Processor Tile Shapes and Interconnect Topologies for Dense On-Chip Networks

Zhibin Xiao and Bevan M. Baas, *Senior Member, IEEE*

***Abstract*— We propose two eight-neighbor, two five-nearest-neighbor, and three six-nearest-neighbor interconnection topologies for many-core processor arrays—three of which use five-sided or hexagonal processor tiles—which typically reduce application communication distance and result in an overall application processor that requires fewer cores and lower power consumption. A 16-bit processor with the appropriate number of input and output ports is implemented in all topologies and tile shapes. The hexagonal and five-sided processor tiles and arrays of tiles are laid out with industry standard automatic place and route design flow and Manhattan-style wires without full-custom layout. A 1080p H.264/AVC residual video encoder and a 54 Mb/s 802.11a/g OFDM wireless local area network baseband receiver are mapped onto all topologies. The six-neighbor hexagonal tile incurs a 2.9% area increase per tile compared with the four-neighbor 2-D mesh, but its much more effective interprocessor interconnect yields an average total application area reduction of 22% and an average application power savings of 17%.**

***Index Terms*— CMOS digital integrated circuits, digital signal processing (DSP), hexagonal processor, interconnection topology, many-core processor, multimedia, network on chip (NoC).**

I. Introduction

TILED architectures that integrate two or more independent processor cores are called multicore processors. Manufacturers typically integrate multicore processors into a single integrated circuit die [known as chip multiprocessors (CMP)]. CMPs that integrate tens, hundreds, or thousands of cores per die are called many-core chips. For global wires in a many-core chip, their delay and power scale up with the technology compared with gates as their length is nearly constant if the chip size stays the same [1]. Thus, many-core chips that use scalable interconnects and avoid global long wires will attain higher performance.

Network-on-chip (NoC) approaches are used to connect large number of processors on a single chip because they perform better than less scalable methods such as global shared buses that use global long wires. There exist many design alternatives for NoC architectures, which differ mainly in switching policy, topology, and routing algorithms.

Network topologies define how nodes are placed and connected, affecting the latency, throughput, area, and power of a network. Because of its simplicity and the fact that processor tiles are traditionally square or rectangular, the nearest-neighbor 2-D mesh topology is a natural solution for both dynamic and static on-chip communication architectures. Efficient mapping applications, however, can be a challenge for cases that require communication between processors that are not adjacent to the 2-D mesh. This condition could require processors to forward data for static interconnection architectures, and intermediate routers for dynamic router-based NoCs. The power consumption and communication latency also increase, as the number of routing processors or routers between two communicating cores increases. There exist other common topologies for NoCs such as 2-D torus, Spidergon, fat tree and higher dimensional meshes and tori, which provide higher routing capability and communication bandwidth with costs of higher wire density and longer global wires. Furthermore, topologies with irregular layouts present significant challenges for many-core implementations especially with the number of cores per die expected to soon reach thousands and more.

For many applications mapped onto homogeneous chip multiprocessors, communication between processors is often largely localized [2], [3], which may result in local mapping congestion; an increase of local connectivity can ease such congestion. This motivates us to propose new topologies with increased local connectivity while keeping much of the simplicity of a mesh-based topology. This paper targets low-complexity and scalable topologies that increase application performance, reduce communication energy, avoid global wires, and ease physical implementation. We propose regular and scalable topologies combined with tile shapes for dense interconnection of many-core arrays, which result in an overall application processor with fewer cores and a lower total communication length. The main contributions of this paper can be summarized as follows.

1) Seven NoC topologies are proposed and compared with the common 2-D mesh including two eight-neighbor topologies, two five-neighbor topologies, and three six-neighbor topologies. Three of them use hexagonal-shaped or five-sided house-shaped processor tiles. The proposed topologies are analytically compared in terms of geometric property and worst-case communication distance for different communication patterns.

Manuscript received August 16, 2012; revised January 16, 2013 and March 18, 2013; accepted May 9, 2013. Date of publication June 25, 2013; date of current version May 20, 2014. This work was supported in part by ST Microelectronics, Intel, UC Micro, the NSF under Grant 0430090 and CAREER Award 0546907, SRC GRC under Grant 1598, CSR under Grant 1659, Intellasys, S Machines and the C2S2 Focus Center, one of six research centers funded under the Focus Center Research Program, and a Semiconductor Research Corporation entity.

Z. Xiao is with Oracle America Inc., Santa Clara, CA 95050 USA (e-mail: zhibin.xiao@oracle.com).

B. M. Baas is with the Electrical and Computer Engineering Department, University of California, Davis, CA 95616 USA (e-mail: bbaas@ucdavis.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2013.2265937

TABLE I
CHARACTERISTICS OF VARIOUS REGULAR TOPOLOGIES FOR AN HOMOGENOUS MANY-CORE ARRAY WITH $n \times n$ PROCESSORS WHERE n IS THE NUMBER OF PROCESSORS ON ONE EDGE AND $n \geq 2$

Topology	Degree	Maximum Links Hops	Link Number	Diameter	Bisection	Clustering Degree
2-D Mesh	4	0	$2n(n-1)$	$2(n-1)$	n	0
2-D Torus	4	1	$2n^2$	n	$2n$	0
8-8 Rect	8	1	$4n^2-6n+2$	$n-2$	$3n-2$	0.86
8-4 Rect	8	1	$2n^2-n-2$	$(n \mod 2)+n$	$2n$	0.21
5-5 House and Rect	5	0	Note[+]	$(2n-3)$[*]	n	0.30
6-6 Hex and Rect	6	0	$3n^2-4n+1$	$n+\lfloor\frac{n-2}{2}\rfloor$	$2n-1$	0.40

[+] Omitted because of space limitation. The total number of links for 5-5 House and Rect is: $n(n-1)+n(\lfloor\frac{n-1}{2}\rfloor)+(2n-1)(\lceil\frac{n-1}{2}\rceil)$.
[*] This is for $n \geq 4$. If $n \leq 3$, the diameter of the topology is: $n+\lfloor\frac{n-1}{2}\rfloor$.

2) A complete functional H.264/AVC residual encoder and an 802.11a/g OFDM baseband receiver are mapped onto all topologies for realistic comparisons.
3) All seven topologies including the hexagonal and house-shaped processor tiles are physically implemented in 65-nm CMOS using standard cells and Manhattan-style wires without full-custom layout. The final layouts are all design rule check (DRC) and LVS clean.

The remainder of this paper is organized as follows. Section II reviews related work. Section III describes and evaluates the proposed interprocessor communication topologies. Section IV presents mapping of two applications onto a 2-D mesh and all proposed topologies. Section V describes the physical design flow and the approach to implement the nonrectangular processor tiles. Section VI presents the chip implementation results and Section VII concludes this paper.

II. RELATED WORK

Many topologies were used for on-chip interprocessor communication, such as buses, meshes, tori, binary trees, octagons, hierarchical buses, and custom topologies for specific applications [4]. The low complexity 2-D mesh was used by most fabricated many-core systems including RAW [5], AsAP [6], [7], Intel 80-core [8], TILE64 [9], AsAP2 [10], [11], and Intel 48-core single-chip cloud computer [12].

Prior research was reported using hexagonal interconnections for on-chip wire routing and off-chip multiprocessor communication. Chen *et al.* [13] proposed a Y architecture for on-chip interconnections and showed that it can increase communication throughput by 20.6% over the 2-D mesh with Manhattan-style wires. Zhou *et al.* [14] proposed a hierarchical three-way interconnection, Y tree architecture, for hexagonal processors. These two papers only theoretically proposed hexagonal interconnection architecture and showcase the throughput benefit only if non-Manhattan style wires were used. Shin [15] proposed a hexagonal mesh for the interconnection of multiple processors in a system, which was demonstrated to have higher communication performance and robustness than other topologies. Furthermore, Decayeux and Seme proposed a 3-D hexagonal network as an extension of 2-D hexagonal networks [16]. As mentioned before, such off-chip hexagonal networks were used to connect computation nodes, which is different from our proposed on-chip hexagonal-shaped processor tiling.

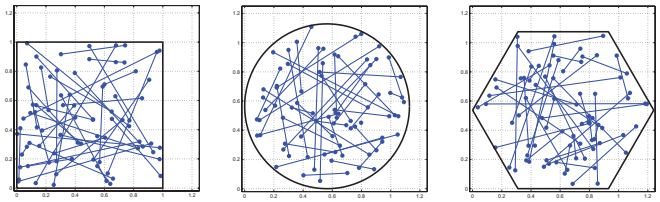
Becker *et al.* [17] developed a hexagonal field-programmable analog array in a 0.13-μm CMOS technology. The basic building block was a hexagonal analog circuit block which communicated with six neighbors. Extension to a many-core processor was similar in topology, but very different in terms of impact on tile area and total application interconnect. Malony studied the 2-D regular processor arrays, which are geometrically defined based on nearest-neighbor connections and space-filling properties [18]. He theoretically proved that the hexagonal array is the most efficient topology in emulating other topologies by analyzing the geometric characteristics.

III. INTERPROCESSOR COMMUNICATION TOPOLOGIES AND PROCESSOR SHAPES

NoC topologies can be analyzed by a few criteria [19] as follows.

1) *Degree*: The number of direct neighbors for one node. A high degree allows more nodes to communicate directly with low latency.
2) *Diameter*: The largest number of hops between any two nodes. A small diameter shows low maximum latency of a network.
3) *Bisection*: The minimum number of links to be removed to separate a network into two equal ones. A high bisection shows a high bandwidth yielding high throughput.
4) *Number of Links*: The total number of bidirectional links in a network.
5) *Clustering Degree*: Also called clustering coefficient, is a measure of degree to which nodes in a network tend to cluster together. The local clustering degree for a node i can be defined as: $2l_i/(n_i(n_i-1))$, where n_i is the number of direct neighbors and l_i is the number of links between its neighboring nodes. A high clustering degree shows that local nodes close to each other are strongly connected.
6) *Max Link Hops*: The maximum hops that a link can cross after the topology is physically mapped to a 2-D chip. This is a criteria proposed in this paper to measure the length of global wires of a topology.

The above criteria can be used to compare various topologies and provides an initial indication on performance. The first two rows of Table I lists the characteristics of two popular



(a) Square (b) Circle (c) Hexagon

Fig. 1. Example tiles of constant area with random uniformly distributed wire endpoints.

topologies 2-D mesh and 2-D torus. 2-D mesh has a maximum degree of four and a maximum link hop equals to zero as all of the links are nearest neighbors. For a $n \times n$ array, 2-D mesh has a number of links equals to $2n(n-1)$, a diameter equals to $2(n-1)$, bisection n, and a clustering degree equals to 0. Compared with 2-D mesh, 2-D torus has the same degree, more links, smaller diameter, higher bisection bandwidth, and the same clustering degree. All of these criteria show 2-D torus could achieve higher throughput and lower latency at the cost of more long nonnearest neighbor links.

This paper explores low-complexity topologies with higher degree, larger number of links, smaller diameter, higher bisection compared with 2-D mesh. We also limit the maximum link hops being less than or equal to one to avoid global long wires. These requirements result in proposed topologies that have a strong local connectivity with a nonzero clustering degree. In the following sections, several topologies combined with nonrectangular processor shapes are proposed and analyzed.

A. Processor Tile Shapes

To the best of our knowledge, all previously fabricated VLSI processors are of a rectangular shape, often nearly square. We can intuitively reason for a circular shape might allow shorter wires for a given netlist, resulting in smaller area and wire capacitance.

Fig. 1 shows a simple wiring experiment by randomly placing one million wires in a square, a circle, and a hexagon with equal size. The circular tile yields a 2.2% reduction in total wire length compared with a square tile. On the negative side, it is clear that circles do not pack together without wasted space between tiles. On the positive side, circles pack with six neighbors while rectangles obviously have only four. In contrast to the circle, the hexagonal shape does pack efficiently without gaps between tiles and it retains the six-nearest-neighbor property. The same wiring experiment shows a hexagonal tile achieved a 1.8% reduction in total wire length compared with the square tile. A reduction in total wire length yields a pure benefit in area, energy, and delay for processor tile design. This simple experiment motivates us to explore different processor tile shapes for many-core processor design. The inclusion of common rectangular blocks such as memory arrays in a processor tile increases routing congestion but is shown in Section VI to be tolerable. In addition, we show that Manhattan-style wire routing is fully compatible with nonrectangular tile shapes.

B. Proposed Topologies

The eight different topologies in combination with processor tile shapes are shown in Fig. 2. Switch fabrics are assumed to reside inside each processor tile. The well-known 2-D mesh in Fig. 2(a) is used as the baseline topology for comparison. All topologies are named by the following: 1) the total number of direct interconnection links; 2) the number of nearest-neighbor interconnection links, where nearest neighbors are defined as directly connected processors that touch at the edge or the vertices; and 3) the processor's shape. For example, the baseline 2-D mesh is named 4-4 Rect where tiles are rect-shaped and connected by four links, all of which are nearest-neighbor interconnect links.

The next logical extension of the 2-D mesh is to include the four diagonal processors in an eight-neighbor arrangement named 8-8 Rect as shown in Fig. 2(b), where each rect tile can directly communicate with eight neighbors. This approach increases routing congestion in the tile corners because of the four (unidirectional) links that pass through each corner [the dashed lines in Fig. 2(b)].

The third topology is an eight-neighbor mesh (8-4 Rect) as shown in Fig. 2(c), where the baseline 2-D mesh is augmented with direct connections with processors two tiles away. In this case, the pass-through routes are not just in the corners, but pass through the entire tile.

Fig. 2(d) shows a five-nearest-neighbor topology (5-5 House) where each tile is a house-shaped pentagon. There are various house shapes and the center-to-center Euclidean distances between a tile's center and its five neighbors are not equal. The center of a house-shaped tile, however, can be chosen hence the Euclidean distances from the center to all five vertices are equal, which yields only one type of house-shaped tile where the rectangular shape at the bottom is a square. If the square shape has an edge length of w, the center-to-center distance for three of the five connections is w and the other two connections have a length of $w * \sqrt{(2+\sqrt{2})/2}$.

Fig. 2(e) shows an alternative five-neighbor topology (5-5 Rect Alt. Offset) where every other rows of rect tiles are offset. The 5-5 Rect Alt. Offset has the same interconnection topology as the 5-5 House. All processors are square shaped with an edge length of w. The center-to-center Euclidean distance between two processor tiles can be either w (if tiles are aligned) or $\sqrt{5}/2 * w$ (if the tiles are in an offset position). This topology has the advantage of a regular processor shape while achieving the same routing capability as the house-shaped tile topology.

Our sixth proposed interconnect topology is the six-nearest-neighbor array using hexagonal-shaped processor tiles, as shown in Fig. 2(f). The processor center-to-center Euclidean distance is $\sqrt{3} * w$ if the length of the hexagon edge is w. The hexagonal grid is commonly used in mobile wireless networks because of its desirable feature of approximating circular antenna radiation patterns and its optimal characteristic of six nearest neighbors. The symmetry and space-filling property make the hexagonal processor tile topology an attractive design option for many-core processor tiles.

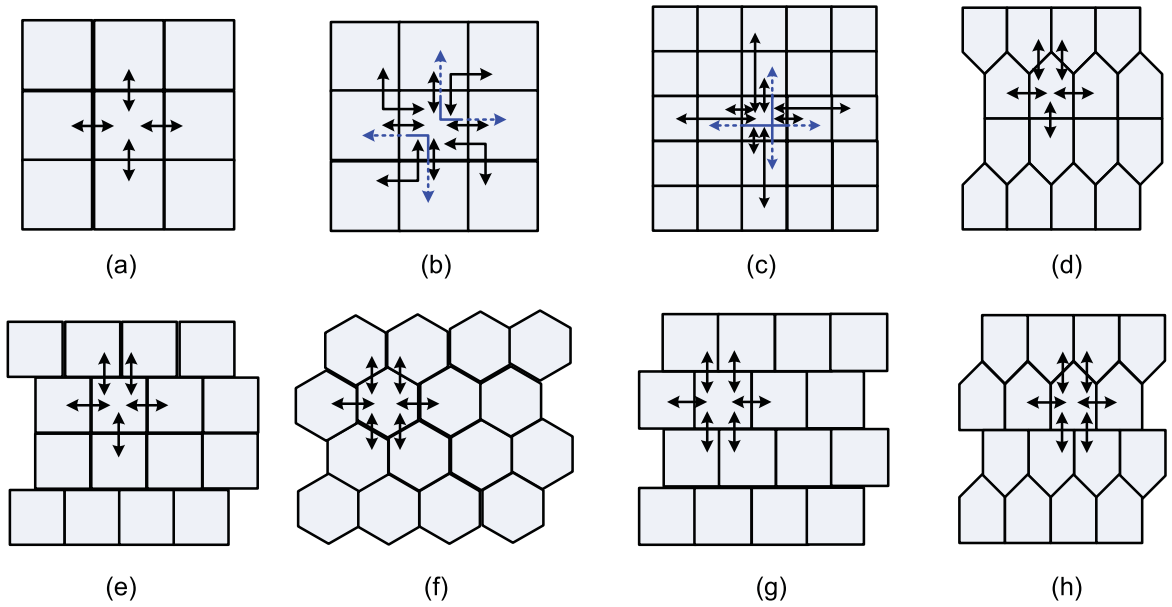


Fig. 2. (a) Baseline 2-D mesh (4-4 Rect). Seven proposed topology/shape combinations: (b) 8-8 Rect, (c) 8-4 Rect, (d) 5-5 House, (e) 5-5 Rect Alt. Offset, (f) 6-6 Hex, (g) 6-6 Rect Offset, and (h) 6-6 House Offset. (Designs are named using: the total number of interconnection links, the number of nearest-neighbor interconnection links, and the processor's shape.)

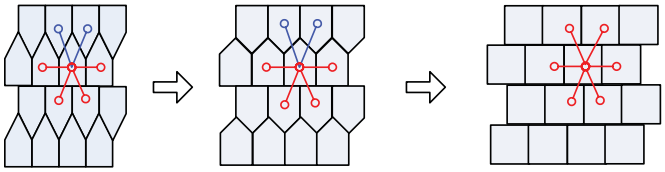


Fig. 3. Spectrum of six-neighbor topologies with offset row house-shaped tiles, which differ in the area of the triangle roof of the house shape. The length of the blue and red links is different.

Fig. 2(g) shows our seventh topology named 6-6 Rect Offset where every row of the tiles is offset hence each tile has six nearest neighbors. For tiles with height h and width w, the center-to-center distance in the horizontal direction is clearly w. For adjacent tiles in the row above and below, the center-to-center Euclidean distance is $\sqrt{w^2/4+h^2}$. Thus, if we set $w=\sqrt{w^2/4+h^2}$, or $h=\sqrt{3}/2*w$, then all six neighbors will reside at equal center-to-center Euclidean distances.

Fig. 2(h) shows the eighth topology (6-6 House Offset) where every neighboring rows of house-shaped tiles are offset hence each tile has six neighbors. As shown in Fig. 3, there are a spectrum of topologies that fall into this category where the triangle roof of the house-shaped tile can have varying area. There is, however, no geometrically optimal topology with six equal Euclidean distance neighbors. If the area of the roof triangle is zero, it becomes the 6-6 Rect Offset topology, which has the advantage of equal center-to-center Euclidean distances for all six neighboring tiles, as shown in Fig. 2(g). Therefore, we will consider only the 6-6 Rect Offset for this type of topology in the following sections.

The center-to-center distance can be used to represent the communication link length between two processor tiles. Table II shows the number of different types of communication links and the corresponding link length for all topologies. For comparison purpose, the link lengths are calculated based on both Euclidean and Manhattan rules. As shown in Table II, if Euclidean rule is used, the 4-4 Rect, 6-6 Hex and 6-6 Rect Offset have only one type of communication link because of equal center-to-center Euclidean distance. The 8-8 Rect, 8-4 Rect, 5-5 House, and 5-5 Rect Offset topologies have two types of links because of the unequal center-to-center Euclidean distance between processor tiles. If Manhattan rule is used, all topologies have two types of links except the 4-4 Rect 2-D mesh. The 6-6 Hex and 6-6 Rect Offset have two short links and four long links instead.

Because of limitations of the current wafer sawing technologies, chips from round wafers are traditionally square or rectangular. The opportunities and limitations of nonrectangular processors on a chip are analogous to nonrectangular chips on a wafer. For a rectangular chip composed of nonrectangular processors, there are areas on the periphery of the chip in which processors cannot be placed for the topologies shown in Fig. 2(d)–(h). Fig. 4 shows the percentage of unavailable area for the four topologies with varying processor array sizes. If the processor array size is larger than 30 by 30, this area overhead becomes less than 2.7% of the total chip area for the hexagonal-shaped tile array and 2.0% for the house-shaped tile array. The overhead area for type (e) and (g) is less than 1.7% of the total chip area. In practice, these areas could be filled with useful chip components such as decoupling capacitors, or portions of hardware accelerators, global shared memory modules, global I/O circuits or power conversion circuits. For the house-shaped and hexagonal-shaped tiles, physically it is possible to integrate both rectangular and nonrectangular modules into the unavailable areas if the DRC is not violated.

C. Performance Evaluation

Table I also lists the characteristics of all proposed topologies for an homogenous many-core array with $n \times n$ processors. Compared with 2-D mesh, all proposed topologies have larger node degree, smaller diameter, larger or equal bisection bandwidth, and larger clustering degree. Among the

TABLE II
EUCLIDEAN AND MANHATTAN LINK LENGTHS FOR ALL TOPOLOGIES WITH ONE UNIT OF LENGTH EQUAL TO THE SQUARE ROOT OF THE AREA, WHICH IS ONE FOR ALL TOPOLOGIES AND SHAPES

Topology	Nearest-Neighbor Link			Longer Link		
	Number	Euclidean Distance	Manhattan Distance	Number	Euclidean Distance	Manhattan Distance
4-4 Rect	4	1.00	1.00	0	—	—
8-8 Rect	4	1.00	1.00	4	1.41	2.00
8-4 Rect	4	1.00	1.00	4	2.00	2.00
5-5 House	3	0.95	0.95	2	1.24	1.51
5-5 Rect Alt. Offset	3	1.00	1.00	2	1.12	1.50
6-6 Hex	6(2)*	1.07	1.07	0(4)*	—	1.47
6-6 Rect Offset	6(2)*	1.07	1.07	0(4)*	—	1.46

* The 6-6 Hex and 6-6 Rect Offset have six nearest-neighbor links using Euclidean wires. The two topologies, however, have two nearest-neighbor links and four longer links using Manhattan-style wires.

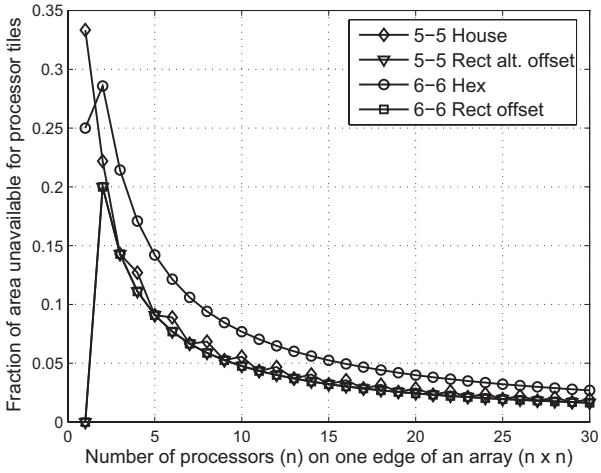


Fig. 4. Fraction of area unavailable for processor tiles in a nonmesh array ($n \times n$) for type d–g in Fig. 2: 5-5 House, 5-5 Rect Alt. Offset, 6-6 Hex, and 6-6 Rect Offset, respectively.

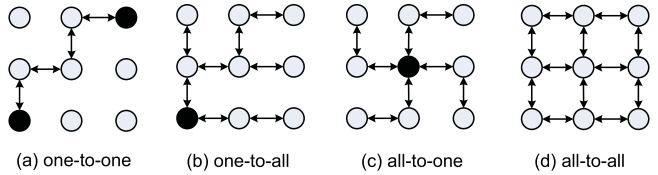


Fig. 5. Example of four communication patterns on a 3×3 array with 4-4 Rect topology and four-port processors. (a) One-to-one. (b) One-to-all. (c) All-to-one. (d) All-to-all.

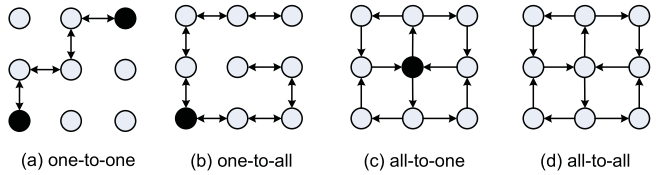


Fig. 6. Example of four communication patterns on a 3×3 array with 4-4 Rect topology and two-port processors. (a) One-to-one. (b) One-to-all. (c) All-to-one. (d) All-to-all.

proposed topologies, the 8-8 Rect topology has the largest bisection, smallest diameter, and largest clustering degree, which shows lower maximum latency and high maximum throughput. Although the 8-4 Rect topology has a high degree of eight, the smallest clustering degree shows 8-4 topology does not fit for applications with many local communications. This insight will be demonstrated by the application mapping results presented in the following section. The advantage of the five-neighbor and six-neighbor topologies is that global long wires are not required.

We can also analyze the performances of the proposed seven topologies using the worst-case communication distance of four basic communication patterns that include one-to-one communication (in which two processors at opposite corners of the processor array communicate with each other), one-to-all broadcast (in which one corner processor broadcasts data to all the other processors), all-to-one communication (in which all processors communicate with the processor in the middle of the array), and all-to-all communication (in which every processor communicates with all other processors).

The number of neighboring interprocessor communication links and the number of input ports determine the local communication capability of a topology. The input port of a processor is the communication interface including buffers and related circuits. The number of input ports can be less than the number of neighboring interconnection links of one processor.

As shown in Fig. 2, depending on the number of interconnection links, the topologies require a different number of input ports to make maximal use of nearest-neighbor interconnections. The baseline 4-4 Rect mesh requires four input ports and the two eight-neighbor Rect topologies require eight input ports. The house-shaped tile and hexagonal-shaped tile topologies require five ports and six ports, respectively. The increase of the number of input ports incurs significant hardware overhead in terms of buffers and related communication circuitry. The number of input ports into the local processor, however, can be less than the number of neighboring interconnections, in which processors are capable of talking to all connected neighboring processors but not at the same time.

Fig. 5 shows an example of four communication patterns on a 3×3 array with a 4-4 Rect topology and two-port processors. In this case, each node can be configured to connect bidirectionally with up to four nearest neighbors. The worst-case distance for one-to-one communication is four hops, as shown in Fig. 5(a). Fig. 5(b) shows the routes to allow one corner node to communicate with all the other nodes. The worst-case distance is also four hops. Fig. 5(c) shows the routes to allow all processors to communicate with the center node and the worst-case distance is two hops. For all-to-all communication, the worst-case distance is four hops as shown in Fig. 5(d).

Fig. 6 shows a similar 4-4 Rect topology with four-port processors. In this case, each node can be configured to

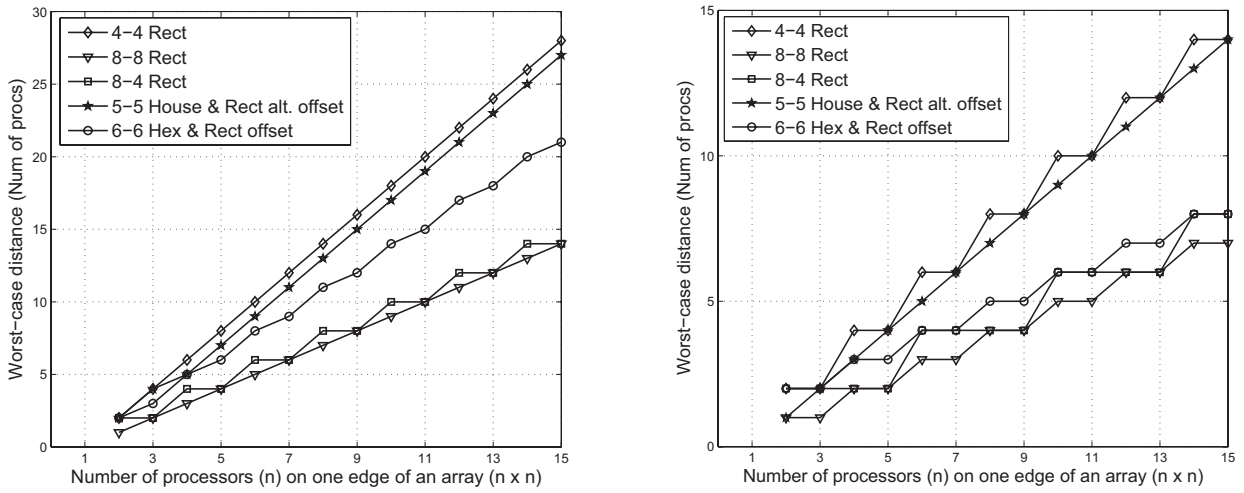


Fig. 7. Comparisons of the worst-case communication distance across a processor array ($n \times n$) for different topologies where the number of input ports of each processor is equal to the number of interconnection links for four basic communication patterns. (a) One-to-one, one-to-all, and all-to-all. (b) All-to-one.

connect with two nearest neighbors as inputs and four nearest neighbors as outputs. The routes in one-to-all, all-to-one, and all-to-all communications are different from previous example. The worst-case distances of one-to-one and one-to-all communications are not affected by the number of ports, which are still four hops. Fig. 6(b) and (c) shows the routes for all-to-one and all-to-all communication patterns are the same. The worst-case distances of all-to-one and all-to-all communications are three and four hops, respectively. Compared with the four-port case, the total communication distance for all-to-one and all-to-all increases because of the use of unidirectional wires and the two-way communication between two nodes may be asymmetrical.

The following subsections discuss both the case with the same number of input ports as the neighboring interconnections and the case with a limitation of two input ports for all proposed topologies.

1) Varying Number of Input Ports: For each proposed topology, the worst-case communication distances of one-to-one, one-to-all, and all-to-all communications are the same as the diameter of the topology, as shown in Table I. As for different topologies, Fig. 7(a) shows that the 4-4 Rect, 5-5 House, and 5-5 Rect Alt. Offset have similar worst-case communication distances which are approximately linearly proportional to the size of the array. The worst-case communication distances of the 6-6 Hex and 6-6 Rect Offset topologies are shorter than those of the 4-4 Rect, 5-5 House, and 5-5 Rect Alt. Offset topologies. The two eight-neighbor Rect meshes have the shortest worst-case communication distances because of more interconnection links.

Fig. 7 shows the all-to-one worst-case communication distances for all seven topologies. The performance trends are similar to the one-to-one, one-to-all, and all-to-all communication cases. The 6-6 Hex and 6-6 Rect Offset topologies show very close performance to the two eight-neighbor Rect meshes, which have two more sets of communication links.

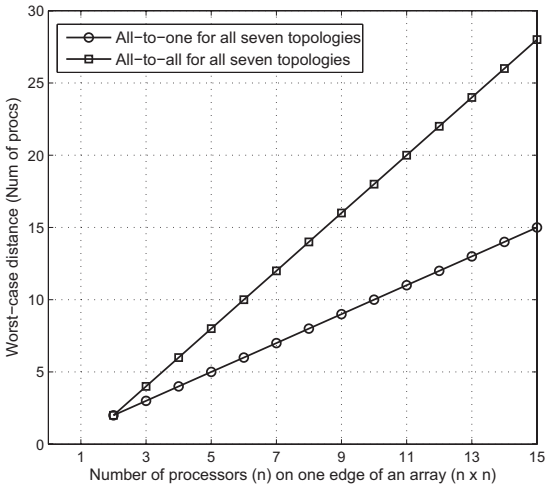


Fig. 8. Comparisons of the worst-case communication distance across a processor array ($n \times n$) with a limitation of two input ports for each processor.

2) Two Input Ports: Through limiting the number of input ports of the local processor to two, all topologies have the same one-to-one and one-to-all performances as the architecture with varying number of ports [Fig. 7(a)]. For the all-to-one and all-to-all communication patterns, the topologies with more links have the same worst-case communication distance as the topologies with fewer links if one processor has only two input ports, as shown in Fig. 8. This shows that a reduction of input ports decreases the performances of the topologies with more links in the all-to-one and all-to-all communications.

For simple single-issue processors that normally consume not more than two operands per clock cycle, adding more than two input ports may not have the benefits as shown in the worst-case analyses with varying number of input ports. Of course, there may be benefits if the processor uses complex instruction styles that process more than two operands per cycle. For many cases, it is attractive to use two input ports for all proposed topologies, in which the hardware overhead is

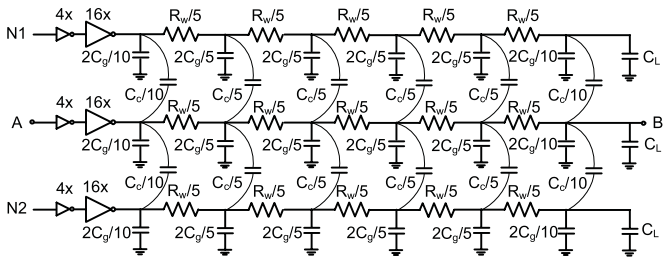


Fig. 9. Π5 lumped *RC* circuit model used to simulate the wire delay for different shaped processor tiles considering crosstalk effects between the main wire transmitting signal from A to B and two adjacent wires N1 and N2. R_w, C_g, and C_c are metal wire resistances, ground capacitance,s and coupling capacitances from adjacent intralayer wires, respectively.

minimized without affecting the communication performance too much. This limitation is justified by mapping two complex applications onto the proposed topologies in Section IV.

D. Interconnect Wire Delay

All discussed topologies permit an easy tiling of processors for dense on-chip networks without very long global wires. The two topologies in Fig. 2(b) and (c) have a maximum global interconnect link length no more than the dimension of one processor tile. There are no global wires for 2-D mesh, house-shaped, and hexagonal-shaped tile topologies. The actual delay of local interconnect wires, which is proportional to the size of processor tiles, depends on the physical position of the switch fabrics inside the local processor tile. Table III shows the estimates of the maximum interconnect link lengths and maximum link length differences for the square-shaped, house-shaped, and hexagonal-shaped processor tiles based on three nominal processor tile sizes 0.04, 4, and 36 mm^2, which approximate the scaled area of one processor tile in 32-nm CMOS for AsAP2 [10], [11], TI C64x DSP [20], and the Intel Sandy Bridge processor [21], respectively. All wire lengths are calculated based on the Manhattan-style wiring. Fig. 9 shows the Π5 *RC* model used to simulate wire delay while considering effects of crosstalk noise. As a common case, the wires are assumed to be in an intermediate layer, which incurs both ground and coupling capacitances depending on the metal wire dimensions (space, width, thickness, and length) and interlayer dielectric. The simulation is based on HSPICE using the device model from 32-nm CMOS predictable technology model (PTM) [22]. The wire dimensions used for simulation are derived from International Technology Roadmap for Semiconductors [23] reports. The metal resistance, ground, and coupling capacitance values (R_w, C_g, and C_c in Fig. 9) are calculated by PTM online interconnect tool. The center victim wire delay is measured from input A to output B including the buffer composed of two FO4 inverters. With the single buffered wire delay data (wire length from 0.1 to 2 mm^2 with 0.1-mm^2 interval), long wires can be optimally segmented, which provides a more realistic delay estimation. As shown in Table III, the delay data are based on the worst-case scenario where the signal on the center victim wire moves in the opposite direction of its aggressor neighbors N1 and N2.

Table III shows the interconnect wire delay and delay skew for square-shaped, house-shaped, and hexagonal-shaped processor tiles with the three sizes. For the 0.04-mm^2 small processor tile running at a 2-GHz clock frequency, the maximum interconnect wire delays for all processor shapes range from 5.7% to 6.9% of one clock cycle. For the 4-mm^2 medium-sized processor tile running at a 2-GHz clock frequency, the maximum interconnect wire delays for all three shapes range from 67% to 78% of one clock cycle. For the 36 mm^2 large processor tile running at 4 GHz, the maximum interconnect wire delay takes 4.0–4.7 clock cycles for the three shapes. Compared with square-shaped tile for all sizes, the maximum wire delay of the house-shaped tile increases by 15.9% to 21.3%, and the maximum wire delay of hexagonal-shaped tile increases by 2.8% to 8.7%. For a fully-synchronous system, special design effort is required to balance the interconnect wire delay skew to increase the maximum achievable frequency. As shown in Table III, the actual max link wire delay skew is smaller than the maximum link wire delay. For the 0.04-mm^2 small processor tile and the 4-mm^2 medium-sized processor tile running at a 2-GHz clock frequency, the maximum interconnect wire delay skews for all processor shapes take around 5% and 55% of one clock cycle, respectively. For the 36-mm^2 processor tile running at 4 GHz, the maximum interconnect wire delay skew is around 3.7 clock cycles on average. Compared with square-shaped tiles for all sizes, the maximum wire delay skew of the house-shaped tile increases by 8.7% to 29.7% and the maximum wire delay skew of hexagonal-shaped tile increases by 20.1% to 49.1%. The results suggest that the placement of switch fabrics for nonrectangular tiles has higher impact on link wire delay skew than the square tile.

IV. Application Mapping

A. Target Interconnect Architecture

A NoC is defined by its topology, switching policy, and routing algorithms. The proposed topologies can be used for dense on-chip network with either dynamic routers or static circuit switches. Fig. 10 shows the interprocessor communication in a typical 2-D mesh processor array using dynamic routers. As the diagram shows, the processors are connected by five-port routers each with five buffers and one 5 by 5 crossbar. The dynamic routers also include hardware logic to implement different routing algorithms. For the proposed non-2-D mesh topologies, special routers with more ports and routing algorithms are required. The routers with more than four neighbors, however, could be expensive in terms of area and power. The routing algorithms also need to be carefully designed to avoid deadlock and tolerate fault.

The static circuit-switch interconnection has smaller area, lower power dissipation, and lower complexity than dynamic router interconnection. In this paper, we assume processor tiles are connected with circuit switches which are suitable for applications with steady communication patterns. Fig. 11 shows the 4-4 Rect mesh array using circuit switches each with four nearest-neighbor interconnection links and two ports connecting to the processor core. In this case, each processor is capable of taking two inputs from the four directions and sending data to all four directions. The long distance

TABLE III
CHARACTERISTICS OF INTERCONNECT WIRES FOR PROCESSOR TILES WITH DIFFERENT SHAPES AND SIZES IN 32-nm CMOS

	Processor tile area = 0.04 mm^2			Processor tile area = 4.0 mm^2			Processor tile area = 36 mm^2		
	Square	House	Hex	Square	House	Hex	Square	House	Hex
Maximum link length (mm)	0.30	0.35	0.32	3.00	3.53	3.16	9.00	10.59	9.48
Maximum link length difference (mm)	0.20	0.26	0.29	2.00	2.58	2.93	6.00	7.74	8.79
Maximum link wire delay (ps)	28.6	34.7	31.1	338.0	391.7	347.6	1014.0	1186.0	1060.0
Maximum link wire delay skew (ps)	22.9	24.9	27.5	217.8	282.6	324.8	676.0	866.6	983.2

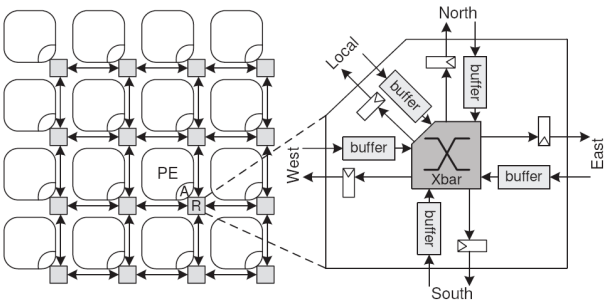


Fig. 10. 2-D mesh processor array using five-port routers where one port connects to the local processor core.

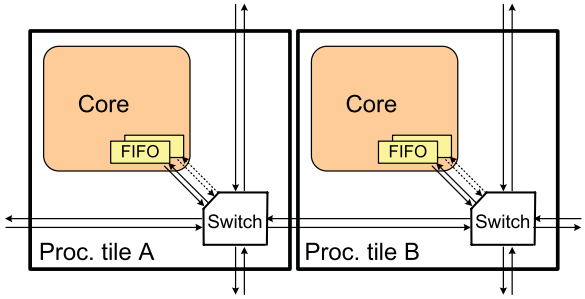


Fig. 11. Diagram of two processor tiles in the 4-4 Rect mesh processor array with four interconnection links and two input ports per tile.

communication is performed by software in the intermediate processors. The circuitry diagram of other topologies is similar, which differs in the number of links among neighboring processor tiles.

B. Two Benchmark Applications

Parallel programming on the discussed many-core systems with dense on-chip networks includes two main steps as follows: 1) partitioning the algorithms at a fine-grained level and 2) mapping the tasks to the nodes of the processor array and connecting the nodes with available links defined by the topology [24]. The two steps might be repeated iteratively for throughput optimization where we can identify the bottleneck task of the design and partition it even more until the throughput meets the requirement. To be specific, in the partitioning step, an estimate of task workload and required resources such as data and instruction memories are used to generate a fine-grained task graph where each task can be assigned to one processor node. Following the fine-grained partition, the mapping is conducted either manually or automatically by an automatic mapping tool. Application mapping is essentially an

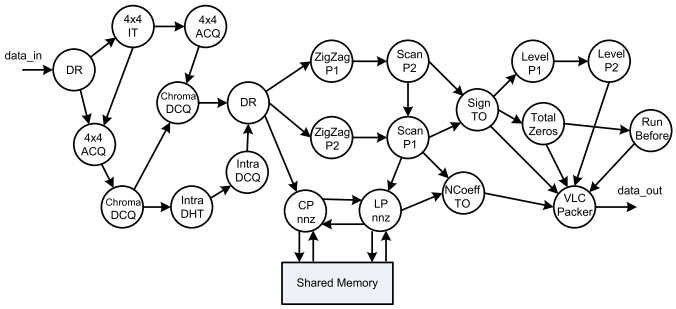


Fig. 12. Task graph of a 22-node H.264/AVC video residual encoder.

optimization problem, which can be formed as integer linear programming problem [25] and solved by heuristic algorithms. In this paper, we use a manual mapping method and the primary optimization target is to minimize area and maximize local communication.

To compare all discussed topologies, two complete applications including an H.264/AVC residual encoder and an 802.11a/g OFDM baseband receiver are first manually partitioned to meet the throughput requirement and ensure that each task can be handled by one processor. To be fair to compare all topologies, we chose not to partition tasks specifically for one topology and mapping the two applications onto all topologies is based on the same task graph.

Fig. 12 shows a 22-node task graph of an H.264/AVC residual baseline encoder composed of integer transform, quantization, and context-adaptive and variable length coding functions [24]. The encoder also requires a shared memory module as shown in the task graph.

Fig. 13 shows an example mapping of the H.264/AVC residual encoder capable of 1080p HDTV encoding at 30 f/s (frames per second) on the baseline 4-4 Rect mesh that uses 32 processors plus one shared memory. The 4-4 Rect mesh is inefficient in handling a complex application such as H.264/AVC encoding. A total of 10 processors are used for merging and forwarding data, which accounts for 31% of the total number of processors.

Fig. 14 shows a possible 25-processor mapping on the proposed 6-6 Hex topology. Compared with the design using a 4-4 Rect mesh, seven processors are saved, which accounts for a 22% reduction in the total number of processors.

Fig. 15 shows a 22-node task graph of a complete 802.11a/g WLAN baseband receiver which is computation intensive requiring two dedicated hardware accelerators: Viterbi decoder and FFT [26]. Fig. 16 shows a mapping of the 802.11a/g baseband receiver (54 Mb/s) on the baseline 4-4 Rect mesh that uses 32 processors plus the Viterbi decoder and FFT

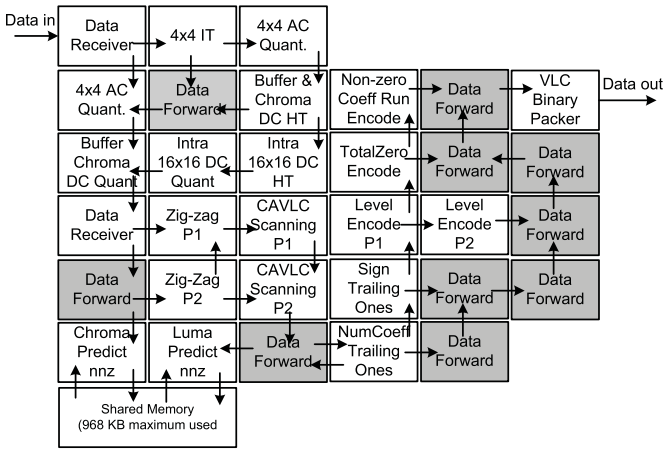


Fig. 13. H.264/AVC video residual encoder mapped on a processor array with 4-4 Rect mesh topology. The processors in gray are used for merging and forwarding data.

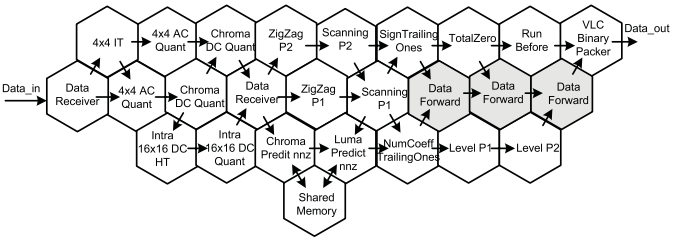


Fig. 14. H.264/AVC residual video encoder mapped on a processor array with 6-6 Hex topology.

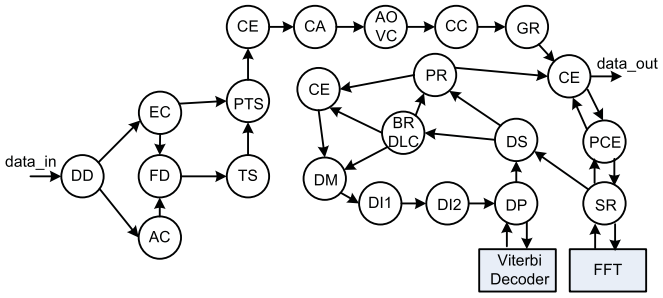


Fig. 15. Task graph of a 22-node 802.11a/g WLAN baseband receiver.

accelerators with 10 processors used for merging and forwarding data. Fig. 17 shows a mapping on the hexagonal-shaped tile architecture, which requires only 24 processors plus the Viterbi decoder and FFT accelerators—25% fewer processors than those used in the 4-4 Rect mesh mapping.

C. Application Mapping Results

We map the two applications to all proposed topologies. With the mapping task graphs, we can estimate the number of used processors and total communication link length, which provide initial indication on the final application area and performance after chip physical implementation.

1) Total Number of Used Processors: All six proposed topologies (type (b)–(g) in Fig. 2) are much more efficient than the *4-4 Rect* mesh (type (a) in Fig. 2), resulting in processor count reductions of 16% to 22% for the H.264

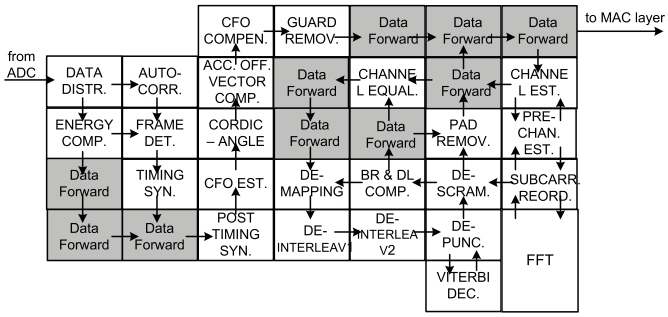


Fig. 16. 802.11a/g baseband receiver mapped on the processor array with baseline 4-4 Rect mesh topology.

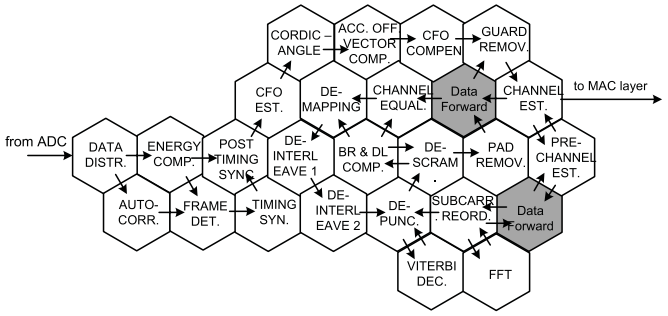


Fig. 17. 802.11a/g baseband receiver mapped on the processor array with 6-6 Hex topology.

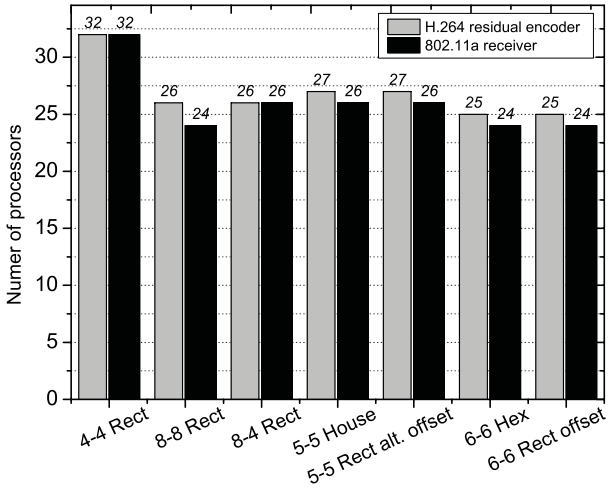


Fig. 18. Number of processors used for mapping two applications to the seven topologies (type (a)–(g) in Fig. 2).

encoder and 19% to 25% for the 802.11a/g baseband receiver, as shown in Fig. 18. The results are the same for the 5-5 House and 5-5 Rect Alt. Offset architecture because of essentially the same topology property. Similarly, the 6-6 Hex has the same result as the 6-6 Rect Offset architecture. The number of used processors of the 8-8 Rect and 8-4 Rect meshes is smaller than the 5-5 House and 5-5 Rect Alt. Offset topologies because of more communication links between processors. The two eight-neighbor Rect meshes, however, require a slightly larger number of processors than the two six-neighbor topologies, which yield the largest processor number reduction (24%) compared with the 4-4 Rect mesh.

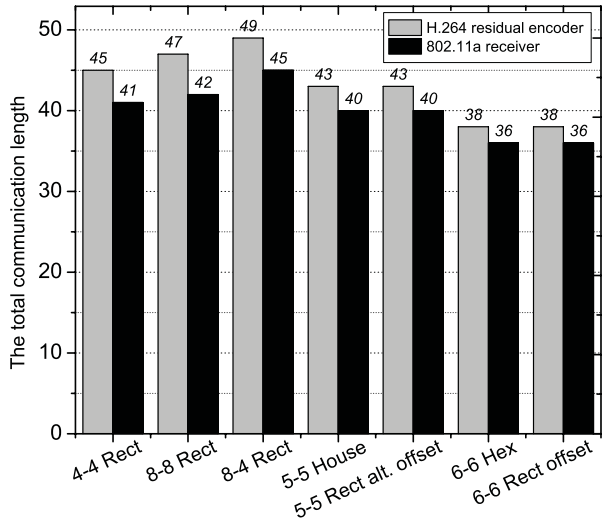


Fig. 19. Total communication length based on non-Manhattan-style wires (Euclidean link length) for the two applications mapped on the seven topologies (type (a)–(g) in Fig. 2). The link length is estimated based on the assumption the area of each processor tile is equal to one square unit of the length.

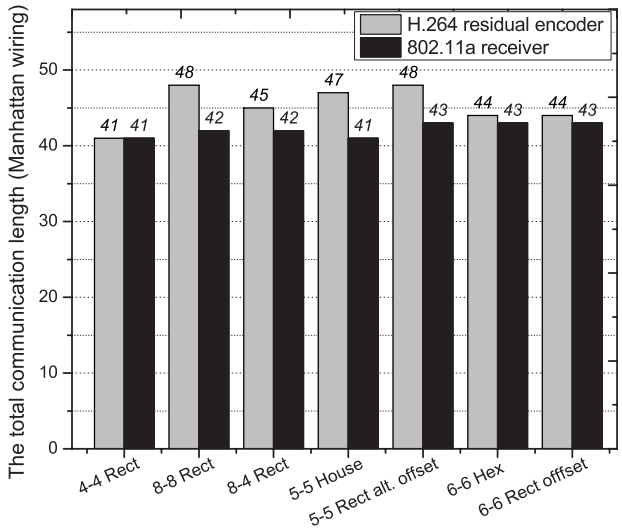


Fig. 20. Estimated total communication length based on Manhattan-style wires for the two applications mapped on the seven topologies (type (a)–(g) in Fig. 2). The link length is estimated based on the assumption the area of each processor tile is equal to one square unit of the length.

This is because the communication patterns of the two applications are mostly localized. Thus, topologies with more nearest-neighbor links yield more benefits than topologies with fewer nearest-neighbor links.

2) Total Communication Link Length: The total communication link length for the two applications can be calculated based on either Euclidean or Manhattan link length as shown in Table II and the application mapping diagrams.

Fig. 19 shows the total communication length based on non-Manhattan-style wires. The 8-8 Rect and 8-4 Rect have an average of 3% and 9% longer communication lengths than the 4-4 Rect mesh because they use more long communication links. The 6-6 Hex and 6-6 Rect Offset are the most efficient topologies, yielding the largest reduction (19%) in average total communication link length compared with the baseline 4-4 Rect mesh.

Fig. 20 shows the total communication link length based on Manhattan-style wires. All proposed topologies result in a slight increase of the total link length ranging from 1% to 5% compared with the baseline 4-4 Rect mesh. This small link length increase because of Manhattan wires has little influence in application performance, area, and power consumption, which will be demonstrated by the following physical implementation results.

V. Physical Design Methodology and Nonrectangular Processor Tile Design

A. Physical Design Methodology

For performance evaluation, a small processor with configurable circuit-switch interconnection is used for all physical designs. The processor contains a 16-bit datapath with a 40-bit accumulator and 560-byte instruction and 256-byte data memories. Each processor also contains a configurable clock oscillator (OSC) and two 128-byte FIFOs for data buffering and synchronization between two processors [10], [11]. Each interprocessor link is composed of 19 signals including a clock, 16-bit data, and two flow-control signals [27]. This processor is tailored for all topologies under test with a different number of neighboring interconnections ranging from four to eight. The internal switch fabrics are changed accordingly. The hardware overhead is minimal for four-neighbor, six-neighbor, and eight-neighbor processors with only 0.5%, 0.7%, and 2.0% hardware overhead based on synthesis results. The two eight-neighbor topologies add more complexity because processors communicate with two far-away processors via dedicated links, as shown in Fig. 2. To make CMP integration simpler, four additional sets of pins are inserted into the processor netlist after synthesis and are directly connected with bypass wires. This adds routing congestion in the corner for the topology shown in Fig. 2(b) and across the processor tile for the topology in Fig. 2(c).

All processors are implemented with a fully automated design flow spanning from RTL description to layout level verification with STMicroelectronics 65-nm CMOS technology. The processors are synthesized from Verilog with Synopsys Design Compiler and laid out with an automatic timing-driven physical design flow with Cadence SoC Encounter. Timing is optimized after each step of the physical design flow: floorplan, power planning, cell placement, clock tree insertion, and detailed routing. A configurable OSC is manually designed from standard cells and laid out separately.

B. Nonrectangular Processor Tiles Design and CMP Integration

The house-shaped and hexagonal-shaped tiles bring challenges for physical implementation. The first challenge to design the hexagonal processor is how to create a hexagonal shape at the floor plan stage. Rectangular placement and routing blockage in SoC encounter are used to create approximate triangle corner blockages with each rectangular blockage

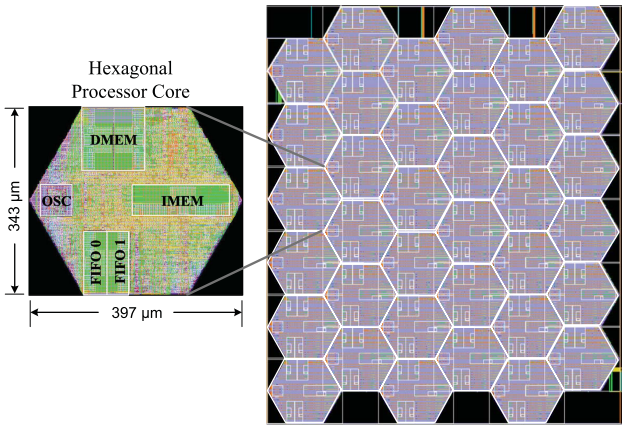


Fig. 21. DRC clean and LVS clean layouts of a hexagonal processor and a 6×6 multiprocessor array.

differing by one unit in width and height. All rectangular blockages are piled together to create an approximate triangle in the four corners of the rectangular floor plan.

A proper placement of pin positions can help to achieve efficient global routing and easy CMP integration. At the floor plan stage, four sets of pins are placed along the diagonal edge of the corner and two set of pins are placed in the horizontal top and bottom edges. As all macroblocks have rectangular shapes (OSC, IMEM, DMEM, and two FIFOs), this presents a challenge to place the macroblocks. In this design, the macroblocks are placed along the edge and the OSC and IMEM are placed in the left and right corners, respectively, as shown in Fig. 21.

Metals 6 and 7 are used to distribute power over the chip and the automatically created power stripes can stop at the created triangle edge in the corner. The power pins are created on the top and bottom horizontal edges. When integrating the hexagonal processor together, the power nets along the triangle edge can be connected automatically or manually by simple abutment.

Once a hexagonal processor tile is laid out, a script is used to generate the RTL files of the multiprocessor. The CMP array can be synthesized with empty processor tiles inside. Another script places the hexagonal tiles with the blockage area overlap with nearest-neighbor processors along the triangle edge of each hexagonal tile. SoC encounter can connect all pins automatically although there are overlaps between library exchange format files. The final GDSII files are read into Cadence ICFB for DRC.

Fig. 21 shows the final layout of a hexagonal-shaped processor tile and a 6 by 6 hexagonal-tiled multiprocessor array.

VI. Chip Implementation Results

For all discussed topologies, there is no long-distance intercommunication link across more than two processors and processors are pipelined in a way that the critical path is not in the interconnection links. Therefore, the maximum achievable frequency of an array is the same as an individual core, which is one of the key advantages of our proposed dense on-chip networks.

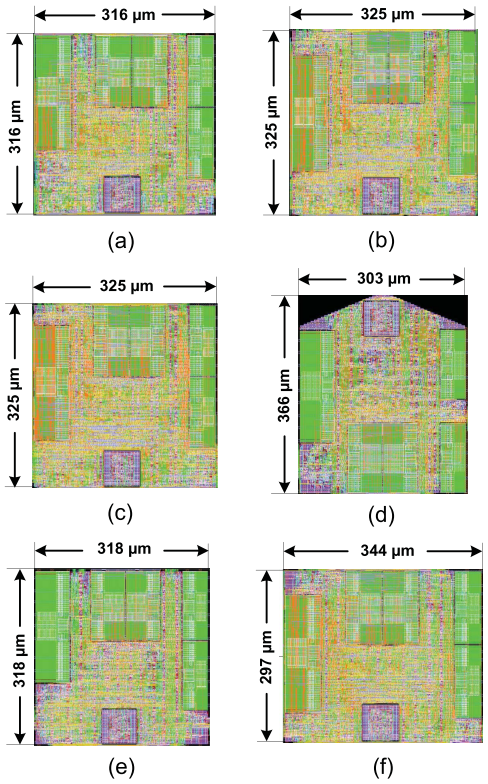


Fig. 22. Final DRC and LVS clean processor tile layouts corresponding to topologies. (a) 4-4 Rect. (b) 8-8 Rect. (c) 8-4 Rect. (d) 5-5 House. (e) 5-5 Rect Alt. Offset. (f) 6-6 Rect Offset. The hexagonal tile (6-6 Hex) shown in Fig. 21 is not included. All tiles have cell utilizations from 81% to 83%.

A. Processor Tile Implementation

Seven tile types are implemented from RTL to GDSII layout to get reliable estimates of how the topologies affect the system performance in nanoscale chip design. All floor plans use the same power distribution design and the I/O pins and macros are placed along edges reasonably depending on the topology. Rectangular hard macros might increase the area of house-shaped and hexagon-shaped tiles because there will be more corners where rectangular macros do not fit. We cannot draw a conclusion as this is affected by many design factors such as processor tile size, number of macros, and the CAD tool (floor plan and routing).

In standard cell design, a higher cell utilization ratio can both save area and increase system performance if the design is routable. To get a minimum chip area for all tiles, we start with a relatively large tile area, which results in a small cell utilization ratio. Then, the tiles are repeatedly laid out while maintaining the aspect ratio and reducing the area by 5% in each iteration with minor pin and macroblock position adjustments in the floor planning phase. Once a minimum area within 5% is reached, the area change is reduced to 2.5%. The layout tool is pushed until it is not able to generate an error-free GDSII layout for all tiles. Fig. 22 shows the final layouts of the other six processor tiles besides the hexagonal tile shown in Fig. 21. Our methodology results in high cell utilizations for all tiles ranging from 81% to 83%.

Fig. 23(a) shows the absolute area of the seven processors and Fig. 23(b) shows the area increments compared with the

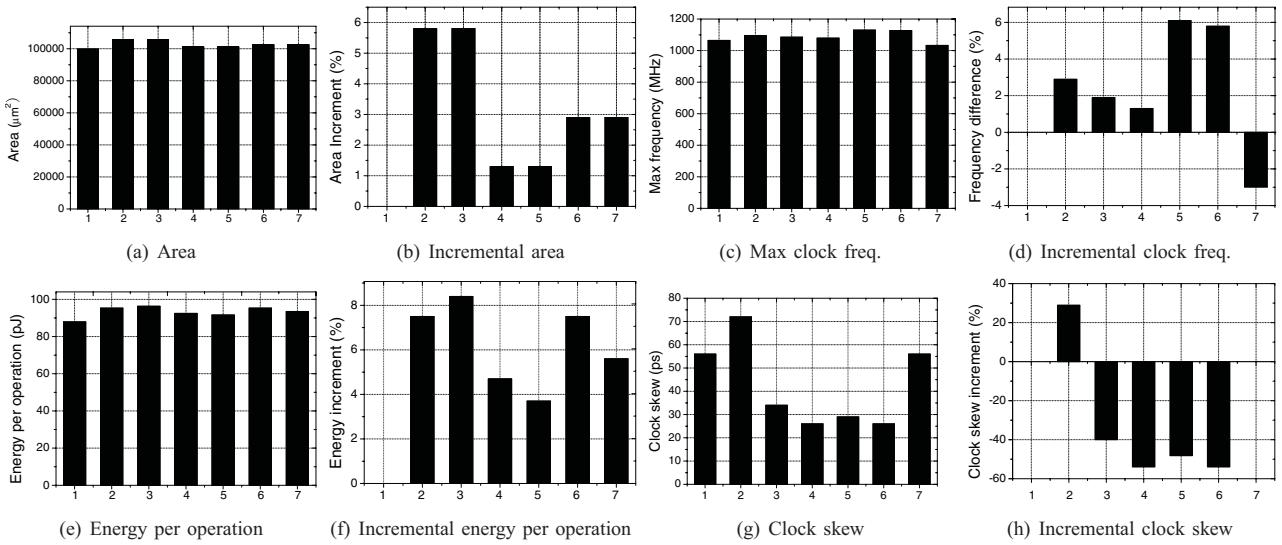


Fig. 23. Comparison of seven optimized processor tiles showing. (a) Absolute area. (b) Incremental area relative to the 4-4 Rect tile. (c) Absolute maximum clock frequency. (d) Incremental clock frequency relative to the 4-4 Rect tile. (e) Absolute energy per operation. (f) Incremental energy per operation relative to the 4-4 Rect tile. (g) Absolute clock skew. (h) Incremental clock skew relative to the 4-4 Rect tile. The processor types 1 to 7 correspond to the topologies shown in Fig. 2(a) to (g) which are 4-4 Rect, 8-8 Rect, 8-4 Rect, 5-5 House, 5-5 Rect Alt. Offset, 6-6 Hex, and 6-6 Rect Offset.

baseline 4-4 Rect tile which has the smallest area and the highest cell utilization of 83%. The hardware overhead of all processor tiles is very small. For the other six designs, the relative area increment is proportional to the number of nearest-neighbor connections. Compared with the baseline 4-4 Rect tile, an area increase of 1.3%, 2.9%, and 5.9% are required for the five-neighbor, six-neighbor, and eight-neighbor tile designs, respectively. All six designs have a cell utilization of 81%.

Fig. 23(c) shows the maximum clock frequency of all seven designs and Fig. 23 (d) shows the frequency increment relative to the baseline 4-4 Rect tile which can operate at a maximum of 1065 MHz at 1.3 V. Because of an increase of area, the two eight-neighbor mesh tiles can operate at 1.9% and 2.9% higher frequencies. The 5-5 Rect Alt. Offset and 6-6 Hex tiles have noticeably higher frequencies than other designs, which achieve a frequency increase of 6.1% and 5.8%, respectively. The 5-5 House tile has the same processor logic design and area as the 5-5 Rect Alt. Offset tile, while it has a frequency increment of only 1.5%. This is probably because the required aspect ratio for the house-shaped tile is not a good fit for this particular physical implementation. This can also explain why the 6-6 Rect Offset tile has the lowest frequency, a reduction of 3.0% in maximum frequency compared with the baseline 4-4 Rect tile.

Fig. 23(e) shows the energy per operation and Fig. 23(f) shows the incremental energy per operation compared with the 4-4 Rect tile. The energy is estimated based on a 20% activity factor for all internal nodes. All six proposed tiles have a higher energy per operation ranging from 3.7% to 8.4% because of the extra circuits for interconnections. Like the area increment, the average energy increments are proportional to the number of neighboring interconnections, as shown in Fig. 23(f).

Fig. 23(g) shows the worst-case clock skew for all seven processor tiles and Fig. 23(h) shows the clock skew increments compared with the 4-4 Rect tile. The 8-8 Rect tile shows a 29% higher clock skew probably because routing congestion in the corners affects the clock tree synthesis. The more circlelike shape helps the layout tool to generate a clock tree with smaller clock skew. As expected, the house-shaped and hexagonal-shaped tiles have the lowest clock skew with a reduction of 54% compared with the baseline 4-4 Rect tile.

B. Application Area and Power

Application area depends solely on the number of used processors and the processor tile sizes if processors are compactly tiled. Fig. 24 shows the normalized application area of two benchmark applications for all seven topologies. Compared with 4-4 Rect, the six proposed topologies reduce application area by 14%–22%. Corresponding to the largest reduction of the number of used processors, 6-6 Hex and 6-6 Rect Offset achieves the largest application area savings, a 22% reduction compared with the 4-4 Rect.

For applications mapped to the many-core processor array, the average power can be estimated by

$$P_{\text{Total}} = \sum_i P_{\text{Core},i} + \sum_i P_{\text{Comm},i} + P_{\text{other}} \tag{1}$$

where $P_{\text{Core},i}$ and $P_{\text{Comm},i}$ are the power consumption of processor core and communication circuits of the ith processor. P_{other} is the average power of other chip components such as memory modules or accelerators.

The power consumption of processor cores can be estimated based on their activity factors and measured power consumption when a processor core is 100% actively executing instructions. The two applications are simulated based on the 4-4 Rect topology to collect the computational processor activity factors and their output link activity factors. Because of a minimal workload change on computational processors across different topologies, the computational processor activity factors of all topologies are almost the same. Simulation

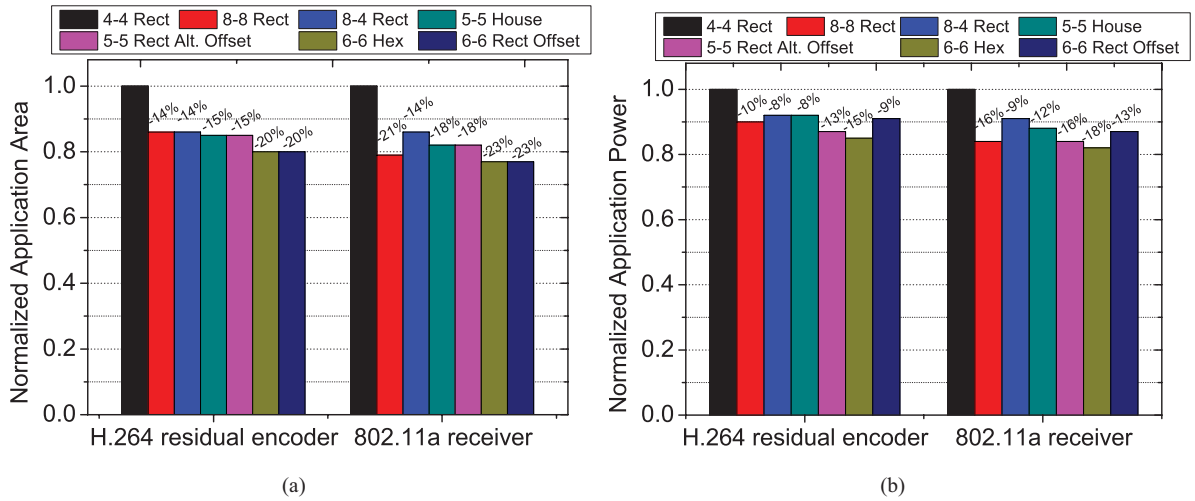


Fig. 24. Mapping results of the H.264 residual encoder capable of HD 1080p encoding at 30 f/s and 802.11a/g baseband receiver in 54-Mb/s mode. (a) Normalized application area. (b) Normalized power consumption.

results show that the average computational processor activity factors are 58% for H.264/AVC residual encoder and 49% for 802.11a/g baseband receiver, respectively. The activity factors of routing processors are estimated based on the number of input links and the corresponding link activity factors. The routing processor activity factors are 9.0% and 18.2% for H.264/AVC residual encoder and 802.11a/g baseband receiver, respectively.

The communication power of processor i can be estimated as follows:

$$P_{\text{Comm},i} = \sum_{j} \left(\delta_{ij} \cdot P_{\text{CommActive},L_j} + P_{\text{CommIdle},L_j}\right) \quad (2)$$

where δ_{ij} is the communication active percentage of link j; $P_{\text{CommActive},L_j}$ and P_{CommIdle,L_j} are the average power consumed by a link with a length L while the link is 100% active and idle. The communication link power is estimated based on simulation which is in a range 5%–10% of the processor power consumption. The link idle power (mainly leakage power) is nearly zero because of the simplicity of the communication circuits.

To meet the throughput requirement for the two mapped applications, processors need to run at 959 MHz at a supply voltage of 1.15 V for H.264 residual encoder and 594 MHz at a supply voltage of 0.92 V for the 802.11a/g baseband receiver. All processors run at the same clock frequency and supply voltages.

With the above equations as well as the processor power consumption numbers, application mapping diagrams, the required clock frequencies, and supply voltages for processors, Fig. 24(b) shows the normalized average power consumption of the H.264 residual encoder (encoding 1080p video at 30 f/s) and the 802.11a/g baseband receiver (54 Mb/s) for all seven topologies.

Compared with 4-4 Rect, the six proposed topologies reduce application power by 9%–17%. The 6-6 Hex achieves the largest average application power savings, a 17% reduction compared with 4-4 Rect. The 5-5 Rect Alt. Offset is the second most power-efficient topology, yielding 15% average power consumption compared with 4-4 Rect. Although the 6-6 Rect Offset has essentially the same topology property as 6-6 Hex, it reduces only 11% application power compared with 4-4 Rect.

VII. Conclusion

This paper presented seven low area overhead and low design complexity topologies other than the commonly used 2-D mesh for dense on-chip networks. The proposed topologies included two eight-neighbor meshes, two five-nearest-neighbor, and three six-nearest-neighbor topologies—three of which used a novel house-shaped and hexagonal-shaped tiles. The application mapping and chip implementation results showed the effectiveness of the interprocessor interconnect of all proposed topologies. Compared with 2-D mesh, the hexagonal-shaped six-nearest-neighbor topology reduced 22% application area and 17% average power consumption with a 2.9% area increase per processor tile. The rectangular-shaped six-nearest-neighbor topology provided the same interconnect architecture as the hexagonal-shaped tile. Despite being less power-efficient, its simpler physical design made it an attractive design alternative for many-core dense on-chip networks.

Acknowledgment

The authors would like to thank D. Truong for the offset-row topology idea and A. Tran for providing the 2-D mesh mapping of the 802.11a/g baseband receiver.

References

[1] M. Horowitz, R. Ho, and K. Mai, "The future of wires," *Proc. IEEE*, vol. 89, no. 4, pp. 490–504, Apr. 2001.

[2] H. Zhang, M. Wan, V. George, and J. Rabaey, "Interconnect architecture exploration for low-energy reconfigurable single-chip DSPs," in *Proc. IEEE Comput. Soc. Workshop VLSI*, Apr. 1999, pp. 2–8.

[3] P. P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh, "Effect of traffic localization on energy dissipation in NoC-based interconnect," in *Proc. IEEE Int. Symp. Circuits Syst.*, vol. 2. May 2005, pp. 1774–1777.

[4] E. Salminen, A. Kulala, and T. D. Hamalainen, "Survey of network-on-chip proposals," in *Proc. Open Core Protocol, Int. Partnership White Paper Conf.*, 2008, pp. 1–13.

[5] M. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, W. Lee, A. Saraf, N. Shnidman, V. Strumpen, S. Amarasinghe, and A. Agarwal, "A 16-issue multiple-program-counter microprocessor with point-to-point scalar operand network," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2003, pp. 170–171.

[6] Z. Yu, M. Meeuwsen, R. Apperson, O. Sattari, M. Lai, J. Webb, E. Work, T. Mohsenin, M. Singh, and B. M. Baas, "An asynchronous array of simple processors for DSP applications," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2006, pp. 428–429.

[7] Z. Yu, M. J. Meeuwsen, R. W. Apperson, O. Sattari, M. Lai, J. W. Webb, E. W. Work, D. Truong, T. Mohsenin, and B. M. Baas, "AsAP: An asynchronous array of simple processors," *IEEE J. Solid-State Circuits*, vol. 43, no. 3, pp. 695–705, Mar. 2008.

[8] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, and N. Borkar, "An 80-Tile 1.28TFLOPS network-on-chip in 65 nm CMOS," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2007, pp. 98–589.

[9] S. Bell, B. Edwards, J. Amann, R. Conlin, K. Joyce, V. Leung, J. MacKay, M. Reif, L. Bao, J. Brown, M. Mattina, C.-C. Miao, C. Ramey, D. Wentzlaff, W. Anderson, E. Berger, N. Fairbanks, D. Khan, F. Montenegro, J. Stickney, and J. Zook, "TILE64 processor: A 64-core SoC with mesh interconnect," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2008, pp. 88–89.

[10] D. Truong, W. Cheng, T. Mohsenin, Z. Yu, T. Jacobson, G. Landge, M. Meeuwsen, C. Watnik, P. Mejia, A. Tran, J. Webb, E. Work, Z. Xiao, and B. Baas, "A 167-processor 65 nm computational platform with per-processor dynamic supply voltage and dynamic clock frequency scaling," in *Proc. Symp. VLSI Circuits*, Jun. 2008, pp. 22–23.

[11] D. Truong, W. Cheng, T. Mohsenin, Z. Yu, A. Jacobson, G. Landge, M. Meeuwsen, A. Tran, Z. Xiao, E. Work, J. Webb, P. Mejia, and B. Baas, "A 167-processor computational platform in 65 nm CMOS," *IEEE J. Solid-State Circuits*, vol. 44, no. 4, pp. 1130–1144, Apr. 2009.

[12] J. Howard, S. Dighe, S. R. Vangal, G. Ruhl, N. Borkar, S. Jain, V. Erraguntla, M. Konow, M. Riepen, M. Gries, G. Droege, T. Lund-Larsen, S. Steibl, S. Borkar, V. K. De, and R. Van Der Wijngaart, "A 48-core IA-32 processor in 45 nm CMOS using on-die message-passing and dvfs for performance and power scaling," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 173–183, Jan. 2011.

[13] H. Chen, C.-K. Cheng, A. B. Kahng, I. I. Mandoiut, Q. Wang, and B. Yao, "The Y architecture for on-chip interconnect: Analysis and methodology," *IEEE Trans. Comput. Aided Design Integr. Circuits Syst.*, vol. 24, no. 4, pp. 588–599, Apr. 2005.

[14] F. Zhou, E. Y. Cheng, B. Yao, C.-K. Cheng, and R. Graham, "A hierarchical three-way interconnect architecture for hexagonal processors," in *Proc. Int. Workshop Syst. Level Interconnect Predict.*, Apr. 2003, pp. 133–139.

[15] K. G. Shin, "Harts: A distributed real-time architecture," *IEEE Comput.*, vol. 24, no. 5, pp. 25–35, May 1991.

[16] C. Decayeux and D. Seme, "3D hexagonal network: Modeling, topological properties, addressing scheme, and optimal routing algorithm," *IEEE Trans. Parallel Distrib. Syst.*, vol. 16, no. 9, pp. 875–884, Sep. 2005.

[17] J. Becker, F. Henrici, S. Trendelenburg, M. Ortmanns, and Y. Manoli, "A continuous-time hexagonal field-programmable analog array in 0.13 μm CMOS with 186 MHz GBW," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2008, pp. 70–71.

[18] A. D. Malony, "Regular processor arrays," in *Proc. 2nd Symp. Frontiers Massively Parallel Comput.*, Oct. 1988, pp. 499–502.

[19] A. Chariete, M. Bakhouya, J. Gaber, and M. Wack, "An approach for customizing on-chip interconnect architectures in SoC design," in *Proc. Int. Conf. High Perform. Comput. Simul.*, Jul. 2012, pp. 288–294.

[20] S. Agarwala, T. Anderson, A. Hill, M. D. Ales, R. Damodaran, P. Wiley, S. Mullinnix, J. Leach, A. Lell, M. Gill, A. Rajagopal, A. Chachad, M. Agarwala, J. Apostol, M. Krishnan, D. Bui, Q. An, N. S. Nagaraj, and T. Wolf, "A 600-MHz VLIW DSP," *IEEE J. Solid-State Circuits*, vol. 37, no. 11, pp. 1532–1544, Nov. 2002.

[21] M. Yuffe, E. Knoll, M. Mehalel, J. Shor, and T. Kurts, "A fully integrated multi-CPU, GPU and memory controller 32 nm processor," in *Proc. IEEE Int. Solid-State Circuits Conf.*, Feb. 2011, pp. 264–266.

[22] (2013). *Predictable Technology Model, Interconnect Section*, PTM. Tempe, AZ, USA [Online]. Available: http://www.eas.asu.edu/ptm/

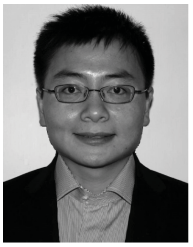
[23] (2010). *International Technology Roadmap for Semiconductors*, ITRS. Taiwan, South Korea [Online]. Available: http://www.itrs.net/reports.html

[24] Z. Xiao and B. Baas, "A 1080p H.264/AVC baseline residual encoder for a fine-grained many-core system," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 7, pp. 890–902, Jul. 2011.

[25] S. Tosun, O. Ozturk, and M. Ozen, "An ILP formulation for application mapping onto network-on-chips," in *Proc. Int. Conf. Appl. Inf. Commun. Technol.*, Oct. 2009, pp. 1–5.

[26] A. T. Tran, D. N. Truong, and B. M. Baas, "A complete real-time 802.11a baseband receiver implemented on an array of programmable processors," in *Proc. Asilomar Conf. Signals, Syst. Comput.*, Oct. 2008, pp. 165–170.

[27] Z. Yu and B. M. Baas, "A low-area multi-link interconnect architecture for GALS chip multiprocessors," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 5, pp. 750–762, May 2010.



Zhibin Xiao received the B.S. (Hons.) and M.S. degrees in information science and electrical engineering from Zhejiang University, Hangzhou, China, in 2003 and 2006, respectively, and the Ph.D. degree in electrical and computer engineering from the University of California, Davis, CA, USA, in 2012. His Ph.D. research focuses on high-performance many-core processor architecture, parallel video encoding, and scalable memory system design.

He participated in the development of a DSP processor for a dual core multimedia decoding chip with Zhejiang University. He is currently a Senior Hardware Engineer with Oracle America, Inc., where he is building co-processors to speed up various database applications for a high-end server chip.

Dr. Xiao served as a Technical Program Committee Member of the IEEE International Conference on Computer Design in 2013.

Bevan M. Baas (SM'99) received the B.S. degree in electronic engineering from California Polytechnic State University, San Luis Obispo, CA, USA, in 1987, and the M.S. and Ph.D. degrees in electrical engineering from Stanford University, Stanford, CA, in 1990 and 1999, respectively.

He was with Hewlett-Packard, Cupertino, CA, from 1987 to 1989, where he participated in the development of the processor for a high-end minicomputer. In 1999, he joined Atheros Communications, Santa Clara, CA, and served as a core member of the team which developed the first IEEE 802.11a (54 Mb/s, 5 GHz) Wi-Fi wireless local area network solution. In 2003, he joined the Department of Electrical and Computer Engineering, University of California, Davis, CA, where he is currently an Associate Professor. He leads projects in architecture, hardware, software tools, and applications for VLSI computation with an emphasis on digital signal processing workloads. In 2006, he was a Visiting Professor with Intel's Circuit Research Lab. His recent projects include the 36-processor asynchronous array of simple processors chip, applications, and tools, a second-generation 167-processor chip, low-density parity check decoders, fast Fourier transform processors, Viterbi decoders, and H.264 video codecs.

Dr. Baas was a fellow of the U.S. National Science Foundation from 1990 to 1993 and the NASA Graduate Student Researcher from 1993 to 1996. He was a recipient of the U.S. National Science Foundation CAREER Award in 2006 and the Most Promising Engineer Scientist Award by AISES in 2006. He received the Best Paper Award at ICCD in 2011 and the Best Paper Nominations at Asilomar in 2011 and BioCAS in 2010. He is an Associate Editor for the IEEE JOURNAL OF SOLID-STATE CIRCUITS and a Guest Editor of IEEE MICRO in 2012. He has been the Program Committee Co-Chair of HotChips in 2011, and a Program Committee Member of Hotchips from 2009 to 2010, ICCD from 2004 to 2005 and 2007 to 2009, ASYNC in 2010, and the ISSCC SRP Forum in 2012.