A Case for Wireless 3D NoCs for CMPs

Hiroki Matsutani[1], Paul Bogdan[2], Radu Marculescu[2], Yasuhiro Take[1],
Daisuke Sasaki[1], Hao Zhang[1], Michihiro Koibuchi[3], Tadahiro Kuroda[1], and Hideharu Amano[1]

[1]Keio University
3-14-1 Hiyoshi, Kohoku-ku, Yokohama
cube@am.ics.keio.ac.jp

[2]Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh
{pbogdan,radum}@ece.cmu.edu

[3]National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo
koibuchi@nii.ac.jp

***Abstract*—Inductive-coupling is yet another 3D integration technique that can be used to stack more than three known-good-dies in a SiP without wire connections. We present a topology-agnostic 3D CMP architecture using inductive-coupling that offers great flexibility in customizing the number of processor chips, SRAM chips, and DRAM chips in a SiP after chips have been fabricated. In this paper, first, we propose a routing protocol that exchanges the network information between all chips in a given SiP to establish efficient deadlock-free routing paths. Second, we propose its optimization technique that analyzes the application traffic patterns and selects different spanning tree roots so as to minimize the average hop counts and improve the application performance.**

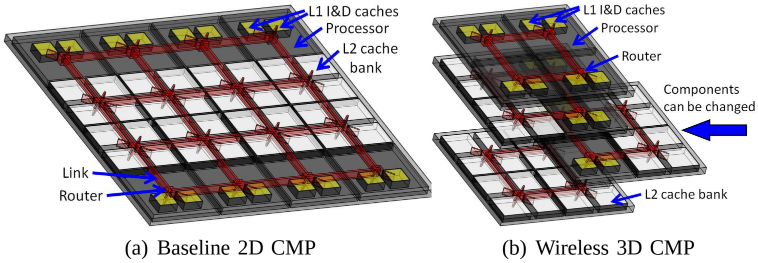
I. Introduction

Due to the increase in the design costs of custom System-on-Chips (SoCs) in recent process technologies, System-in-Packages (SiPs) or 3D ICs that can be used to select and stack necessary known-good-dies in response to given application requirements have become one of promising design choices. Various interconnection techniques have been developed to connect multiple chips in 3D IC packages: wire-bonding, micro-bump [1][2], wireless interconnects (e.g., capacitive- and inductive-coupling) [3][4][5][6] between stacked dies, and through-silicon via (TSV) [3][7] between stacked wafers. Many recent studies on 3D IC architectures have focused on micro-bumps and TSVs that offer the largest interconnect density. However, we consider inductive-coupling that can connect more than three known-good-dies without wire connections to be yet another 3D integration technique, because it provides a large degree of flexibility in building target 3D ICs, such as enabling chips in the package to be added, removed, and swapped after the chips have been fabricated, similar to what is done with building blocks.

We propose a novel wireless 3D Chip Multi-Processor (CMP) architecture as a baseline, in which the numbers of processor chips and cache chips in the same package can be customized for a given application set. That is, if the application set requires more cache capacity or bandwidth, we can add more cache chips to satisfy such demands. If the target application set has more thread-level parallelism, we can add more processor chips to improve performance. Traditional wired Network-on-Chips (NoCs) are used as intra-chip networks in the proposed wireless 3D CMPs, while inter-chip communication is based on wireless inductive-coupling. Since a wireless 3D CMP is a collection of various chips provided by different vendors (e.g., memory, processor, and GPU vendors), we cannot expect to know any pre-determined network topology for intra-chip communications; this makes it difficult to establish routing paths that are free from deadlocks in such ad-hoc 3D CMPs.

We first propose a routing protocol that exchanges the network information between all chips in a given 3D CMP to establish efficient deadlock-free routing paths. We employ a spanning-tree based routing algorithm, in which the average hop count increases depending on the location of the roots of the spanning trees embedded in the target network. To minimize the hop counts, we then propose a technique of optimizing networks, in which the traffic patterns of applications are analyzed and a different spanning tree root is selected for each message class to minimize the average hop count and improve the performance of applications.

The rest of this paper is organized as follows. Section II presents the proposed wireless 3D CMP architecture. Section III proposes a plug-and-play routing protocol for wireless 3D CMPs. Section IV describes the results we obtained from evaluations and Section V concludes the paper.



(a) Baseline 2D CMP
(b) Wireless 3D CMP

Fig. 1. 2D and 3D CMP architectures.

II. Wireless 3D Chip Multi-Processors (CMPs)

A. Baseline 2D and 3D CMP Architectures

Figure 1(a) provides an example of a CMP. The chip is divided into sixteen tiles, and each tile has a single processor (gray box) or four shared L2 cache banks (white boxes), in addition to a single on-chip router (red box). The main memory modules and their controllers are connected to four edges of the chip (not shown in the figure). These tiles are interconnected via their routers. As all L2 cache banks are shared by all processors, a Static Non-Uniform Cache Architecture (SNUCA) with a cache coherence protocol was assumed in this work.

A natural extention of the baseline 2D CMP is the 3D CMP, in which processors, SRAM, and DRAM chips are stacked in a single package, and various organizations have investigated [1][8]. 3D integration enables us to integrate multiple chips fabricated with different process technologies (e.g., logic and DRAMs) into a single package to reduce wire lengths, mitigate the pin count problem, and improve performance. To connect them, 3D NoC architecture [9] is a promising solution as a scalable communication infrastructure.

Most studies on 3D CMPs and 3D NoCs to date have assumed micro-bumps or TSVs (i.e., wired solution) for their inter-chip communication; none of these except [10][11] have focused on the wireless approach, which offers a higher degree of flexibility to add, remove, and swap chips after the chips have been fabricated. Note [10] introduces a real implementation of inductive-coupling based wireless 3D NoC and [11] proposes some communication schemes for wireless approaches (e.g., data compression), although these works assume fixed regular topologies, such as ring and $4 \times 4 \times 2$ mesh.

B. Wireless 3D CMP Architecture

Figure 1(b) outlines the proposed architecture for wireless 3D CMP, where the baseline 2D CMP is divided into four chips, each of which has four tiles, i.e., four processor tiles or four cache tiles. Wired links are used for the horizontal network that connects the four tiles on each chip, while wireless inductive-coupling links are used for the inter-chip network that vertically connects four chips. The memories and their controllers are connected to the bottom chip (not shown in this figure). We have assumed this simple 3D SNUCA to be the baseline for the sake of simplicity.

Three types of chips are illustrated in this figure. The bottom (first) chip that consists of sixteen L2 cache banks is a cache chip. The second and third ones that consist of two processors and eight L2 cache banks are hybrid chips. The top (fourth) one that consists of four processors is a processor chip. The performance of applications is typically limited by either computational power or memory bandwidth; thus applications can be classified into those that are computation-bound and memory-bound. Depending on the target application set, wireless 3D CMP using inductive-coupling enables us to customize the numbers and types of chips stacked in a package after the chips have been fabricated. For example, more processor chips can be added for computation-bound applications, while more cache chips will be useful for memory-bound applications. This flexibility is attractive, because designing a new mask pattern for each set of applications has been too costly with recent process technologies.

C. Network Requirements for Wireless 3D CMP

A wireless 3D CMP is a collection of various hardware components, such as processors, SRAMs, and DRAMs. They are based on different process technologies and fabricated by different vendors. As long as chips satisfy minimum requirements (e.g., performance, chip size, and vertical link pitches), their detailed horizontal architecture (e.g., processors, memories, and horizontal NoC parameters) should be left up to each vendor. A key challenge with wireless 3D CMPs is to stack such post-fabricated hardware components, such as processors and memory chips, provided by different vendors so that these components can cooperate with each other with minimum design rules.

The following minimum design rules are required. To connect two arbitrary chips vertically, 1) all chips must have inductors at pre-determined locations. 2) The inductors are connected to input and output ports of on-chip routers to form vertical links, and 3) all the hardware components are topologically reachable to at least an on-chip router that has a vertical link. Note that inductors can be implemented with a common CMOS process and their footprint is small, so the cost could be compensated by the plug-and-play flexibility. In addition, the on-chip routers support the plug-and-play routing protocol proposed in the next section.

If the above-mentioned rules are satisfied, we do not have to expect any pre-determined network topology for each chip. For example, one chip may have a mesh-based NoC while another may have a ring-based NoC. In an extreme case, it is possible to only have vertical links (i.e., no intra-chip links), if modules on the same chip rarely communicate with one another. Their communication in this case must go through another upper or lower chip via vertical links. Such routing paths cannot be expected to be known at the design time since a chip vendor cannot know the topological structures of upper and lower chips. Thus, conventional static routing strategies for NoCs are not sufficient for wireless 3D CMPs.

III. Plug-and-Play Routing with Spanning Trees Optimization

A plug-and-play routing protocol that exchanges the topology and traffic information to establish efficient global routing paths is proposed for wireless 3D CMPs to address these problems. The routing paths are established based on spanning trees, which are statically or dynamically optimized for given traffic patterns. Since the network optimization phase is light-weight, it can be carried out at any time if needed, for instance, when the configuration of a SiP, applications running on the SiP, or input data for the applications are changed.

A. Overall Procedure and Router Architecture

Routing paths are calculated by a single processor inside or outside a 3D CMP, based on the topological information collected from all chips. We assume that every vertical link controller has topology information on its chips. This assumption is reasonable since the horizontal information for chips is known a priori and can be embedded in the vertical link controller at the design time.

The topological information on a 3D CMP is exchanged when the system configuration is changed. Then, global routing paths are established or updated if needed. The five steps in the proposed procedure are given below.

- **Routing computation node selection:** A processor inside or outside the 3D CMP is manually selected as a "routing computation node." It is in charge of the routing path calculations and optimization. For example, node 7 has been selected as a routing computation node in Figure 2.
- **Topology request:** The routing computation node sends a "topology request" message (blue dotted line) to at least one vertical link controller in every chip to collect all the topological information.
- **Topology reply:** The vertical link controllers reply and send their topological information (red solid line) to the routing computation node. The topological information is represented by an $n \times n$ adjacency matrix, in which an element, $a_{i,j}$, represents a horizontal link between two vertical links, i and j, on the same chip, where n is the number of vertical links in each chip. Figure 2 shows an adjacency matrix for the bottom chip.
- **Routing computation and optimization:** The routing computation node calculates the global paths with a deadlock-free routing algorithm with the optimization techniques for spanning trees discussed in Sections III-B and III-C.
- **Routing table distribution:** The routing computation node distributes the routing table information to every router in the 3D CMP. The routing table information is routed along with the horizontal dimension, then routed along with the vertical one, as seen in Figure 3.

This detection-based approach is versatile for both regular and irregular topologies. For example, if a target topology is 3D mesh, the plug-and-play protocol can detect the topological regularity, and it uses the regular routing (e.g., XYZ routing). The topology-agnostic routing is used only when no topological regularity is detected. This can avoid any performance degradation due to the topology-agnostic routing when the topology is regular.

Figure 4 illustrates the router and vertical link controller. The router is a conventional VCT or wormhole router with routing tables and traffic counters. It has some horizontal channels depending on a given topology. For example, the example router has X-, Y+, and Y- channels while it does not have X+ channel. It also has local channels connected to a local processor or caches (not shown in this figure). Its vertical channels (Z+ and Z-) are connected to TX or RX inductors. It maintains traffic counters for dynamic network optimization (mentioned later) that simply count the number of packets injected into each destination.

The vertical link controller is attached to the router. It has horizontal topology information embedded at the time of design. It is a simple state machine and is in charge of replying and

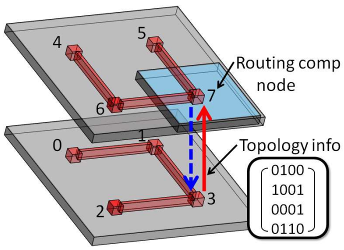


Fig. 2. Topology request and reply.

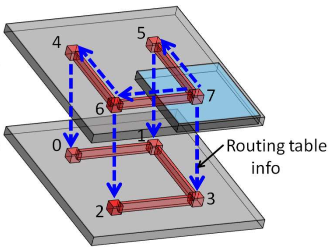


Fig. 3. Routing table update.

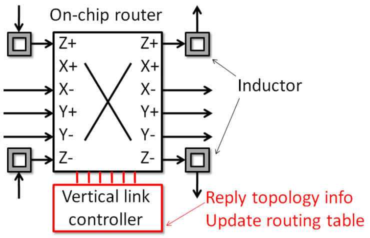


Fig. 4. Router and link controller.

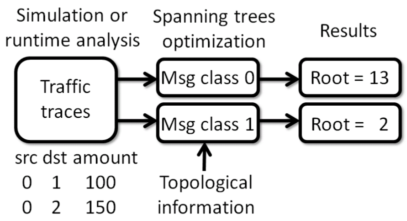
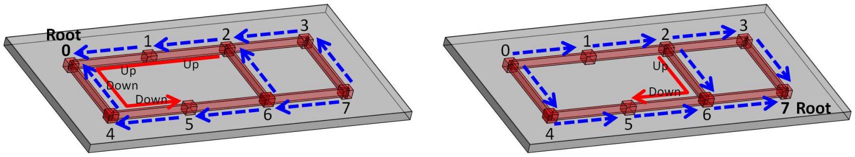


Fig. 5. Optimization flow



(a) Root node 0

(b) Root node 7

Fig. 6. Up*/Down* routing paths with different spanning tree roots.

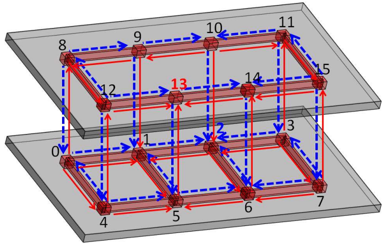


Fig. 7. Optimization result. Red solid lines indicate spanning tree for message class 0, while blue dotted lines indicate that for message class 1.

sending topology information to the routing computation node and updating the router's routing table if needed.

B. Spanning Tree Based Routing Algorithm

To route packets on irregular wireless 3D NoCs, a topology-agnostic routing algorithm without virtual channels is essential. Up*/Down* routing avoids deadlocks in irregular topologies without virtual channels, based on the assignment of direction (up or down) to network channels [12]. Figure 6 has two examples of spanning trees, each of which has a different spanning tree root (node 0 or 7). A legal path must traverse zero or more channels upward followed by zero or more channels downward to guarantee freedom from deadlocks while maintaining network reachability. For example, the red solid line in Figure 6(a) indicates a routing path from nodes 2 to 5. This routing path uses two upward channels followed by two downward channels.

Needless to say, the location of a spanning tree root significantly affects the hop count and utilization of links. For example, the routing path from nodes 2 to 5 in Figure 6(a) employs a non-minimal path, because the minimal path is prohibited by the spanning tree whose root is located at node 0; thus, four hops are required to send a packet. However, the minimal path in Figure 6(b) is allowed by the spanning tree whose root is located at node 7; thus, only two hops are required.

C. Spanning Trees Optimization

Two levels of granularity are proposed to optimize spanning trees: message-class-level and finer-level optimizations. The frequency of optimization and its impact are also discussed.

Message-class-level optimization: A message class is a group of messages that can share the same virtual channel without introducing protocol deadlocks. Up*/Down* routing generally does not require any virtual channels to avoid structural deadlocks. However, we have assumed a shared cache CMP architecture, in which a cache coherence protocol is running on the NoC. Since such cache coherence protocols typically induce end-to-end or request-and-reply protocol deadlocks, they are divided into multiple message classes, each of which uses one or more dedicated virtual channels, to achieve freedom from deadlocks.

Table I summarizes an example of virtual channel assignment for a directory-based cache coherence protocol. Three virtual channels are used for the protocol messages so that no end-to-end protocol deadlocks are induced. The network resources (i.e., buffers) of one virtual channel and those of another virtual channel are logically separated and do not introduce any cyclic dependencies between them; thus, we can assign different spanning trees to them, depending on their traffic patterns. For example, request messages from L1 caches (i.e., processors) to L2 cache banks move in the opposite direction of the reply messages from the L1 caches to the L2 caches. Since Up*/Down* routing introduces non-minimal paths and imbalanced link utilizations depending on the spanning tree roots and traffic patterns, their optimal spanning tree roots that can minimize the hop count will be different. Thus, an optimal spanning tree root should be selected for each message class to improve performance.

TABLE I
ASSIGNMENT OF VIRTUAL CHANNELS (VCS) FOR A CACHE COHERENCE PROTOCOL.

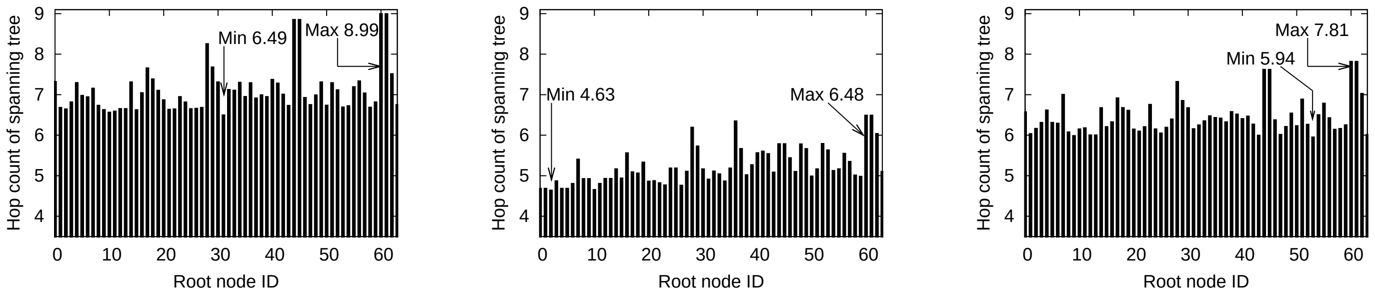
Message class 0
(VC0)
	Request *from* {L1$} *to* {L2$ bank}
	Request *from* {L2$ bank} *to* {L1$}

Message class 1
(VC1)
	Request *from* {L2$ bank} *to* {directory controller}
	Request *from* {directory controller} *to* {L2$ bank}

Message class 2
(VC2)
	Response *from* {L1$ *or* directory} *to* {L2$ bank}
	Response *from* {L2$ bank} *to* {L1$ *or* directory}

Finer-level optimization: It is possible to use multiple spanning trees or virtual channels for a certain message class. That is, an optimal combination of multiple spanning trees that can minimize the hop count can be assigned to a message class. The best spanning tree among them is selected for routing, depending on the source and destination pair in this case. Moreover, such a scheme of multiple spanning trees provides additional opportunities to further reduce the hop counts by virtual channel transitions at intermediate nodes. For example, a packet is first routed to an intermediate node along with a spanning tree (VC0), and then it is routed to the destination with another spanning tree (VC1). Note that such transitions in virtual channels should be done so that no cyclic dependencies between multiple spanning trees are introduced, e.g., VC transitions are allowed only either in ascending or descending order [13] (e.g., $\mathrm{VC}_i \mapsto \mathrm{VC}_{i-1} \mapsto \mathrm{VC}_{i-2}$). We will evaluate the assignment of two spanning trees for each message class (called Irr6(min)) in Section IV-C.

Optimization flow: Figure 5 outlines the optimization flow for spanning trees. First, the traffic patterns of applications on a target 3D CMP are pre-analyzed with system-level simulations or analyzed at run-time for a certain time period to generate a traffic trace. Then, an optimal spanning tree root is selected for each message class to minimize the average hop count of the trace and



(a) Message class 0
(b) Message class 1
(c) Message class 2

Fig. 8. Example of hop count distribution with different spanning tree roots (IS application on 64-tile CMP).

improve the execution time for the application; toward this end, we use the following cost function:

$$Cost = \sum_{s} \sum_{d} W_{s,d} H_{s,d}, \tag{1}$$

where $W_{s,d}$ denotes the number of packets from source router s to destination router d in the trace and $H_{s,d}$ denotes the hop count from s to d. Figure 7 shows an example of optimized spanning trees for a 16-tile 3D CMP that defines just two message classes for the sake of simplicity. Packets that belong to message class 0 are routed based on the red spanning tree whose root is node 13, while those belonging to message class 1 are routed based on the blue dotted spanning tree whose root is node 2. Note that this cost function only takes into consideration spatial locality, to fit run-time light-weight optimization of spanning trees. A more sophisticated cost function that takes into account temporal locality or traffic burstiness is also possible for off-line optimization of spanning trees.

Optimization frequency: The time to calculate the best spanning tree root for each message class is short. The calculation time for target wireless 3D CMPs with three message classes (see Section IV-A) is 0.008sec for 8-tile, 0.020sec for 16-tile, 0.315sec for 32-tile, and 9.376sec for 64-tile with an AMD 2.7GHz Opteron processor. Thus, it is possible to optimize routes at every boot time so that routing paths can be customized for the latest characteristics of applications. More frequent or run-time optimization of routes is also possible if the system has spare or escape virtual channels to remove cyclic dependencies formed across previous and next spanning trees. Dynamic reconfiguration schemes for interconnection networks, such as the Double scheme [14], can be used for this purpose.

Impact of spanning trees optimization: Figure 8 has an example of hop count distributions in a 64-tile wireless 3D CMP when the location of the spanning tree root is varied from nodes 0 to 63. Detailed parameters for the simulations and the evaluation environments will be presented in Section IV-A. There are three message classes each of which uses a single virtual channel in the network. For example, in the message class 0 (Figure 8(a)), the best spanning tree root that minimizes the hop count is node 31 and its average hop count is 6.49, while the worst spanning tree root is node 60 and its hop count is 8.99. There are significant differences in hop counts between Min and Max in all message classes. The proposed technique of optimizing spanning trees is significant because the optimization of routes enables us to "always" select the best spanning tree root that minimizes the hop count for each message class. We will explain performance benefits obtained with the optimization of spanning trees in Section IV-C.

IV. Practical Considerations and Experimental Results

The proposed plug-and-play routing protocol for wireless 3D CMPs is denoted an "irregular approach" since it can cope with any ad-hoc topologies that meet the minimum design rules stated in Section II-C. The prototype system of inductive-coupling reported in [10], on the other hand, uses a single ring-based topology, which significantly limits scalability; thus, the irregular approach is compared with 3D mesh networks instead of ring.

TABLE II
Irregular topologies to be tested (parameters).

	Topology	#routers	#CPUs	#L2$ banks	#Memories
8-tile	(2,1,4)	8	4	16	2
16-tile	(2,2,4)	16	4	32	4
32-tile	(4,2,4)	32	8	64	8
64-tile	(4,4,4)	64	8	128	16

TABLE III
Simulation parameters (processor, memory, and network).

Processor	UltraSPARC-III
L1 I/D cache size	64 KB (line:64B)
L1 cache latency	1 cycle
L2 cache bank size	256 KB (assoc:4)
L2 cache latency	6 cycle
Memory size	4 GB
Memory latency	160 (± 2) cycle
Router pipeline	[RC/VSA][ST][LT]
Buffer size	5-flit per VC (default)
Flit size	128 bit
Protocol	MOESI directory
# of message classes	3 (see Table I)
Control / data packet size	1 flit / 5 flit

A. Target CMP Architecture and Evaluation Environments

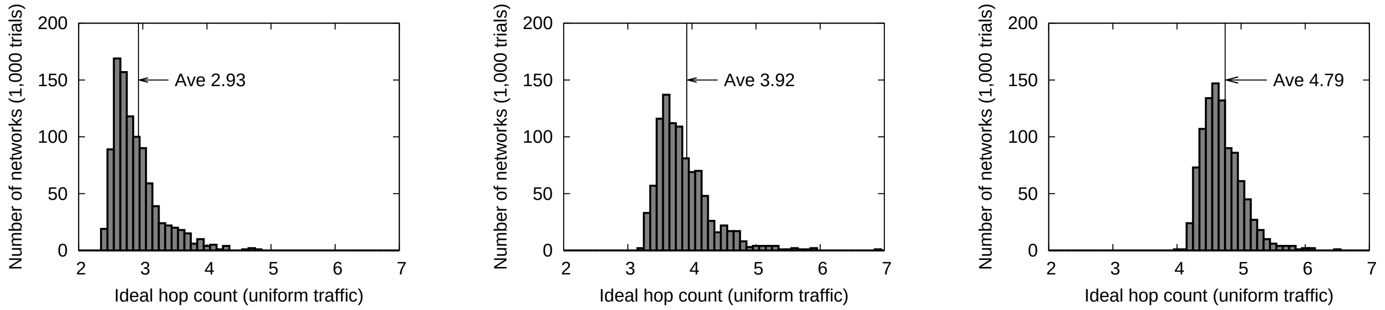
Table II summarizes the configurations for 8-tile, 16-tile, 32-tile, and 64-tile wireless 3D CMPs. Topology (x, y, z) denotes a wireless 3D CMP that consists of z chips where each chip consists of $x \times y$ tiles. The number of chips z is fixed at four in the irregular approach, while their sizes correspond to (2,1), (2,2), (4,2), and (4,4) for 8-tile, 16-tile, 32-tile, and 64-tile CMPs.

Table III lists the processor and network parameters. We used a full-system CMP simulator that combines GEMS [15] and Wind River Simics [16] to simulate the wireless 3D CMPs. We modified a detailed network model of GEMS to accurately simulate the proposed plug-and-play routing protocol. A directory-based MOESI coherence protocol that defines three message classes was used.

We used ten parallel programs from the OpenMP implementation of NAS Parallel Benchmarks (NPB) to evaluate the performance of applications in these communication schemes on the wireless 3D CMPs. Sun Solaris 9 operating system (OS) was running on the CMPs. These benchmark programs were compiled with Sun Studio 12 and were executed on Solaris 9 OS. The number of threads was set to four or eight, depending on the number of processors (e.g., four threads for 8-tile CMP).

B. Random Generation of Irregular Topologies

A wireless 3D CMP is a collection of various chips whose horizontal architecture is not known by individual chips; the proposed plug-and-play routing protocol can cope with such irregularities. One thousand irregular topologies were randomly generated for



(a) 16-tile (b) 32-tile (c) 64-tile

Fig. 9. Hop count distribution of randomly generated topologies. 8-tile results have been omitted due to page limitations. Average hop count in this case is 2.29.

each CMP configuration to evaluate the irregular approach so that each horizontal link appeared with 50% probability while each vertical one appeared with 100%. This is because all chips must have vertical links at pre-specified locations as defined in the minimum design rules, while they can use an arbitrary or customized topology for their intra-chip network. Note that a network topology that has all horizontal and vertical links is 3D mesh here, in which packets are routed with dimension-order routing (i.e., XYZ routing).

Figure 9 shows the hop count distribution of 1,000 randomly generated topologies for 16-tile, 32-tile, and 64-tile wireless 3D CMPs. Out of 1,000 trials for each network size, a network that had the closest hop count value to the average was selected for full-system CMP simulations. Communication latency typically and significantly affects the performance of applications on shared-memory CMPs. Thus, it is a good approximation where a network configuration that has the closest hop count value to the average represents the average performance of applications for 1,000 trials. For example, a network configuration whose hop count is 2.93 has been selected for the full-system simulations of the 1,000 random topologies in the 16-tile configuration.

C. Performance Improvements by Spanning Trees Optimization

One or more spanning tree roots are selected with the proposed technique of optimizing spanning trees for each message class of applications using the cost function (Equation 1) to minimize the average hop count. Four configurations are compared to find improvements to performance by optimizing spanning trees.

- **Irr3(max):** The worst spanning tree root that maximizes $Cost$ is selected for each message class of each application.
- **Irr3(min):** The best spanning tree root that minimizes $Cost$ is selected for each message class of each application.
- **Irr6(min):** The best pair of spanning tree roots that minimizes $Cost$ is selected for each message class of each application. Six VCs are used in total.
- **Irr(ideal):** Minimal paths without Up*/Down* restrictions are selected. They may cause deadlocks.

Figure 10 shows the hop counts for NPB applications with Irr3(max), Irr3(min), Irr6(min), and Irr(ideal). Again, if we can take into account the traffic patterns, we can always use the best routing path set. We can reduce the hop counts by up to 31.4% in the 64-tile case compared to the worst case. Also, we found that more than two spanning tree roots for each message class can further reduce the hop counts, but the benefits are small compared to the additional hardware cost of virtual channels. Thus, a single spanning tree root is sufficient for each message class in the CMPs that have 16 and 64 tiles.

Finally, Figure 11 shows the execution time for NPB applications with Irr3(max), Irr3(min), and 3D Mesh. 3D Mesh has 100% of horizontal links, while the irregular approach has only 50% of horizontal links. As can be seen, by taking into account traffic patterns, we can always use the best routing path set, and we can reduce the application execution time by up to 15.1% compared to the worst case. In addition, performance gap between Irr3(min) and 3D Mesh is much smaller than that between Irr3(min) and Irr3(max). These can be achieved with the run-time optimization of spanning trees with a light-weight cost function.

D. Energy Consumption

The irregular approaches with the optimization of spanning trees were evaluated in terms of the average energy consumption needed to transmit a single flit from the source to destination nodes. They are also compared with 3D Mesh. This energy per flit can be estimated as

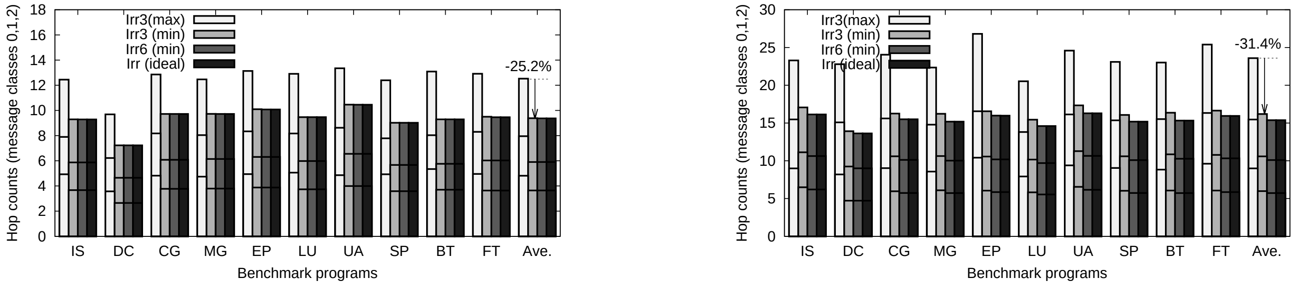
$$\begin{aligned} E_{flit} &= w(E^{1hop}_{router}h_{router} + E^{1hop}_{hlink}h_{hlink} + E^{1hop}_{vlink}h_{vlink}) \\ &= w(E_{router} + E_{hlink} + E_{vlink}). \qquad (2) \end{aligned}$$

Here, w represents the flit-width. h_{router}, h_{hlink}, and h_{vlink} represent the number of routers, horizontal links, and vertical links traversals on average. E^{1hop}_{router}, E^{1hop}_{hlink}, and E^{1hop}_{vlink} correspond to the energy consumed by transmitting single bit data via a router, a horizontal 2mm link, and a vertical link (i.e., inductor). The E^{1hop}_{router} was set to 0.20pJ in this evaluation, based on the post-layout simulations of on-chip routers when a 65nm CMOS process with a 1.2V supply voltage was used. The E^{1hop}_{hlink} was set to 0.43pJ, assuming that a semi-global interconnect whose wire capacitance was 0.20pF/mm (from ITRS 2007) was used for the 2mm horizontal links with repeaters inserted. Finally E^{1hop}_{vlink} was set to 0.14pJ from [5].

We calculated the $(E_{router} + E_{hlink} + E_{vlink})$ values of Irr(max), Irr(min), and 3D Mesh based on the hop count values (i.e., h_{router}, h_{hlink}, and $_{vlink}$) of all applications extracted from the full-system simulation results. Figure 12 shows the results obtained from evaluation, in which each bar represents its E_{router}, E_{hlink}, and E_{vlink} from the bottom. As shown, the irregular approach with the best spanning tree (Irr3(min)) reduces the energy consumption by up to 24.9% compared to that of the worst spanning tree. In addition, energy efficiency of Irr3(min) is relatively close to that of 3D Mesh that has twice number of horizontal links.

V. Conclusions

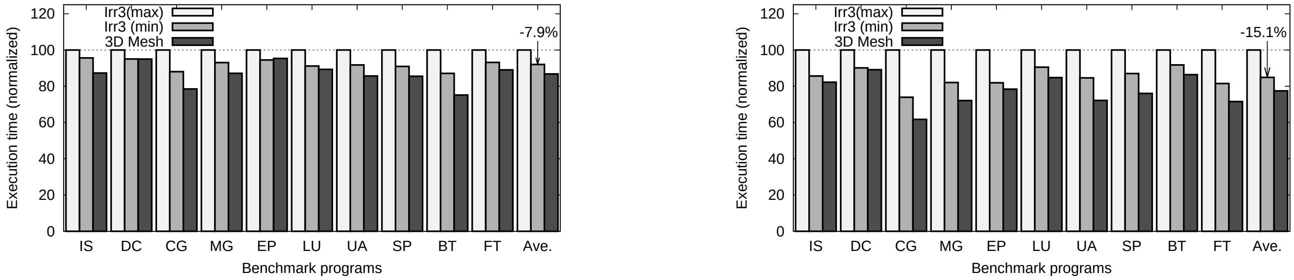
As future wireless 3D CMPs, we assume that chips (card-style computer components) are inserted to a micro cartridge that provides only power and clock pins to the chips. Packets are transferred between the chips wirelessly without going through the cartridge which typically limits the bandwidth due to pin-count limitation. Since card-style computer components are inserted when needed, we cannot expect any pre-determined topology for the system. Toward this end, we proposed a plug-and-play routing protocol that was able to cope with the irregularity of wireless 3D CMPs. The proposed routing protocol exchanges network information between all chips to establish routing paths that are free from deadlocks based on spanning trees. This approach is specific to inductive-coupling based 3D NoCs, and it is pointless



(a) 16-tile

(b) 64-tile

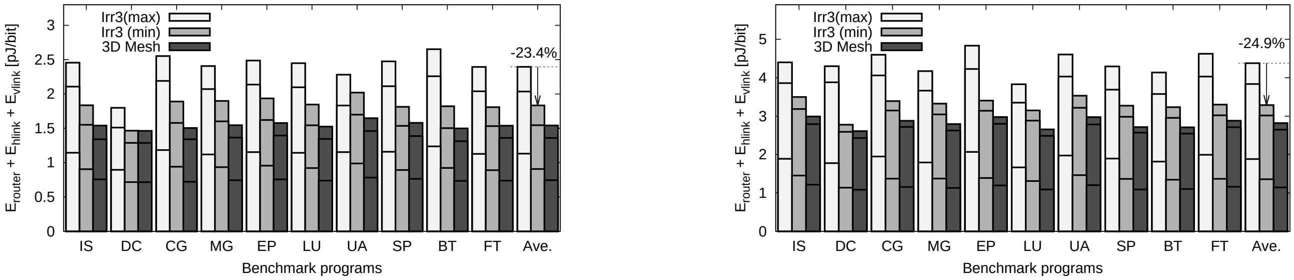
Fig. 10. Hop counts for irregular approach with different spanning trees. Each bar consists of hop counts for message classes 0, 1, and 2, starting from the bottom.



(a) 16-tile

(b) 64-tile

Fig. 11. Application execution times of 3D Mesh and irregular approaches with different spanning trees.



(a) 16-tile

(b) 64-tile

Fig. 12. Energy consumptions of 3D Mesh and irregular approaches with different spanning trees. Each bar consists of E_{router}, E_{hlink}, and E_{vlink}, starting from the bottom.

for TSV-based 3D chips since the network topology is fixed at design time and never changed afterwards in wired 3D systems. In addition, we proposed a new technique of optimizing spanning trees that selects the best spanning tree root for each message class to minimize the average hop count and reduce the application execution time. Full-system CMP simulations using parallel applications revealed that the proposed technique of optimizing spanning trees improves the application performance and energy consumption by up to 15.1% and 24.9%, respectively.

Acknowledgements This research was performed by the authors for STARC as part of the Japanese Ministry of Economy, Trade and Industry sponsored “Next-Generation Circuit Architecture Technical Development” program. The authors thank to VLSI Design and Education Center (VDEC) and JST CREST for their support. H. Matsutani was also supported by Grant-in-Aid for Research Activity Start-up #23800053.

References

[1] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. P. Shen, and C. Webb, “Die Stacking (3D) Microarchitecture,” in *Proceedings of the International Symposium on Microarchitecture (MICRO’06)*, Dec. 2006, pp. 469–479.

[2] K. Kumagai, C. Yang, S. Goto, T. Ikenaga, Y. Mabuchi, and K. Yoshida, “System-in-Silicon Architecture and its application to an H.264/AVC motion estimation fort 1080HDTV,” in *Proceedings of the International Solid-State Circuits Conference (ISSCC’06)*, Feb. 2006, pp. 430–431.

[3] W. R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A. M. Sule, M. Steer, and P. D. Franzon, “Demystifying 3D ICs: The Pros and Cons of Going Vertical,” *IEEE Design and Test of Computers*, vol. 22, no. 6, pp. 498–510, Nov. 2005.

[4] K. Kanda, D. D. Antono, K. Ishida, H. Kawaguchi, T. Kuroda, and T. Sakurai, “1.27-Gbps/pin, 3mW/pin Wireless Superconnect (WSC) Interface Scheme,” in *Proceedings of the International Solid-State Circuits Conference (ISSCC’03)*, Feb. 2003, pp. 186–187.

[5] N. Miura, H. Ishikuro, T. Sakurai, and T. Kuroda, “A 0.14pJ/b Inductive-Coupling Inter-Chip Data Transceiver with Digitally-Controlled Precise Pulse Shaping,” in *Proceedings of the International Solid-State Circuits Conference (ISSCC’07)*, Feb. 2007, pp. 358–359.

[6] N. Miura, D. Mizoguchi, M. Inoue, K. Niitsu, Y. Nakagawa, M. Tago, M. Fukaishi, T. Sakurai, and T. Kuroda, “A 1Tb/s 3W Inductive-Coupling Transceiver for Inter-Chip Clock and Data Link,” in *Proceedings of the International Solid-State Circuits Conference (ISSCC’06)*, Feb. 2006, pp. 424–425.

[7] J. Burns, L. McIlrath, C. Keast, C. Lewis, A. Loomis, K. Warner, and P. Wyatt, “Three-Dimensional Integrated Circuits for Low-Power High-Bandwidth Systems on a Chip,” in *Proceedings of the International Solid-State Circuits Conference (ISSCC’01)*, Feb. 2001, pp. 268–269.

[8] Y. Xie, G. H. Loh, B. Black, and K. Bernstein, “Design Space Exploration for 3D Architectures,” *ACM Journal on Emerging Technologies in Computing Systems*, vol. 2, pp. 65–103, Apr. 2006.

[9] A. Sheibanyrad, F. Petrot, and A. Janstch, *3D Integration for NoC-Based SoC Architectures*. Springer, 2010.

[10] H. Matsutani, Y. Take, D. Sasaki, M. Kimura, Y. Ono, Y. Nishiyama, M. Koibuchi, T. Kuroda, and H. Amano, “A Vertical Bubble Flow Network using Inductive-Coupling for 3-D CMPs,” in *Proceedings of the International Symposium on Networks-on-Chip (NOCS’11)* , May 2011, pp. 49–56.

[11] J. Ouyang, J. Xie, M. Poremba, and Y. Xie, “Evaluation of Using Inductive/Capacitive-Coupling Vertical Interconnects in 3D Network-on-Chip,” in *Proceedings of the (ICCAD’10)* , Nov. 2010, pp. 477–482.

[12] M. D. Schroeder, A. D. Birrell, M. Burrows, H. Murray, R. M. Needham, and T. L. Rodeheffer, “Autonet: A High-speed, Self-configuring Local Area Network Using Point-to-point Links,” *IEEE Journal on Selected Areas in Communications*, vol. 9, pp. 1318–1335, Oct. 1991.

[13] M. Koibuchi, A. Jouraku, and H. Amano, “Descending Layers Routing: A Deadlock-Free Deterministic Routing using Virtual Channels in System Area Networks with Irregular Topologies,” in *Proceedings of the International Conference on Parallel Processing (ICPP’03)*, Oct. 2003, pp. 527–536.

[14] T. M. Pinkston, R. Pang, and J. Duato, “Deadlock-Free Dynamic Reconfiguration Schemes for Increased Network Dependability,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 14, no. 8, pp. 780–794, 2003.

[15] M. M. K. Martin *et al.*, “Multifacet General Execution-driven Multiprocessor Simulator (GEMS) Toolset,” *ACM SIGARCH Computer Architecture News (CAN’05)*, vol. 33, no. 4, pp. 92–99, Nov. 2005.

[16] P. S. Magnusson *et al.*, “Simics: A Full System Simulation Platform,” *IEEE Computer*, vol. 35, no. 2, pp. 50–58, Feb. 2002.