k-circulant $(k \geq 2)$ networks for the restricted shortest path in $O(\log n)$ time [16]. Shortest path routing is proposed for WK-recursive incomplete networks with $O(d.t)$ time preprocessing. Where $d > 1$ is the size of basic building block and $t \geq 1$ is the level of expansion [17]. This algorithm takes $O(t)$ time for each intermediate node to determine the next node along the shortest path. Reference [18] proposed shortest path routing algorithm to investigate the properties of Hyper de

Brujin network. Shortest path oblivious routing algorithm is proposed for n-level fully connected networks (FCCNs) with $N = 8^n$ nodes, at the cost of O(N) parallel offline steps and list of N stored at each node [19]. This algorithm requires O(n) time at each related node. Reference [20] presented a routing algorithm that finds n disjoint shortest paths in n-dimensional hypercube in $O(n^3 \log n)$. Shortest path routing algorithm is proposed for Multi-Mesh of Trees on n^4 processor network [21]. This algorithm requires 12 log n + 1 time in the worst case. Distributed deterministic routing algorithm (DDRA) [22] for THIN is proposed, but it is not the shortest path. The Min-DDRA [23] is proposed to further improve DDRA to obtain the shortest path. To avoid processing at each intermediate node, source routing is opted [24] using SPORT for the shortest path calculations in THIN network.

3. Topology and Construction of HCCR

In this section, following definitions are introduced to explain the construction of the proposed topology:

Definition 1. The HCCR network H is represented by strongly connected directed multigraph , $H = G(V, C)$, where vertex V represents the set of cores and C represents the set of communication channels. For k-level HCCR network (L_k), number of nodes (N) are defined as follows, where $k \geq 0$

$$N = 4^{(2+k)} \quad (1)$$

Definition 2. Fig.1 (a) shows the basic module consisting of four nodes, is used to construct the level 0 HCCR network (L_0). Any part of the HCCR network, in which four basic modules are connected as shown in Fig 1 (b) is referred as "local block" and denoted as L_0

Definition 3. As shown in Fig.1 (c), K Level HCCR (L_k) is constructed with four recursive L_{k-1} HCCR networks, where k > 0. e.g. Fig.2 (a) shows 2 Level HCCR(L_2), which is constructed using four recursive 1 Level HCCRs(L_1). There are 16 local blocks (L_0) and 256 nodes in L_2 HCCR.

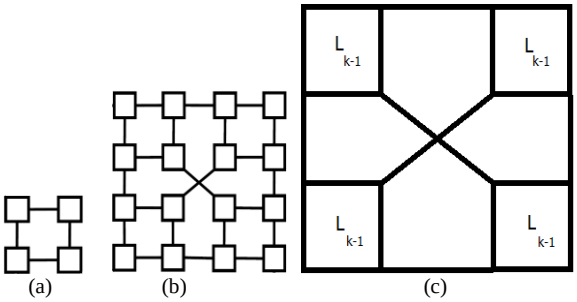

Figure 1. a) Basic Module of HCCR; (b) L_0 ; (c) L_k

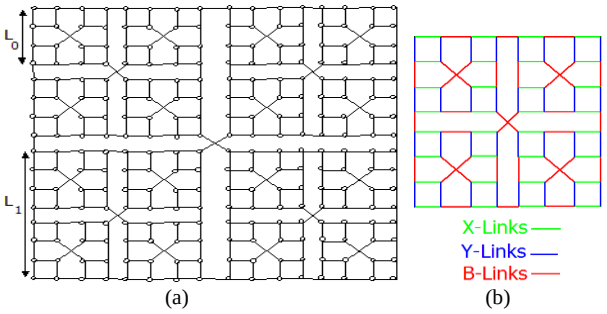

Figure 2. a) 2 Level HCCR (L_2); b) Link disciption

Definition 4. In L_k three types of links are used to connect any node with its neighbours and are labeled as 'X-links' (along x-axis), 'Y-links' (along y-axis) and 'B' as shown in the Fig. 2b. B is called "bridge link" which joins one basic module to another basic module in L_k HCCR

Definition 5. In HCCR , Three ports (portx, porty and portb) are defined at each node corresponding to appropriate link. These ports determine the link on which packet will be trasmitted.

4. Network Properties of HCCR

In this section, principal properties of HCCR are defined to characterize HCCR; node degree, number of links and diameter. Later in the section, properties of HCCR are compared with Spidergon, THIN, 2-D Mesh and Hypercube. Properties of HCCR networks are defined as follows.

Definition 6. Given a vertex $i \in V$ of a graph G, the number of channels connecting any node to its neighbors is called the degree and denoted by d_i. The maximum degree of all nodes in G is called network degree and is denoted by d. by definition.

$$d = max(d_i)$$

d_{HCCR} denote the network degree of L_k

$$d_{L_k} = 3 \quad (2)$$

Definition 7. B_k denotes the number of links in L_k HCCR and represented by (3). Where k is the level and N represents the number of nodes in L_K HCCR.

$$B_K = \frac{1}{2}\left(3 \times 4^{(2+K)} - 4\right) = \frac{1}{2}(3N - 4) \quad (3)$$

Definition 8. The maximum number of edges in graph G among the all shortest paths in a network is called the diameter. Let $D_{HCCR(K)}$ denotes the diameter in in L_K and is represented by (4).

$$D_{HCCR(K)} = (2^{k+1} + \sqrt{4^{(2+K)}} - 1)$$
$$= (2^{(log_4 N)-1} + \sqrt{N} - 1) \quad (4)$$

Definition 9. The average distance is the sum of all shortest paths between any two nodes in the network divided by the total number of shortest paths available in the network. Average distance of L_K HCCR is investigated using GSR and discussed in the section VIII.

A network's theoretical cost is usually determined by the number of links, number of routers, wire density, number of interfaces and VLSI layout area. Networks with lower degree are likely to have less communication complexity of the router, including wire congestion in the physical layout. Moreover with constant nodes degree, independent of the network size, the network is more modular and may operate at higher frequency as compared to high radix networks. For all switching strategies, a small network diameter improves network contention (in buffer and edges) and reduces propagation delay in point-point communication. Table 1 compares the network properties of the HCCR with Spidergon, THIN, 2-D mesh and Hypercube. Fig. 3, 4 and 5 compare the links and communication complexity of five different networks with different sizes, node degree and diameters. Fig. 3 illustrates that HCCR network not only out performs low degree networks such as Spidergon and THIN, in terms of diameter, but also outperforms 2-D mesh. It is due to the fact that HCCR has better scalability and modularity with small node degree. Fig. 3 and Fig. 4 show that despite having a low diameter over large network size, Hypercube becomes complex structure due to more number of links and variable node degree. A major drawback of Hypercube-based networks is that they lack expansibility. As their size grows, the number of ports gives an upper bound on their expansion. On the other hand HCCR have the fewest number of links when compared to 2-D Mesh and Hypercube (Fig. 4). A network with this property gains an advantage of easy implementation and low cost. As discussed earlier in the case of Hypercube, increasing network degree can reduce the diameter but may give rise to implementation complexity. We present a normalized diameter (diameter x degree), to study the tradeoff between node degree and diameter. Both parameters determine the overall system's cost and performance. Fig. 5 shows that the HCCR has the lowest normalize diameter and a show promising tradeoff between cost and performance for many core systems.

5. Hierarchical Address Encoding Scheme for HCCR

This section presents a novel hierarchical addressing scheme to accurately locate any node in L_K. The scheme gives a unified representation to HCCR and provides the reference point to find possible shortest paths in the network. The proposed addressing scheme divides the HCCR network into logical groups of same as well as different sizes. Packets are routed between logical groups by using local and global addresses of sender and receiver. The address mapping and GSR is discussed in section VI and VII. All the nodes in L_K HCCR are grouped and addressed according to rules defined as follows.

A. Local Address

According to the definition 3, L_k may consist of one or more L_0. All the nodes in L_0 have local addresses. Local address of any node is used for local communication between any sender-receiver pair inside L_0. Local address remains same and independent of size of network (Fig. 6). This gives a unified representation to all nodes

TABLE 1. COMPARISON OF TOPOLOGY PROPERTIES

Types of network	Network degree	Number of links	Diameter
Spidergon	3	$\frac{3}{2}(N)$	$\left\lceil\frac{N}{4}\right\rceil$
THIN	3	$\frac{3}{2}(N-1)$	$2^{\log_3 N}-1$
HCCR	3	$\frac{1}{2}(3N-4)$	$(2^{(log_4 N)-1}+\sqrt{N}-1)$
2-D mesh	4	$2(N-\sqrt{N})$	$2(\sqrt{N}-1)$
Hypercube	$\log_2^N$	$\frac{N\log_2^N}{2}$	$\log_2^N$

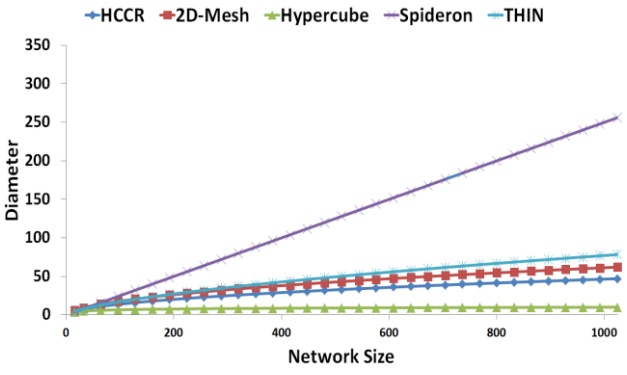

Figure 3. Diameter vs Network Size.

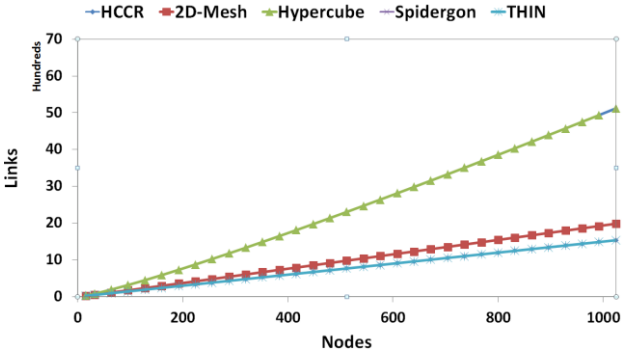

Figure 4. Links vs Network Size.

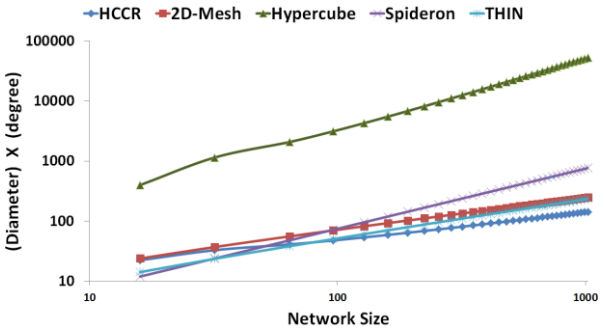

Figure 5. Normalize diameter vs Network Size.

with same local addresses in L_k. Local address of any node in L_0 is represented by 4 bits. Fig. 6 shows the local address of node with local ID = 01 and local group ID= 11. Following are the definitions for local ID and local group ID:

Definition 10. Fig. 6 shows, all nodes inside L_0 are arranged in four distinct groups called "local groups". Each local group consists of four nodes having unique local IDs. If G_L defines the set of 4 local groups in L_0 then

$$G_L = \{ G_{00}, G_{01}, G_{10}, G_{11} \} = \{ 00, 01, 10, 11 \} \quad (5)$$

Where G_{00}, G_{01}, G_{10} and G_{11} are local groups IDs for set of four nodes in L_0. Local group ID of any node is represented by 2 bits and independent of network size (Fig. 6).

Definition 11. If N_L defines the set of 4 unique local IDs in each local group then.

$$N_L = \{N_{00}, N_{01}, N_{10}, N_{11}\} = \{00, 01, 10, 11\}. \quad (6)$$

Where N_{00}, N_{01}, N_{10} and N_{11} are local IDs of each node. local ID of each node is represented by 2 bits (Fig. 6).

B. Local index

In order to identify the position of any node inside L_0 "local index" is defined as follows.

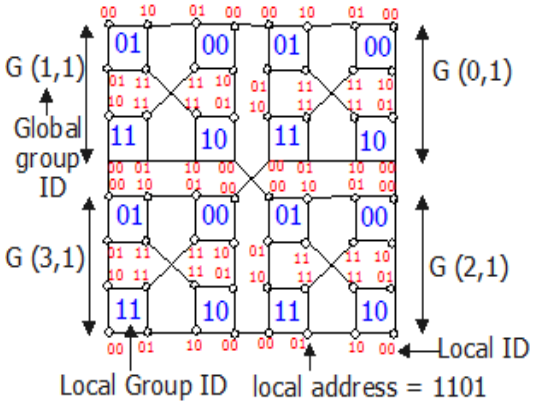

Figure 6. Address enconding in L_1 HCCR.

Definition 12. Let (n_x, n_y) denote the position vector of node n in L_0 where $0 \le n_x \le 3$ and $0 \le n_y \le 3$. n_x and n_y are referred as *local index* of node n in x and y dimension respectively. Local index of node remains same inside L_0 and independent of the position of L_0 in L_k HCCR.

C. Global Addresses

"Global address" are the logical addresses which are used for the communication between sender-receiver pair outside L_0, in L_K where $k > 0$. In Fig. 7 for global representation of any node, four symmetrical quadrants (Q1, Q2, Q3, and Q4) are introduced corresponding to four global groups. All four quadrants have only positive x and y axis. The size of global groups or quadrant depends on the level of communication (Lcom) between sender and receiver. Any node can be addressed by its "Global group ID" and represented as G (i, Lcom), where i denotes the quadrant number and $0 \le i < 3$, and Lcom denotes the level of communication between sender and receiver. Hence global group ID of any node depends on the position of L_0 in L_k. In Fig. 7 Sender (S1) belongs to Q3 of L_2 HCCR and receiver (R1) belongs to Q1 of L_2 HCCR. Therefore the Lcom between S1 and R1 is 2 i.e. (Lcom = 2). In this case global group IDs of S1 and R1 are G(3,2) and G(1,2) respectively. In Fig. 7, sender (S2) belongs to Q0 of L_1 HCCR and receiver (R2) belongs to Q3 of L_1 HCCR. Lcom between S2 and R2 is 1. Global group IDs of sender and receiver s are G(0,1) and G(3,1) respectively.

D. Global index

In order to identify the position of any node outside side L_0 "global index" is defined as follows.

Definition 13. Let (g_x, g_y) denote the position vector of node g in L_k, where $0 \le gx \le 2^{k+1} - 1$, $0 \le gy \le 2^{k+1} - 1$ and $k > 0$. g_x and g_y are referred as *global index* of node g in x and y dimension respectively. This definition shows that global index dependents on the size of HCCR network.

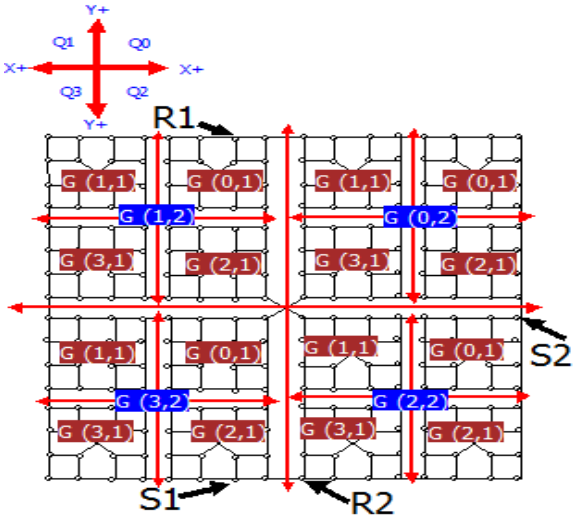

Figure 7. Global groups IDs in L_2 HCCR.

6. Address Mapping for Shortest Path Routing Algorithm

This section describes the unique address mapping scheme for L_K where $k > 0$, based on the address encoding scheme discussed in previous section. This scheme maps every sender with fixed number. These fixed numbers are used as the reference point to find the shortest path and remain fixed for any size of HCCR network. The generation of fixed numbers depends on the communication pattern that exists between sender and receiver with different global group IDs. Fig. 8 illustrates the two "basic communication patterns" in HCCR networks i.e. "up-down communication" and "diagonal communication". Fig. 8 shows that for communication between any sender and receiver with global group ID G(3,Lcom) and G(1,Lcom) in L_k, belongs to up-down communication and referred to as G(3,Lcom) → G(1,Lcom), whereas communication between sender and receiver with global group ID G(3,Lcom) and G(0,Lcom), belongs to diagonal communication and referred as G(3,Lcom) → G(0,Lcom). It can be seen in Fig. 8 that the topology is symmetrical along the horizontal, vertical as well as the diagonal axes. Therefore all the existing communication patterns in the network are the mirror of G(3,Lcom) → G(1,Lcom) or G(3,Lcom) → G(0,Lcom). In proposed algorithm G(3,Lcom) → G(1,Lcom) and G(3,Lcom) → G(0,Lcom) are used as "reference communication patterns" for all other patterns, to find the shortest path in L_k HCCR for $k > 0$.

Definition 14. If P_{updown} defines the set of up-down communication patterns in L_k for $k > 0$then

$$P_{updown} = \{ P_{3\text{-}1}, P_{1\text{-}3}, P_{2\text{-}0}, P_{0\text{-}2}, P_{2\text{-}3}, P_{3\text{-}2}, P_{1\text{-}0}, P_{0\text{-}1} \}. \quad (7)$$

Where,

$P_{3\text{-}1}$ = G(3,Lcom) → G(1,Lcom), $P_{1\text{-}2}$ = G(1, Lcom) → G(3, Lcom), $P_{0\text{-}2}$ = G(0, Lcom) → G(2, Lcom) , P_{20} = G(2,Lcom) → G(0,Lcom) P_{23} = G(2,Lcom) → G(3,Lcom) , P_{32} = G(3,Lcom) → G(2,Lcom) , P_{10} = G(1,Lcom) → G(0,Lcom) and P_{01} = G(0,Lcom) → G(1,Lcom).

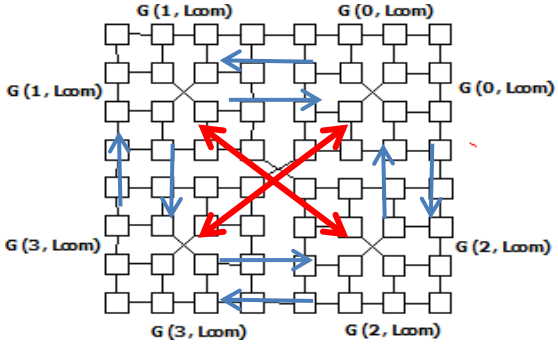

Figure 8. Up-Down and Diagonal Communication patterns in L_1

Definition 15. $P_{diagonal}$ defines the set of diagonal communication patterns in L_k for $k > 0$

$$P_{diagonal} = \{ P_{3\text{-}0}, P_{0\text{-}3}, P_{1\text{-}2}, P_{2\text{-}1} \}. \quad (8)$$

Where,

$P_{3\text{-}0}$ = G(3,k) → G(0,Lcom), $P_{0\text{-}3}$ = G(0,Lcom) → G(3,Lcom), $P_{1\text{-}2}$ = G(1,Lcom) → G(2,Lcom) , P_{21} = G(2,Lcom) → G(1,Lcom).

Following subsections explain the address mapping techniques and the generation of fixed numbers in case of basic communication patterns i.e. up-down and diagonal communication.

A. Address Mapping for Up-Down Communcuation.

In this subsection an address mapping is presented for P_{updown} (7). The purpose of this technique is to map the sender with a fixed number. Routing algorithm use this fixed number as a reference point to decide about the next node for the shortest path. Fig. 9 shows that in case of reference communication pattern i.e. G(3,Lcom) → G(1,Lcom), sender (S) either belong to Upper Half (UH) or Lower Half (LH) according to the following conditions.

Condition 1: If (Sgx ≥ Sgy) Sender belongs to upper half.

Condition 2: If (Sgx < Sgy) Sender belongs to lower half.

Where Sgx and Sgy are global indices.

Finding the next node along the shortest path does not follow the same rule in both conditions. To elaborate it further (Fig. 9) consider a case G(3,1) → G(1,1) where sender with global group id = 3 belongs to the UH of L_1 and receiver has global group id = 1. In this case sender can only choose between top right and top left corner of L_0 to send packet toward receiver using shortest path. For G(3,1) → G(0,1), if sender belongs to LH. In this case sender can only select among top right, top left and bottom right corners of L_0 for shortest path

communication. Due to the fact that two different set of choices are required in both cases, different fixed numbers are generated for each case. For up-down communication from UH, MapForUH( ) will generate fix number and map to it to sender. In Fig. 10, all the senders in UH are mapped with fixed numbers. Each fixed number is representing y-intercept of one of diagonal in receiver's quadrant (Fig. 10). Every diagonal is containing reference nodes and dividing the receiver's quadrant into two sections, such that reference node on the same diagonal should satisfy the following condition:

Condition 3: $D_{topleft} \leq D_{topright}$

Where $D_{topright}$ represents the shortest distance between sender and any reference node on diagonal via top right corner of L_0 and D_{left} denotes the shortest distance of sender from the same reference node via top left corner of L_0(Fig. 10)

According to condition 3, MapForUH( ) assigns the value of y-intercept (fixed numbers) to any sender in UH communication. MapForUH( ) calculates the fixed number by evaluating local ID, local group ID and position of L_0 in L_k HCCR. Since HCCR is a hierarchical structure so there are three possible fixed number arrangements for any sender in L_0 shown in Fig. 11. MapForUH() produces an appropriate fixed number according to the position of sender in network and size of L_k. After generation of fixed number for up-down communication in UH, UpperHalf( ) compares the fixed numbers using condition 4 to select an appropriate corner between right and top left, for the shortest path communication condition 4 is given as follows.

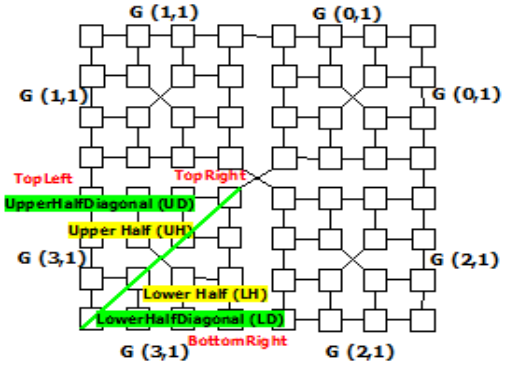

Figure 9. Upper Half (UH) , Lower Half(LH), Upper Half Diagonal (UD) and Lower Half Diagonal(LD) in L_1

Condition 4: if $(R_{gy} - R_{gx} \leq fixednumber)$
Select top left corner of L_0
else
Select top right corner of L_0
Where R_{gx} and R_{gy} are global indices of receiver.

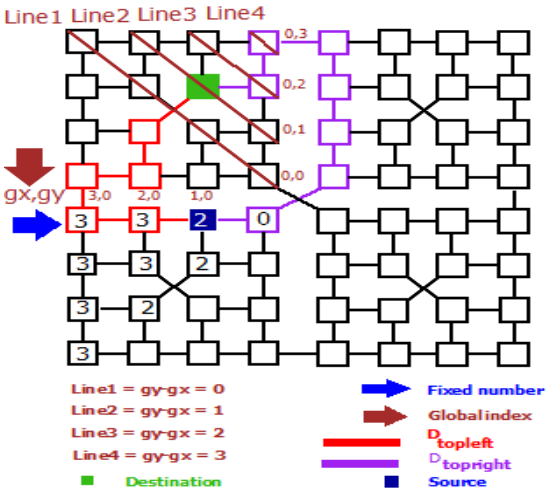

Figure 10. Updown communcation in UpperHalf(UH)

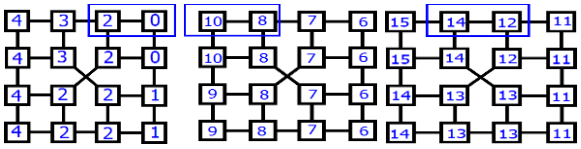

Figure 11. Three possible fixed number represntations in UH

Once the valid corner is selected, getdirection( ) selects an appropriate port (portx, porty or portb) to send the packet towards selected corner in L_0. Pseudo-code of all the defined functions is given in Fig. 14, 15, 16 and 17 for the reference. getdirection( ) and CallLocalBlockCom () are working in similar manners. Therefore pseudo code for one of them is given. Both functions are responsible to send a packet inside L_0. For only local routing CallLocalBlockCom () is used. Whereas for global routing getdirection() is responsible for the transition of packet inside L_0 and will be explained further.

As discussed above, for up-down communication from LH the generation of fixed number is not same as UH. To explain it further we go back to the previous example where G(3,1) → G(1,1) and sender (S) belongs to LH and receiver has global group ID = 1. In this case sender needs to decide among top right, top left and bottom right corner nodes of L_0. As shown in Fig. 12 MapForLH_BR_TL( ) generates two fixed numbers to decide between bottom right and top left corners of L_0 for shortest path communication. To select between bottom right and top right corners of L_0, MapForLH_BR_TR( ) produces another set of 2 fixed numbers (Fig. 13). Both said functions use same approach to find respective fixed numbers. Set of fixed numbers generated by MapForLH_BR_TL( ) represent the global indices of two reference nodes in receiver's quadrant and must satisfy one the following conditions:

Condition 5: sen_grp_id $= 0$ and $D_{TRx1} = D_{TLx2}$

Condition 6: sen_grp_id $= 1$and $D_{TRx1} = D_{TLx2} - 1$

Condition 7: sen__grp_id = 2 and $D_{TRx1} = D_{TLx2} + 1$

Condition 8: sen_grp_id = 3 and $D_{TRx1} = D_{TLx2}$

Where sen_grp_id denotes the local group ID of sender in L_0. D_{TRx1} and D_{TLx2} are the shortest distances between sender and reference nodes form top right and top left corners of L_0 respectively. Subscripts $_{x1}$ and $_{x2}$ represent the global indices of respective reference nodes as shown in Fig. 12. The global index of each reference node represents one fixed number. Both fixed numbers are providing reference to sender for the selection of one corner node among top left and bottom right as shown in Fig. 12. These two reference points divide the receiver quadrant into two sections along y-axis. Inrange ( ) checks the availability of receiver in both sections and selects the nearest possible corner in L_0 for shortest path communication, as illustrated in Fig. 12. Inrange ( ) returns true in case of bottom right corner of L_0, whereas it returns false for top left of L_0 (Fig. 15)

MapForLH_BR_TR( ) needs to satisfy the following conditions for the generation of fixed numbers:

Condition 9: sen_grp_id = 0 and $D_{BRy1} = D_{TRy2} + 1$

Condition 10: sen_grp_id = 1and $D_{BRy1} = D_{TRy2}$

Condition 11: sen_grp_id = 2 and $D_{BRy1} = D_{TRy2} + 1$

Condition 12: sen_grp_id = 3 and $D_{BRy1} = D_{TRy2}$

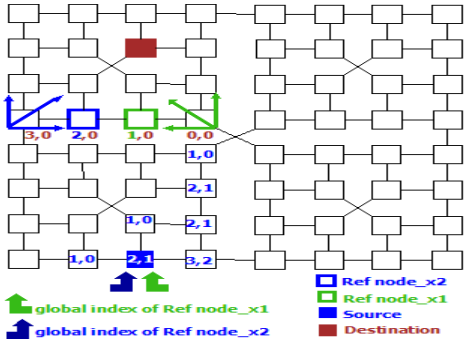

Figure 12. LH communication via bottom right and top left corner of L_0 .

Where D_{BRy1} and D_{TRy2} are the shortest distances between sender and reference nodes from bottom right and top right corners of L_0 respectively. Subscripts $_{y1}$ and $_{y2}$ denote the global index belonging to respective reference nodes in the receiver quadrants. The global index of the each reference nodes represents one fixed number. These two reference nodes divide the receiver quadrant into two sections along x-axis, with respect to the sender. Inrange ( ) checks the availability of receiver in both sections and finds the nearest possible corner between top right and bottom right corners in L_0 . Inrange ( ) returns true if top right corner of L_0 is selected or returns false in case of bottom right corner.

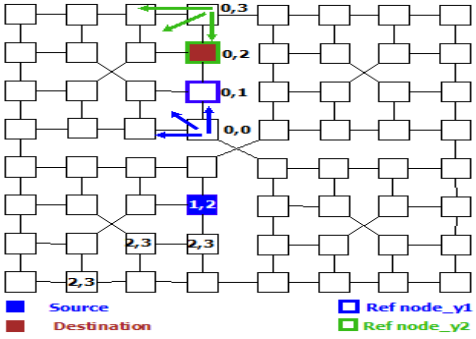

Figure 13. LH communication via bottomright and top right corner of L_0.

As shown in Fig. 15 and 16, for up-down communication, UpperHalf( ) and LowerHalf( ) evaluates the positions of sender-receiver s with respect to three corners in L_0. Result from Inrange() is compared to select the appropriate corner of L_0. Once a valid corner is selected, getdirection( ) selects appropriate port (portx, porty or portb) required to send the packet toward the selected corner in L_0.

B. Address Mapping for Diognal Communcuation.

In this subsection an address mapping is presented for all the communication patterns defined in (8). As shown in Fig. 9, for G(3,k) → G(0,k), any sender either belongs to Upper Half Diagonal(UD) or lower Half diagonal (LD) according the condition 1 and 2.

Fig. 9 For G(3,K) → G(0,K) if sender is in UD, it selects between top left or top right corner of L_0 for the shortest path communication. Whereas for the same case if sender is in LD then it will select between top right and bottom right corner of L_0 for the shortest path communication. Since the same choice is required in both cases therefore generation of fixed number is using same rule for both cases.

MapForUD( ) and MapForLD( ) are used for the generation of fixed numbers in diagonal communication from UD and LD respectively. Each function generates 2 fixed numbers to select between respective choice of corners in L_0. In case of MapForUD( ) available options are top left and top right corners. Whereas in case of MapForLD( ) available options are top right and bottom right corners. Both functions use the same approach to find respective fixed numbers. All the fixed numbers represent the global index of two reference nodes in receiver quadrant and must satisfy one the following conditions:

Condition 13: sen_l_grp_id = 0 and $D_{TRx1} = D_{BRx2} + 1$

Condition 14: sen_l_grp_id = 1and $D_{TRx1} = D_{BRx2}$

Condition 15: sen_l_grp_id = 2 and $D_{TRx1} = D_{BRx2}$

Condition 16: sen_l_grp_id = 3 and $D_{TRx1} = D_{TRx2} + 1$

Condition 17: sen_l_grp_id = 0 and $D_{TRy1} = D_{TLy2} + 1$

Condition 18: sen_l_grp_id = 1and $D_{TRy1} = D_{TLy2} + 1$

Condition 19: sen_l_grp_id = 2 and $D_{TRy1} = D_{TLy2}$

Condition 20: sen_l_grp_id = 3 and $D_{TRy1} = D_{TLy2}$

Conditions 13-16 are defined for MapForLD( ) where D_{BRx2} is shortest distance between sender and reference node form bottom right corner of L_0 . Subscript $_{x2}$ represents the global index of reference node in receiver's quadrant.

Conditions 17-20 are given for MapForUD( ), where D_{TRy1} and D_{TLy2} representing the shortest distances between sender and reference node from top right corner and top left corners of L_0 respectively.Subscripts $_{y1}$ and $_{y2}$ are representing the global index of each reference node. UpperHalfDiagonal( ) and LowerHalfDiagonal( ) evaluates the position of receiver with respect to the three corners in L_0 . Inrange() selects an appropriate corner of L_0 for the shortest path communication. The address of selected corner is used by local routing procedure called getdirection( ). This function selects an appropriate port (portx, porty or portb) required to move the packet toward the selected corner in L_0.

7. Group based Shortest Path Routing Algorithm (GSR) for HCCR

Based on the hierarchical address scheme and mapping techniques discussed in previous sections, a Group Based Shortest path Routing algorithm (GSR) is presented in this section. As shown in the Fig. 5, the algorithm first determines the Lcom between sender-receiver using their global and local addresses. In case of local communication (i.e. sender and receiver pair belong to L_0), local routing is performed by callLocalBlockcom( ). This function selects the output port of sender's node through which the packet needs to move forward for the shortest path in L_0 . During local routing in L_0, all intermediate nodes between sender and receiver call the stated function to find their valid ports for the shortest path communication. In case of global routing i.e communication outside (L_0), the algorithm finds among one of the two basic communication patterns described in previous section. After determining the pattern, index conversions are required in the sender and receiver addresses with respect to the reference communication pattern where $G(3,k) \rightarrow G(1,k)$ is the reference for all up-down communication patterns defined in (11) and $G(3,k) \rightarrow G(0,k)$ for all diagonal communication patterns defined in (12). Find-Routing-Par_up_dwn_com() and Find-Routing-Param_diagonal_comm() performaddress conversions on the sender and receiver addresses. Global routing works in two phases. In first phase corner node is selected for an exit from L_0 . In the second phase, the dimension of selected corner node is provided togetdirection( ) which works similar to the callLocalBlock( ). In the case of global routing, every intermediate node sends packet to appropriate corner of is L_0 until the packet finally reaches its receiver.

Time complexity: For local routing proposed algorithm only uses local address of sender and receiver. According to definition (10-15) local address of any node is independent of size of a HCCR network. Therefore for local communication, callLocalBlockcom( ) and getaddress( ) involve single call with time complexity O(1). In case of global routing time complexity of the algorithm depends on the communication pattern and size of the network.

```
Direction ShortestPathRouting_HCCR()
{
 if (levelOfCommuniation==0)direction = callLocalBlockCom()
 else
 {
  if (up_down_comm)
  {  para = FindRoutingParam_up_down_comm(arg)
     if (para = upperhalf) direction = UpperHalf(arg)
     else direction = LowerHalf(arg)
  }
  else
  { para = FindRoutingParam_diagonal_comm(arg)
    if (low_half_diag) direction = UpperHalfDiagonal(arg)
    else if (upper_half_diag) direction = LowerHalfDiagonal(arg)
    else
    {
    corner = topright
    direction = getdirection(cornernode)
    }
  }
 }
return direction
}// end ShortestPathRouting_HCCR
```

Figure 14. GSR algorithm.

```
Direction UpperHalf(arg)
{FixNumber = MapForUH()
if(Rgy-Rgx < FixNumber) corner = topleft
else corner = topright
direction = getdirection(cornernode)
return direction
}//end UpperHalf()

Direction LowerHalf(arg)
{FixNumberTR_TL = MapForLH_TR_TL()
FixNumberBR_TR = MapForLH_BR_TR()
rangex = Inrange(FixNumberTR_TL)
rangey = Inrange(FixNumberBR_TR)
 if (rangex == true && rangey == false)
 { corner = bottomright
   direction = getdirection(cornernode)
 }
else direction = UpperHalf(arg)
return direction
}//end LowerHalf()
```

Figure 15. Up-Down communication method for global routing

```
Direction UpperHalfDiagonal(arg)
{FixNumber = MapForUD()
rangey = Inrange(FixNumber)
 if (rangey ==  fasle) corner = topleft
else corner = topright
direction = getdirection(cornernode)
return direction
}//end UpperHalfDiagonal()

Direction LowerHalfDiagonal(arg)
{FixNumber = MapForUD()
rangey = Inrange(FixNumber)
 if (rangey ==  fasle) corner = bottomright
else corner = topright
direction = getdirection(cornernode)
return direction
}//end LowerHalfDiagonal()
```

Figure 16. Diagonal communcation method for global routing

If k denotes the level of HCCR network then FindRoutingParam_up_down_comm( ) and FindRoutingParam_diagonal_comm( ) each require 3(k-1) time to find global addresses and index conversions. LowerHalf ( ) takes 2(k-1) time whereas UpperHalfDiaognal( ) and LowerHalfDiagonal( ) each takes k-1 time. UpperHalf( ) involves single call to getadress( ) and requires O(1) time. As shown in Fig. 17, up-down communication requires 5(k-1) time in worst case to determine the next node along the shortest path.

```
Direction callLocalBlockCom(arg)
{// All parameters are local
 if (srcgroup  == dstgroup ) run XY algorithm()
 else
{   if(srcid  == 0) { //'if node if is0
        if (dx == 0 || abs_dx == 1) directions = Y
        else direction = X
      } //'end src_id 0
    if(srcid == 1)
      {
       if(dx == 0 || dy == 0 || (abs_dx == 1 && abs_dy == 1) ||
        (dstid  == srcid ) && (((dstgroup  == 0
        || dstgroup  == 3) && abs_dy == 2) || ((dstgroup  == 2
        || dstgroup  == 1) && abs_dx == 2)))
        direction = B
        else if( (dx < 0 && dy < 0) || (dx > 0 && dy > 0)) direction = Y
        else direction = X
      }//'end src_id 1
    if(srcid == 2)
      {
        if(dx == 0 || dy == 0 || (abs_dx == 1 && abs_dy == 1) ||
        (dstid == srcid) && (((dstgroup  == 0 ||
         dstgroup  == 3) && abs_dx == 2) || ((dstgroup  == 2
         || dstgroup  == 1) && abs_dy == 2)))
        direction = B
        else if ( (dx < 0 && dy < 0) || (dx > 0 && dy > 0)) direction = X
        else direction = Y
      } //'end srcid 2
    if(src.local_id  == 3)
      {
        if((abs_dx == 1 && abs_dy == 1 && dstid != srcid)
        ||(dstid == 0 && abs_dx != abs_dy) ||
        (dstid == srcid && (abs_dx & abs_dy) == 0))
        {
         if((srcgroup == 1 && dst.group  == 3) ||
            ( srcgroup == 3 && dstgroup  == 1) ||
            (srcgroup == 0 && dstgroup  == 2) ||
            (srcgroup == 2 && dstgroup  == 0) )
          directions = X
        else direction = Y
        }
       else  direction = B
    } end srcid 3
 }// end first else
return directions;
}// end callLocalBlockCom(arg)
```

Figure 17. Local block communcation method for local routing

8. EVALUATION

A. Simulation platform

In order to implement and evaluate the proposed routing algorithm, a SystemC based communication framework has been developed. In this platform all the nodes in L_k can send packets to any other nodes in the network. The purpose of this platform is to validate the working of the proposed routing algorithm and to analyze the average distance and frequency of hop counts in L_k. In our simulations only one node at a time can send packet to any other node in the network because the scope of this work is not to find the performance of the network with all load conditions. Number of hops traversed by all packets along their paths, are stored in the communication history. This later is used for analysis.

B. Simulation Results

Verification of shortest path: For test and verification of GSR, L_3 (with 1024 nodes) is considered. The reason to choose large network size is to ensure that GSR is doing all the index conversions accurately with respect to the size of network. The simulation is done by grouping together 16 nodes into one block i.e. L_0 . There are a total of 63 blocks in L_3. All the nodes inside one sender block are sending packets to all nodes inside the network. Sum of all the shortest paths are plotted in Fig. 18 as a result of block level communication. To verify all the shortest paths and hop counts in L_3 , the same simulations are done with Dijkstra's algorithm and a hop count of 1,048,576 possible shortest paths in L_3 are compared. It is observed that proposed algorithm produces exactly same hop counts for all possible shortest paths, as in case of the Dijkstra's algorithm. Fig. 18 shows the hop counts of all possible shortest paths in L_3 for GSR and Dijkstra's algorithm.

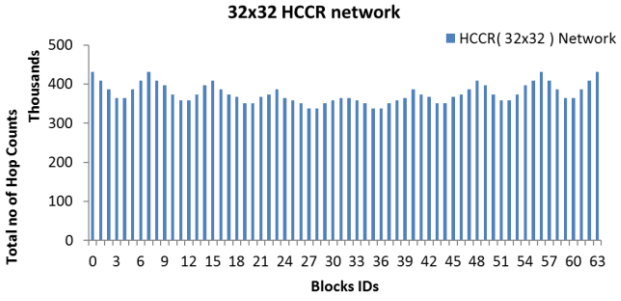

Figure 18. Total number of Hops counts generated by each block GSR in L_3

Average distance & Frequency of hop counts: As discussed in section 4, the network properties of L_k scales well with size of the network. In this section two other network properties of L_k are presented i.e. average distance and frequency of hop counts. Small average distance improves average latency and throughput of overall system. Frequency of hop counts represents distribution of all possible shortest paths in overall network. Group based shortest path algorithm is used to

evaluate the average distance and frequency of hop counts in L_k with different network size upto 1024 node. Average distance is plotted in comparison with 2-D Mesh, Spidergon, THIN networks. It is can be seen from Fig 19 that HCCR networks have small average distance in comparison to Spidergon and THIN for network size greater than 60 nodes, whereas there is only 7.7% increase in the average distance of HCCR (L_3) in comparison to 2-D Mesh for a network size of 1024 nodes. Frequency hops counts in HCCR and 2-D Mesh are evaluated using shortest path routing algorithms. For 2-D Mesh XY algorithm is used. Results are plotted with different network size. It shows that HCCR has always fewer longer paths available and it get better as the network size scales. These results show that HCCR can be a good candidate for providing efficient and low cost communication with locality based traffic in many core systems.

9. Conclusion

We have presented a new interconnection topology called the Hierarchical Cross Connected Recursive network (HCCR) and a shortest path routing algorithm for the HCCR. Network properties of HCCR networks are compared with Spidergon, THIN, 2-D Mesh and hypercube topologies. It is shown that the properties of HCCR make it suitable even for large network sizes due totter expansibility, small node degree and low diameter. We proposed a Group based Shortest Path Routing algorithm (GSR) to guarantee efficient communication in HCCR. The GSR for k-level HCCR (L_k) with $N = 4^{(2+k)}$ nodes, requires $5(k-1)$ time in the worst case to determine the next node along the shortest path. Average distance for L_k is measured by simulating GSR on L_k with the help of SystemC based communication framework. All the shortest paths in L_3 network are verified by Dijkstra's algorithm. Our simulation results show that HCCR networks have a

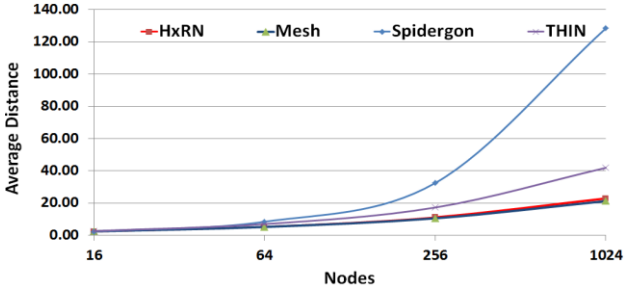

Figure 19. Comparison of Average distance

have a small average distance in comparison to Spidergon and THIN, whereas there is only a 7.7% increase in the average distance of HCCR (L_3) in comparison to 2-D Mesh for a network size of 1024 nodes. However HCCR have fewer longer paths as compared to a 2-D Mesh. Future research will focus on simulating HCCR networks with the shortest path routing algorithm under different network load conditions and comparing the performance with other interconnection networks.

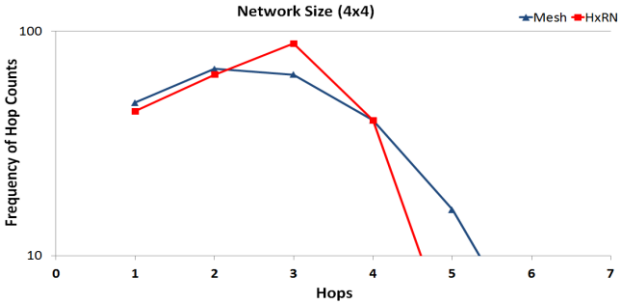

Figure 20. Frequency hop counts in L_0

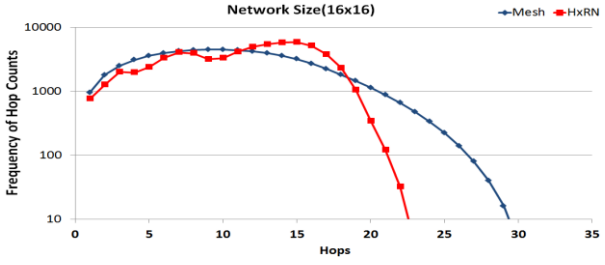

Figure 21. Frequency hop counts in L_1

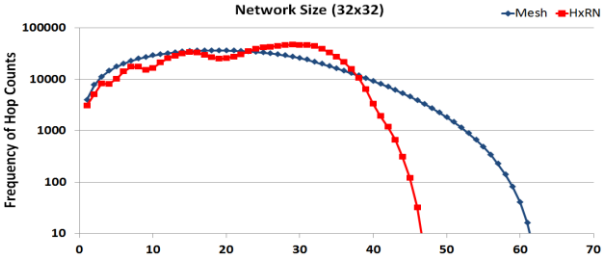

Figure 22. Frequency hop counts in L_2

References

[1] Jeyapaul, Reiley, Fei Hong, Abhishek Rhisheekesan, Ashish Shrivastava, and Kyoungwoo Lee. "UnSync-CMP: Multicore CMP Architecture for Energy-Efficient Soft-Error Reliability." *Parallel and Distributed Systems, IEEE Transactions on* 25, no. 1 (2014): 254-263.

[2] Farahabady, M. Hoseiny, Navid Imani, and Hamid Sarbazi-Azad. "Some topological and combinatorial properties of WK-recursive mesh and WK-pyramid interconnection networks." *Journal of Systems Architecture* 54, no. 10 (2008): 967-976.

[3] Dijkstra, Edsger W. "A note on two problems in connexion with graphs."*Numerische mathematik* 1, no. 1 (1959): 269-271.

[4] Bononi, Luciano, and Nicola Concer. "Simulation and analysis of network on chip architectures: ring, spidergon and 2D mesh." In *Proceedings of the conference on Design, automation and test in Europe: Designers' forum*, pp. 154-159. European Design and Automation Association, 2006.

[5] Lee, Junghee, Chrysostomos Nicopoulos, Hyung Gyu Lee, and Jongman Kim. "TornadoNoC: A lightweight and scalable on-chip network architecture for the many-core era." *ACM Transactions on Architecture and Code Optimization (TACO)* 10, no. 4 (2013): 56.

[6] Siguenza-Tortosa, D., and Jari Nurmi. "Proteo: a new approach to network-on-chip." In *IASTED International Conference on Communication Systems and Networks (CSN'02)*. 2002.

[7] Bourduas, S., B. Kuo, Z. Zilic, and N. Manjikian. "Modeling and evaluation of an energy-efficient hierarchical ring interconnect for System-on-Chip multiprocessors." In *Circuits and Systems, 2006 IEEE North-East Workshop on*, pp. 201-204. IEEE, 2006.

[8] Mirza-Aghatabar, Mohammad, Somayyeh Koohi, Shaahin Hessabi, and Massoud Pedram. "An empirical investigation of mesh and torus NoC topologies under different routing algorithms and traffic models." In *Digital System Design Architectures, Methods and Tools, 2007. DSD 2007. 10th Euromicro Conference on*, pp. 19-26. IEEE, 2007.

[9] Bourduas, Stephan, and Zeljko Zilic. "Modeling and evaluation of ring-based interconnects for Network-on-Chip." *Journal of Systems Architecture* 57, no. 1 (2011): 39-60.

[10] Loucif, Samia. "Performance evaluation of hierarchical-torus NoC." InAdvanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on, pp. 837-842. IEEE, 2013.

[11] Balfour, James, and William J. Dally. "Design tradeoffs for tiled CMP on-chip networks." In *Proceedings of the 20th annual international conference on Supercomputing*, pp. 187-198. ACM, 2006.

[12] Karim, Faraydon, Anh Nguyen, Sujit Dey, and Ramesh Rao. "On-chip communication architecture for OC-768 network processors." In *Proceedings of the 38th annual Design Automation Conference*, pp. 678-683. ACM, 2001.

[13] Bononi, Luciano, and Nicola Concer. "Simulation and analysis of network on chip architectures: ring, spidergon and 2D mesh." In *Proceedings of the conference on Design, automation and test in Europe: Designers' forum*, pp. 154-159. European Design and Automation Association, 2006.

[14] Xue, Licheng, Feng Shi, and Weixing Ji. "3D floorplanning of low-power and area-efficient Network-on-Chip architecture." *Microprocessors and Microsystems* 35, no. 5 (2011): 484-495.

[15] Yu, Xin, and Tao-shen Li. "On shortest path routing algorithm in crossed cube-connected ring networks." In *Cyber-Enabled Distributed Computing and Knowledge Discovery, 2009. CyberC'09. International Conference on*, pp. 348-354. IEEE, 2009.

[16] Dobravec, Tomaž, Janez Žerovnik, and Borut Robič. "An optimal message routing algorithm for circulant networks." *Journal of Systems Architecture* 52, no. 5 (2006): 298-306.

[17] Su, Ming-Yang, Gen-Huey Chen, and Dyi-Rong Duh. "A shortest path routing algorithm for incomplete WK-recursive networks." *Parallel and Distributed Systems, IEEE Transactions on* 8, no. 4 (1997): 367-379.

[18] Nguyen, Ngoc Chi, Thanh Vu Dinh, and Tuan Dang Anh. "Improving shortest path routing in hyper-de Bruijn networks." In *Strategic Technology, 2007. IFOST 2007. International Forum on*, pp. 288-292. IEEE, 2007.

[19] Yang, Xiaofan, Graham M. Megson, and David J. Evans. "An oblivious shortest-path routing algorithm for fully connected cubic networks." *Journal of Parallel and Distributed Computing* 66, no. 10 (2006): 1294-1303.

[20] Sau, Ignasi, and Janez Zerovnik. "An optimal permutation routing algorithm for full-duplex hexagonal mesh networks." *Preprint series, University of Ljubljana, Institute of Mathematics, Physics and Mechanics* 44 (2006): 1017.

[21] Jha, Sudhanshu Kumar, and Prasanta K. Jana. "Shortest Path Routing on Multi-Mesh of Trees." In *Proceedings of the World Congress on Engineering*, vol. 2. 2011.

[22] Qiao, Baojun, Feng Shi, and Weixing Ji. "A New Hierarchical Interconnection network for multi-core processor." In *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pp. 246-250. IEEE, 2007.

[23] Zuo, W. A. N. G., and S. H. I. Feng. "A Shortest Path Routing Algorithm in Triplet-Based Network." *Transactions of Beijing Institute of Technology* 5 (2009): 011.

[24] Zhang, Yang, and Feng Shi. "Design and Evaluation of Low-Latency and Shortest-Path Routing Algorithm for Triplet-Based Hierarchical Interconnection Network." *JOURNAL OF TESTING AND EVALUATION* 41, no. 4 (2013): 541-550.

Omair Inam is a PhD student in Computer Engineering, at COMSATS Institute of Information Technology. He obtained Master Degree in Electronic Engineering in 2008 from University of Sheffield, UK. He completed Bachelor Degree in Computer Engineering from Islamic International Engineering College Pakistan in 2002. His research area include Network-on-Chip, Computer Architecture and System Level Modeling

Sharifa Al Khanjari is a PhD student in Embedded, Networked and Distributed Systems (ENDS) at the School of Computing Science of the University of Glasgow. She received a Master and Bachelor Degree in Computer Science from Sultan Qaboos University, Oman in 2007 and 2010, respectively. Her research focus on the architecture of many

Dr Wim Vanderbauwhede is Lecturer in Embedded, Networked and Distributed Systems at the School of Computing Science of the University of Glasgow. His research focuses on high-level programming, compilation and architectures for heterogeneous manycore systems and FPGAs, with a special interest in power-efficient computing. He is author of the book "High-Performance Computing Using FPGAs". He received a PhD in Electrotechnical Engineering with Specialisation in Physics from the University of Gent, Belgium in 1996.