Network-on-Chip (NoC) Topologies and Performance: A Review

Jie Chen and Cheng Li
Electrical and Computer Engineering
Faculty of Engineering and Applied Science
Memorial University of Newfoundland
{*Jie.chen, licheng*}*@mun.ca*

Paul Gillard
Department of Computer Science
Memorial University of Newfoundland
paul@mun.ca

***Abstract*— With the development of integration technology, System-on-Chip (SoC), composed of heterogeneous cores on a single chip, has entered the billion-transistor era. As the microprocessor industry moves from single-core to multi-core, and eventually to many-core architectures, containing tens to hundreds of similar cores arranged on a single multiprocessor chip requires efficient communication among processors. A high-performance, flexible, scalable, and design-friendly interconnection architecture is highly desired for modern SoC and microprocessor designs. How to provide efficient communication poses a challenge to both academia and industry. Before the advent of Network-on-Chip (NoC), interconnection architectures were usually based on dedicated wires or shared buses. However, they cannot meet the ever-increasing demand from the on-chip systems due to the lack of scalability. NoC has been proposed as a highly structured and scalable solution to address the communication problems in on-chip systems. NoC has several advantages over dedicated wiring and buses, e.g., high-bandwidth, low-latency, low-power consumption and scalability. Messages are transported back and forth via the interconnection networks. Thus, the interconnections among multiple cores on a chip have a significant impact on communication and performance of the chip design in terms of end-to-end delay, throughput, and packets loss ratio. Therefore, it is worthwhile studying the different characteristics of different topologies. In this paper, we review the most popular topologies and also some recent topologies for interconnection networks. We study their performance and summarize their strengths and limitations.**

***Index Terms*— Network-on-Chip, Topology, Performance, Scalability.**

I. Introduction

With the development of integration technology, System-on-Chip (SoC), composed of heterogeneous cores on a single chip, has entered billion-transistor era. As the microprocessor industry is moving from single-core to multi-core and eventually to many-core architectures, containing tens to hundreds of identical cores arranged as chip multiprocessors, which also require efficient communications among processors. Both SoC and microprocessor call for a high-performance, flexible, scalable, and design-friendly interconnection [1]. How to provide efficient communication poses a challenge to researchers.

Before the advent of network-on-chip, interconnection architectures are usually based on dedicated wires or shared buses. Dedicated wires provide point-to-point connection between every pair of nodes, effective for small systems of a few cores. But as the number of cores increases, the number of wires in the point-to-point architecture grows quadratically, making it unable to scale. Compared to dedicated wires, a shared bus which is a set of wires shared by multiple cores, is more scalable and reusable. However, due to the inherent disadvantage of buses, only one communication transaction is allowed at a time, blocking communication for all other cores. The disadvantages of shared bus architectures include long data delay, high energy consumption, increasing complexity in decoding/arbitration, low bandwidth [2]. It would be daunting inefficient if hundreds of nodes are connected by shared buses. Thus, the usage of shared buses is limited to a few dozens of IP cores. To deal with the problems in shared buses, a hierarchical architecture, which segments bus into shorter ones, is introduced. Hierarchical bus architectures may relax some of constraints faced by dedicated wires and shared buses, since different buses may account for different bandwidth needs, protocols and also increase communication parallelism. Nonetheless, scalability remains a problem for hierarchical bus architectures. In order to meet the communication requirements, accelerate time-to-market and cut down the communication energy consumption of large scale SoCs, there is a great need to find a new design alternative to the conventional point-to-point and bus based computation architectures.

NoC has been proposed as a highly structured and scalable solution to address communication problems in SoC. On-chip interconnection network has several advantages over dedicated wiring and buses, i.e., high-bandwidth, low-latency, low-power consumption and scalability. NoC architectures can guarantee communication pipelining with a pre-specified clock rate regardless of the network size, which is infeasible for bus-based architectures. For SoCs, cores can be DSPs, embedded memory blocks, or CPUs, or video processors, etc. Figure 1 gives a typical Nvidia Tegra 600-series SoC [3], which consists 14 components, including CPU, GPU, image processor, video processor, UART, DDR, *etc*. A single chip can be partitioned into functional tiles and interconnection network [4]. Since its inception, NoC has drawn great

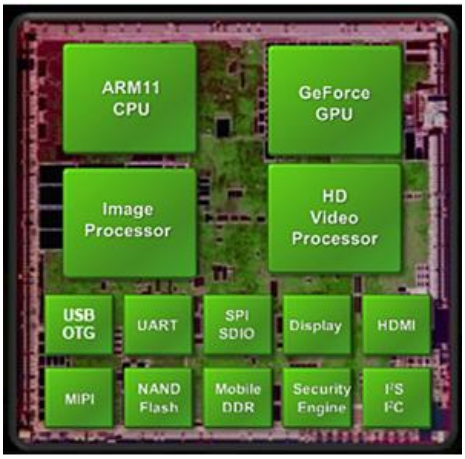

Fig. 1. Nvidia Tegra 600-series SoCs

attention from researchers all over the world. But to fully explore the benefits of NoC, numerous challenges and open problems are to be addressed. Open problems can be classified into four main categories, including application modeling and optimization, NoC communication architecture analysis and optimization, NoC Communication Architecture Evaluation, and NoC Design Validation and Synthesis. Due to the limit of space, we are not going to expand our scope to such a broad range.

II. Review of Some Classic Topologies

In this section, we review the most popular topologies, including ring, star, mesh, tree, fat tree, butterfly, and torus, etc. Later we summarize the strengths and limitations for some of them.

In a ring architecture, all nodes are connected in a ring fashion, as shown in Figure 2. Every node has two neighbors regardless the size of the ring. Its small degree is preferable, but it's diameter increases linearly with the number of nodes. Its strengths include: (1)cable faults are easily located, which makes troubleshooting easier. (2) Moderately easy to install compared with other architectures. Its limitations include: (1)Expansion to the network can cause network disruption. (2)Even a single break in the cable can disrupt the entire network.

In a star architecture, assume there are N nodes, $N-1$ nodes connect to a center node, as shown in Figure 3. Only the center node has a degree of $N-1$. Other nodes have degree of 1. Diameter of a star architecture is 2, regardless its size. Its strengths include: (1) A small diameter means a small average hop distance, which is a favorable characteristic. (2) Simplicity to operate. Each node is isolated free of impact from failed nodes. Its limitations include: (1) Failure of the central node fails the entire network. (2) Central node is the bottleneck.

In a mesh, nodes are connected as a grid, as shown in Figure 4. Expansion is easy for meshes. Little effort is needed when adding more nodes to the existing architecture. Nodes have different degrees according to their locations within the mesh. Corner nodes have degree of 2. Edge nodes have degree of 3. Inner nodes have degree of 4. Its strengths include: (1) Multiple paths between a pair of nodes, tolerant to link failure. (2) Easy to expand. Its limitations include: (1) Diameter can be very large. (2) Irregularity, less bandwidth for nodes at corners and edges.

In a binary tree, the top node is the root and the bottom nodes are the leaves, as shown in Figure 5. Every node except the root node has two offspring nodes. A node's children are those nodes that appear immediately beneath the node itself. A node's parent is the node immediately above it. A tree's root is the single node that contains no parent. Its strengths include: (1) Supported by many network vendors and even hardware vendors. (2)All the nodes have access to the larger and their immediate networks, best for branched out networks. Its limitations include: (1)Bottleneck on the root node. (2) When the tree is big, it is difficult to configure and can get complicated after a certain point.

In a fat binary tree, only leaves are intellectual properties (IP), as shown in Figure 6. Interior nodes are switches. When moving towards the root node, there are more links between a parent node and a child node. The number of inter-node links increases by order of 2.

In a butterfly architecture, as shown in Figure 7. Basic butterfly networks have two main disadvantages. Firstly, it lacks of path diversity. There is only one path from a source node to a destination node. Secondly, long wires are inevitable. Long wires must transverse half of the diameter of the network.

A torus architecture is obtained by adding direct connections to two end nodes in the same row or column in a mesh architecture. A 16-node torus is shown in Figure 8. Compared with mesh, its diameter is reduced. A regular torus has long wrap-around links. By folding a torus, long wires can be avoided at the cost of doubling the wire length.

III. Review of Some Recent Topologies

Hypercube is considered a very useful topology. But the network size is restricted due to its degree limitation. To overcome this major disadvantage, one possible solution is to reduce the node degree. Some variations have been proposed including folded hypercube, crossed cube, dual cube, hierarchical cube, cube-connected cycles and metacube with the goal of minimizing the network diameter or node degree while keeping the diameter small. Table II gives

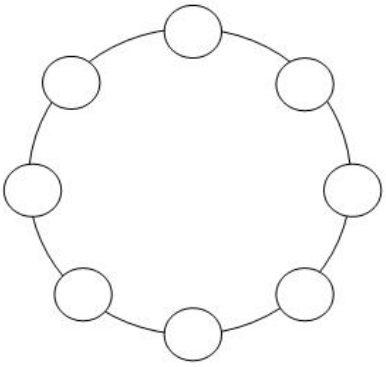

Fig. 2. Ring

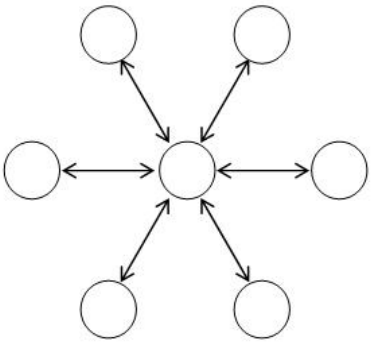

Fig. 3. Star

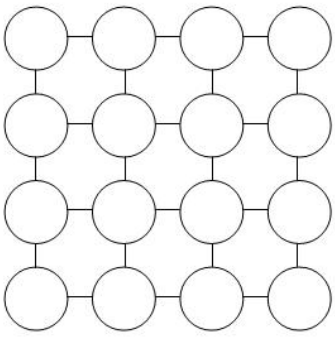

Fig. 4. Mesh

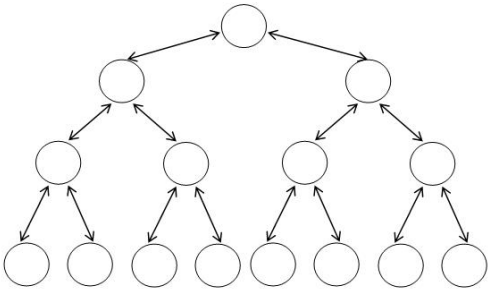

Fig. 5. Binary Tree

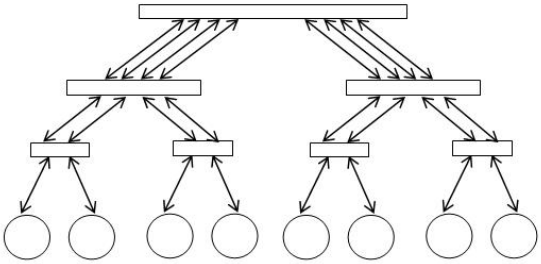

Fig. 6. Fat Binary Tree

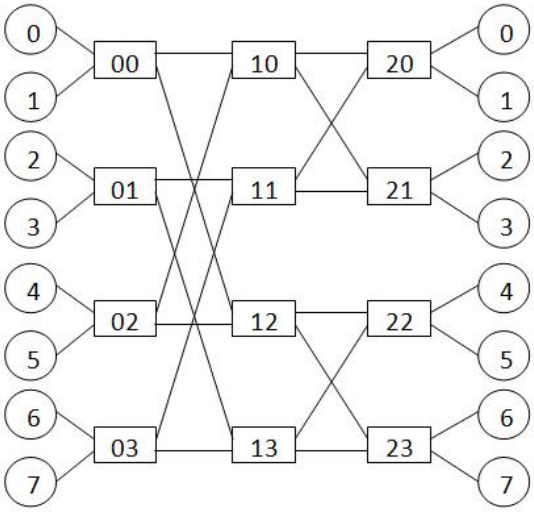

Fig. 7. Butterfly

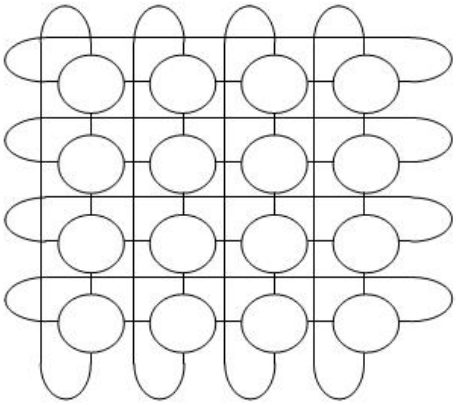

Fig. 8. Torus

several mutations of hypercube and their improvements compared to Hypercube.

Folded hypercube [5] is a variation of hypercube by connecting each node to the unique node farthest from it. Its diameter is reduced to about half of hypercube at the cost of more links. Crossed cube [6] is constructed by repositioning some edges in hypercube to reduce the diameter about half of hypercube without increasing the link complexity.Crossed cube profitably emulate a hypercube. Dual cube [7] is a hierarchical hypercube structure, which has two classes. Each class consists of clusters, and each cluster has $2m$ nodes.Dual cube is self explanatory. It has two classes, class 0 and class 1. The binary address of each node in an m-dual cube is $1+2m$ bit long. The address format of class 0 is left most bit is class ID, the middle m bits represent cluster ID, and right most m bits show its node ID. Reduced-hypercube [8] is obtained by reducing edges from an n-dimensional hypercube to reduce node degree. Hierarchical cubic network (HCN) [9] a (n, n) HCN has n cluster and each cluster has n-cube.Cube-connected cycles [10] is a virtual node Hypercube. Each virtual node in the Hypercube is a circle instead of a single node. Each node has three ports: F,B, and E, which stand for forward, backward, and external. It can emulate a Benes permutation network. It can lend itself to Fast-Fourier-Transform(FFT), sorting algorithms, odd-even merge, matrix multiplication, etc.

Metacube [11] is a two-level hypercube structure, a symmetric network with small node degree and short diameter. Metacube is a multi-class topology with two parameters, $MC(k, m)$. It is an extended version of dual cube in term of structure. A $MC(k, m)$ has 2^k classes; each class is consist of $2^{(2^k-1)m}$ clusters; a cluster has 2^m nodes. Total number of nodes is $2^{(2^k m)+k}$. In a node address, the left most k bits is its class ID, if we divide the other bits in group of m bits, then the $(c+1)th$ m bits from the right represents the node ID, and the rest $(2^k-1)m$ bits represent its cluster ID. We can see the position of its node ID in a node address varies with its class ID. For Metacube, there is a link between two nodes if and only if their addresses differ in only one bit in class ID or in node ID. Hypercube can be considered as a special case of MC. When k is 0, a MC becomes a Hypercube. From Table I, we can see the growth of number of nodes in Metacube family.

TABLE I
TOTAL NUMBER OF NODES VS NODE DEGREE

Links/node	3	4	5	6	7	8
MC(0,m)(Hypercube)	8	16	32	64	128	256
MC(1,m)(Dual-cube)	32	128	512	2048	8192	32,768
MC(2,m)(Quad-cube)	64	1024	16,384	2^{18}	2^{22}	2^{26}
MC(3,m)(Oct-cube)	-	2048	2^{19}	2^{27}	2^{35}	2^{43}
MC(4,m)(Hex-cube)	-	-	2^{20}	2^{36}	2^{52}	2^{68}

TABLE II
VARIATIONS OF HYPERCUBE

Topology	Reduce Diameter	Reduce Degree
Folded Hypercube	√	
Crossed Hypercube	√	
Dual-cube	√	√
Reduced Hypercube		√
Hierarchical Cubic Network	√	√
Cube-connected Cycles		√
Metacube	√	√

IV. IMPACT OF TOPOLOGIES ON PERFORMANCE

A. Topology Parameters

In a network, the topology is the arrangement of nodes and channels [12]. It determines the interconnection of nodes and can usually be modeled as a graph. A topology has parameters such as degree, diameter, link complexity and bisection width, etc. These parameters together characterize a topology and distinguish one from the others.

1) Node Degree: The number of links connected to a node. A network is called regular if all nodes have the same degree; otherwise, it is irregular. Node degree describes the I/O complexity of a node. Node degree can be constant or varied with the size of the network. For instance, ring has fixed node degree 2; the node degree of N-node hypercube is $log_2 N$. Smaller degree and fixed node degree are preferable topological characteristics. A smaller node degree requires less hardware cost on links. A fixed node degree reduces the effort to add new nodes to the existing network.In most cases, there is a constraint on node degree, which is the number of direct neighbors of a node. Limitation of node rise from two considerations: one is hardware limitation,

i.e., the number of ports a node can support, the other is communication protocol limitation, e.g., BlueTooth networks can support up to node degree of eight. Finally, performance considerations, such as space complexity and network scalability, may limit the number of nodes with which each node may communicate directly.

2) Diameter: The maximum shortest path between all pairs of nodes. If there is no direct connection between two nodes, the message has to travel through some intermediate nodes which will introduce multiple hop delay. Since the message delay is proportional to the number of hops, the length of the maximum shortest path becomes an important metric. In minimum routing, diameter determines the worst case distance. One example is broadcasting. The node farthest from the sender determines the worst case distance for broadcasting. While in non-minimum routing, the length of path can be longer than the diameter, depends on the status of the network and the specific routing algorithms. Even for non-minimum routing, small diameter can also help to provide low and predictable latency, predict routing paths, traffic flow and make troubleshooting easier.

3) Link Complexity: The total number of links in the topology. As the network scales, the link complexity increases. Adding more links to a certain network can reduce its diameter and provide better communication among nodes. But links are expensive. Higher link complexity may induce higher hardware complexity and area overhead. Among all topologies, fully connected network has the highest link complexity.

4) Bisection Width: The number of links needed to be removed to divide the topology into two networks with the approximate equal size. A large bisection width is preferable, because it provides more paths between two sub-networks, and thus improves overall performance. Large bisection width provides greater bisection bandwidth. Equation (1) gives the calculation of bisection bandwidth. In a star topology, it is impossible to cut a star into two equal size sub-network; the only way is to cut off one node, except the central one. Among the topologies we discussed in this paper, only ring, star and tree have fixed bisection width, regardless of the size of the network.

$$bisection\ bandwidth = bisection\ width \times channel\ bandwidth \quad (1)$$

B. Performance Metrics

1) Average End-to-End Delay: End-to-end delay is the time elapse for a packet to traverse the network, from source to destination. The way to calculate average delay is to take the mean of the end-to-end delay of each successfully sent packet. Note that lost packets are not considered in average end-to-end delay. Or this value would be infinite large. The average end-to-end delay reflects how fast the network can deliver packets to their destinations. The smaller this value is, the more efficient the network is.

2) Loss Rate: Loss rate is the ratio of packets lost to the total number of packets generated. The way to calculate the loss rate is the sum of dropped packets divided by the total number of packets generated by the source. Low loss rate is preferred due to Quality of Service (QoS). For different applications, there are the maximum acceptable loss rates.

3) Average Throughput: Throughput is the data rate in b/s successfully delivered to their destinations. The way to calculate average throughput is to average the number of packets successfully received by their destinations per second during the simulation time then convert to b/s. The channel which is heaviest loaded determines the maximum load of the topology, measured in b/s.

4) Average Queue Occupancy: Average queue length is the mean of queue length measured in terms of packets. It indicates the utilization of the buffers. The longer queue means a higher buffer utilization and bigger queuing delay. The queue length is sampled every time slot to get the utilization ratio.

V. Conclusion

For a network, topology is the arrangement of nodes and channels. Topology has a great impact on the performance of a network. From the view of high performance, low diameter and high bisection width are preferred. Apparently, fully connected topology satisfies these two requirements. But technically, considering the cost and effort of implementation, low degree, short wires, small area and regular structures are of great interest. As the network grows, fully-connected architecture fails to scale. In this paper, we review some classic topologies and list their strengths and limitations. To overcome the drawbacks of the existing topologies, researchers have proposed some new topologies. Although these recent topologies have not been really applied to practice, they bring opportunities for future Network-on-Chip.

References

[1] J. D. Owens, W. J. Dally, R. Ho, D. N. Jayasimha, S. W. Keckler, and L.-S. Peh, "Research challenges for on-chip interconnection networks," *Micro, IEEE*, vol. 27, no. 5, pp. 96 –108, September, 2007.

[2] C. Grecu, P. Pande, A. Ivanov, and R. Saleh, "Timing analysis of network on chip architectures for mp-soc platforms," *Microelectronics Journal*, vol. 36, no. 9, pp. 833–845, 2005.

[3] http://www.xbitlabs.com/news/mobile/display/20080603141353_Nvidia_Unleashes_Tegra_System_on_Chip_for_Handheld_Devices.html.

[4] W. J.Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Annual Design Automation Conference*, Las Vegas, Nevada, United States, June, 2001, pp. 684–689.

[5] A. El-amawy and S. Latifi, "Properties and performance of folded hypercubes," *IEEE Transaction on Parallel Distributed System*, vol. 2, no. 1, pp. 31–42, 1991.

[6] K. Efe, "The crossed cube architecture for parallel computation," *IEEE Transaction on Parallel Distributed System*, vol. 3, no. 5, pp. 513–524, 2004.
[7] Y. Li, S. Peng, and W. Chu, "Efficient collective communications in dual-cube," *The Journal of Supercomputing*, vol. 28, no. 1, pp. 71–90, 2004.
[8] S. G. Ziavras, "Rh: A versatile family of reduced hypercube interconnection networks," *IEEE Transaction on Parallel Distributed System*, vol. 5, no. 11, pp. 1210–1220, 1994.
[9] K. Ghose and K. R. Desai, "Hierarchical cubic networks," *IEEE Transaction on Parallel Distributed System*, vol. 6, no. 4, pp. 427–435, 1995.
[10] F. P.Preparata and J. Vuillemin, "The cube-connected cycles: A versatile network for parallel computation," *Communication of ACM*, vol. 24, pp. 300–309, 1981.
[11] Y. Li, S. Peng, and W. Chu, "Metacube - a versatile family of interconnection networks for extremely large-scale supercomputers," *The Journal of Supercomputing*, vol. 53, no. 2, pp. 329 – 351, 2010.
[12] W. J. Dally and B. Towles, *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.