Design and Management of 3D Chip Multiprocessors Using Network-in-Memory

Feihui Li, Chrysostomos Nicopoulos, Thomas Richardson, Yuan Xie,
Vijaykrishnan Narayanan, Mahmut Kandemir
Dept. of CSE, The Pennsylvania State University
University Park, PA 16802, USA
{feli,nicopoul,trichard,yuanxie,vijay,kandemir}@cse.psu.edu

Abstract

Long interconnects are becoming an increasingly important problem from both power and performance perspectives. This motivates designers to adopt on-chip network-based communication infrastructures and three-dimensional (3D) designs where multiple device layers are stacked together. Considering the current trends towards increasing use of chip multiprocessing, it is timely to consider 3D chip multiprocessor design and memory networking issues, especially in the context of data management in large L2 caches. The overall goal of this paper is to study the challenges for L2 design and management in 3D chip multiprocessors. Our first contribution is to propose a router architecture and a topology design that makes use of a network architecture embedded into the L2 cache memory. Our second contribution is to demonstrate, through extensive experiments, that a 3D L2 memory architecture generates much better results than the conventional two-dimensional (2D) designs under different number of layers and vertical (inter-wafer) connections. In particular, our experiments show that a 3D architecture with no dynamic data migration generates better performance than a 2D architecture that employs data migration. This also helps reduce power consumption in L2 due to a reduced number of data movements.

1 Introduction and Motivation

The rapid scaling of technology into the deep sub-micron regime has been accompanied by a dramatic increase in transistor densities. The billion-transistor chip is now becoming a reality. At such integration levels, it is imperative to employ parallelism to effectively utilize the transistors [12]. For this purpose, today's microprocessors incorporate a multitude of sophisticated micro-architectural features, such as multiple instruction issue, dynamic scheduling, out-of-order execution, speculative execution, and dynamic branch prediction. However, in order to sustain performance growth, future superscalar microprocessors must rely on even more complex architectural innovations. Olukotun et al. [27] have shown that circuit limitations and limited instruction level parallelism will diminish the benefits afforded to the superscalar model by increased architectural complexity. Increased issue widths cause a quadratic increase in the size of issue queues and the complexity of register files. Furthermore, as the number of execution units increases, wiring and interconnection logic complexity begin to adversely affect performance. These issues have led to the advent of Chip Multiprocessors (CMP) as a viable alternative to the complex superscalar architecture. CMPs are simple, compact processing cores forming a decentralized micro-architecture which scales more efficiently with increased integration densities [27]. The Cell Processor from Sony, Toshiba and IBM (STI) [19], and the Sun UltraSPARC T1 (formerly codenamed Niagara) [22] signal the growing popularity of such systems.

The adoption of CMPs and other multi-core systems is expected to increase the size of both L2 and L3 caches in the foreseeable future. However, diminutive feature sizes exacerbate the impact of interconnect delay, making it a critical bottleneck in meeting the performance and power consumption budgets of a design [13, 1]. In the forthcoming 65 nm regime, up to 77% of the delay will be attributed to the interconnect [31]. Hence, while traditional architectures have assumed that each level in the memory hierarchy has a single, uniform access time, increases in interconnect delays will render access times in large caches dependent on the physical location of the requested cache line. That is, access times will be transformed into variable latencies based on the distance traversed along the chip.

The concept of Non-Uniform Cache Architectures (NUCA) [20] has been proposed based on the above observation. Instead of a large uniform monolithic L2 cache, the L2 space in NUCA is divided into multiple banks, which have different access latencies according to their location relative to the processor. These banks are connected through a mesh-based interconnection network. Cache lines are allowed to migrate within this network, for the purpose of placing more frequently-accessed data in the cache banks closer to the processor. Several recent proposals extend the NUCA concept to CMPs. An inherent problem of NUCA in CMP architectures is the management of data shared by

Proceedings of the 33rd International Symposium on Computer Architecture (ISCA'06)
0-7695-2608-X/06 $20.00 © 2006 IEEE
IEEE COMPUTER SOCIETY

multiple cores. Proposed solutions to this problem include data replication and data migration. Still, large access latencies and high power consumption stand as inherent problems for NUCA-based CMPs.

The introduction of three-dimensional (3D) circuits [9, 25] provides an opportunity to reduce wire lengths. Consequently, this technology can be useful in reducing the access latencies to the remote cache banks of a NUCA architecture. In this paper, we consider the design of a 3D topology for a NUCA that combines the benefits of network-on-chip and 3D technology to reduce L2 cache latencies in CMP-based systems. While network-on-chip and 3D cache designs have been studied in the past in different contexts, to our knowledge, this is the first in-depth study that integrates them. This paper provides new insights on network topology design for 3D NoCs and addresses issues related to processor placement across 3D layers and data management in L2, taking into account network traffic and thermal issues. We evaluate the proposed architecture using a novel simulation environment that includes the Simics multi-core simulator [24] and a cycle-accurate 3D network simulator. Our experiments with the SPEC OMP benchmark suite [33] indicate that the proposed 3D architecture significantly reduces L2 access latencies over two-dimensional (2D) NUCA implementations (around 17 cycles on average), leading to an IPC improvement of up to 37.1%. In addition, our experiments show that a 3D architecture with no dynamic data migration generates better performance than a 2D architecture that employs data migration. This also helps reduce power consumption in L2 due to reduced data movement.

This paper is organized as follows. The next section presents the background in nonuniform cache architectures, networks-on-chip, and 3D IC design. Section 3 discusses the details of the proposed 3D Network-in-Memory architecture. Section 4 explains our data management strategy in 3D L2 cache. The implementation details and an experimental evaluation of our approach are presented in Section 5. We conclude the paper in Section 6.

2 Background

In this section, we first review previous schemes that exploit the non-uniform access patterns of large L2 caches. We then introduce the Network-in-Memory and the three-dimensional architecture design issues.

2.1 NUCA Architectures

The issue of placement and location of data in large L2 caches has been identified by past research as one of the critical problems preventing us from extracting maximum performance from single- and multi-core machines. Recent proposals to large L2 design include decomposing L2 space into multiple, individually-addressable tiles (also called banks) whose access latencies depend on the distance between them and the processor that accesses them. Such architectures, usually known as Nonuniform Cache Architectures (NUCA), present unique challenges, as compared to uniform L2 architectures, in terms of data placement and management. Kim et al. [20] study cache line mapping and search strategies in the context of a NUCA architecture. Chishti et al. [5] propose NuRapid, a NUCA architecture that decouples tag placement from data placement. They later extended NuRapid to work within a CMP environment [6]. A unique contribution of this NuRapid architecture is that it employs a flexible data placement and replication-based management scheme to cut L2 access latencies. Zhang and Asanovic [39] propose a technique whereby copies of local primary cache victims are kept within the local L2 cache slices. Beckman and Wood [2] extend the NUCA concept to a multi-core setting by proposing a data migration scheme. Huh et al. [15] study the problem of how to partition an L2 NUCA to reduce interconnect traffic. Our L2 cache management policies benefit from both [2] and [6]; however, our L2 architecture is three-dimensional. Consequently, we tailored our data placement and migration policies considering multiple layers and the additional proximity provided by 3D. In addition, our results show that a 3D architecture can work very well even if we do not implement any data migration.

2.2 Network-in-Memory

To facilitate the aforementioned variation in access latencies, the cache can no longer be a monolithic structure, since the large size would be detrimental to access time. Instead, it should be divided into self-contained banks which can be individually addressed. This would create a continuum of access times based on the location of each bank relative to the CPU. However, such a scheme demands a very efficient interconnection network to minimize total access time. The solution might come from a related branch in chip design, namely Systems-On-Chip (SoC). SoCs incorporate several homogeneous or heterogeneous processing elements on a single die. The two dominant interconnects in the SoC research community have been buses and Networks-on-Chip (NoC) [8, 3]. Buses, while heavily used today, suffer from resource contention issues – as the number of nodes increases, performance degrades due to excessive conflicts. Hence, they are not considered appropriate for systems of more than about 10 nodes. To overcome these limitations, attention has shifted toward NoCs. NoCs packetize data and transmit it through an on-chip network [3] and, much like traditional macro networks, they are very scalable. Large NUCAs are expected to include tens of banks (as many as 256 have been seen in the literature), thus making NoCs an appropriate choice for the interconnect.

The most popular NoC topology in use today is the mesh, in which nodes are tiled in a Manhattan-like grid. Nodes are connected through a micro-network made up of a series of routers. Each node has a dedicated connection to one router, and each router has dedicated connections to

IEEE COMPUTER SOCIETY

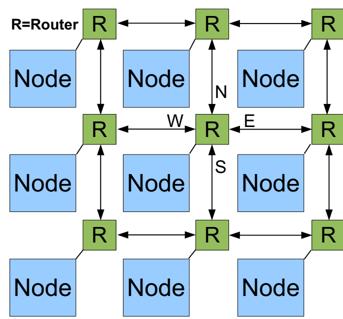

Figure 1. A typical NoC mesh.

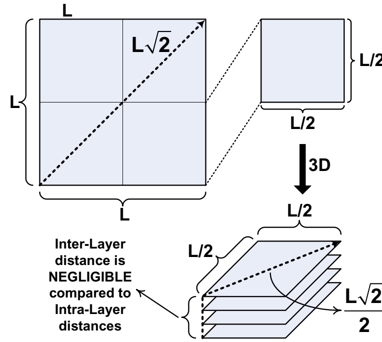

Figure 2. Wiring scales in length as the square root of the number of layers in three dimensions.

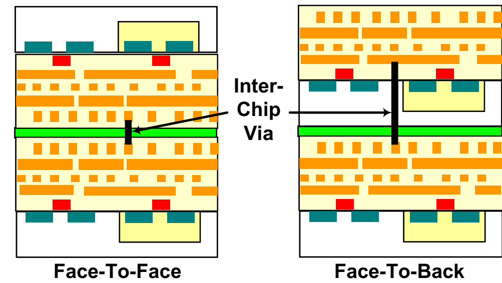

Figure 3. Face-to-Face and Face-to-Back Wafer-Bonding in 3D technology [11].

each of its four neighbors (North, South, East, and West). Each of the dedicated links between nodes and routers and between routers consists of two unidirectional links, each with a width which is equal to the flit size. Flits are the unit of transfer into which packets are broken for transmission. Figure 1 shows an NoC mesh topology, which is the basis for most current NUCA implementations in the literature. Previous work on NoCs focused on task mapping [14], energy management [36], and reliability [21]. In contrast to these studies, our work employs an NoC within a 3D L2 cache.

2.3 Three-Dimensional Design

The idea of having a large number of L2 banks on a two-dimensional plane poses several challenges, even if one employs an NoC. The large chip area occupied by the memory elements necessitates the use of several NoC routers. A cache access request destined for a distant bank would have to traverse a large number of NoC routers. Note that, even when using state-of-the-art single stage routers, the communication delay can be substantial if the number of routers in a flit's path is large. It is, therefore, imperative to limit the number of routers encountered by a flit between its source and destination (hop count) in order to optimize the NUCA's performance. The up-and-coming 3D chip design space is particularly amenable to reducing the Manhattan distance (and thus hop count) in large chips.

A three dimensional (3D) chip is a stack of multiple device layers with direct vertical interconnects tunneling through them [9, 25]. The benefits of 3D ICs include: 1) higher packing density due to the addition of a third dimension to the conventional two-dimensional layout, 2) higher performance due to reduced average interconnect length, and 3) lower interconnect power consumption due to the reduction in total wiring length [17]. Joyner et al. [17] have shown that three-dimensional architectures reduce wiring length by a factor of the square root of the number of layers used. For example, a 4-layer 3D NoC would have, on average, $\approx \sqrt{4} = 2$ times shorter wiring length, as illustrated in Figure 2.

There are currently various 3D technologies being explored in industry and academia, but the two most promising ones are Wafer-Bonding [9] and Multi-Layer Buried Structures (MLBS) [18]. Wafer-bonding technology processes each active device layer separately and then connects the layers in a single entity. For MLBS, the front-end processing is repeated on a single wafer to build multiple device layers, before the back-end process builds interconnects among the devices. A trait of critical importance in these technologies is the size of the 3D vias which connect neighboring layers together. In wafer-bonding, the size of the 3D vias is not expected to scale at the same rate as feature sizes [35]. MLBS, on the other hand, can provide vias which scale down with feature size. However, since MLBS is not compatible with current manufacturing processes, it is not as appealing as wafer bonding techniques [4, 16]. Furthermore, there are currently two primary wafer orientation schemes, Face-To-Face [4] and Face-To-Back [16, 11], as shown in Figure 3. While the former provides the greatest layer-to-layer via density, it is suitable for two-layer organizations, since additional layers would have to employ back-to-back placement using larger and longer vias. Face-To-Back, on the other hand, provides uniform scalability to an arbitrary number of layers, despite a reduced inter-layer via density. Hence, to provide scalability and easy manufacturability, we assume in this work the use of Face-To-Back Wafer-Bonding. One critical issue which has emerged in the design of 3D chips is the inter-layer via pitch, which dictates the density of layer-to-layer interconnections. Via pitches can range from 1x1 μm^2 to 10x10 μm^2 [9], depending on the technology and manufacturing process used. Very recently, IBM has managed to reduce the pitch to a state-of-the-art 0.2 x 0.2 μm^2 using Silicon-On-Insulator (SOI) technology [37]. However, despite the decreasing via sizes, it is the via pads (i.e., the via endpoints) which ultimately limit the via density. Currently, via pads do not scale at

IEEE COMPUTER SOCIETY

the same rate as the vias themselves. Compared to a wire pitch of 0.1 μm, inter-layer vias are significantly larger and cannot achieve the same wiring density as intra-layer interconnects. However, what vertical interconnects lack in terms of density, they compensate for by extremely small inter-wafer distances, ranging from 5 to 50 μm. Such distances are extremely small, providing rapid transfer times when traversing layers [29].

Researchers have so far focused on physical aspects, low-level process technologies, and developing automated design and placement tools [9, 11, 7]. Research at the architectural level has also surfaced [4, 35]. Specific to 3D memory design, [29, 38] have studied multi-bank uniform cache structures. In our work, we focus our attention on the design of a 3D NoC-based non-uniform L2 cache architecture.

3 A 3D Network-in-Memory Architecture

Our proposed architecture for multiprocessor systems with large shared L2 caches involves placement of CPUs on several layers of a 3D chip with the remaining space filled with L2 cache banks. Most 3D IC designs observed in the literature so far have not exceeded 5 layers, mostly due to manufacturability issues, thermal management, and cost. A detailed analysis of via pitch considerations and their impact on the number of inter-layer gateways follows in Section 3.1. As previously mentioned, the most valuable attribute of 3D chips is the very small distance between the layers. A distance on the order of tens of microns is negligible compared to the distance traveled between two network on-chip routers in 2D (1500 μm on average for a 64 KB cache bank implemented in 70 nm technology). This characteristic makes traveling in the vertical (inter-layer) direction very fast as compared to the horizontal (intra-layer).

One inter-layer interconnect option is to extend the NoC into three dimensions. This requires the addition of two more links (up and down) to each router. However, adding two extra links to an NoC router will increase its complexity (from 5 links to 7 links). This, in turn, will increase the blocking probability inside the router since there are more input links contending for an output link. Moreover, the NoC is, by nature, a multi-hop communication fabric, thus it would be unwise to place traditional NoC routers on the vertical path because the multi-hop delay and the delay of the router itself would overshadow the ultra fast propagation time.

It is not only desirable, but also feasible, to have single-hop communication amongst the layers because of the short distance between them. To that effect, we propose the use of dynamic Time-Division Multiple Access (dTDMA) buses as "Communication Pillars" between the wafers, as shown in Figure 4. These vertical bus pillars provide single-hop communication between any two layers, and can be interfaced to a traditional NoC router for intra-layer traversal using minimal hardware, as will be shown later. Furthermore, hybridization of the NoC router with the bus requires only one additional link (instead of two) on the NoC router. This is the case because the bus is a single entity for communicating both up and down. Due to technological limitations and router complexity issues (to be discussed in Section 3.1), not all NoC routers can include a vertical bus, but the ones that do form gateways to the other layers. Therefore, those routers connected to vertical buses have a slightly modified architecture, as explained in Section 3.2.

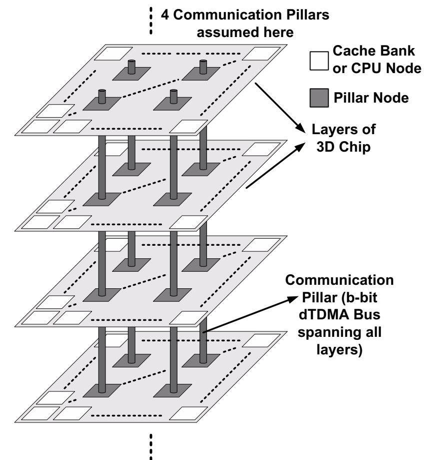

Figure 4. Proposed 3D Network-in-Memory architecture

3.1 The dTDMA Bus as a Communication Pillar

The dTDMA bus architecture [30] eliminates the transactional character commonly associated with buses, and instead employs a bus arbiter which dynamically grows and shrinks the number of timeslots to match the number of active clients. Single-hop communication and transaction-less arbitrations allow for low and predictable latencies. Dynamic allocation always produces the most efficient timeslot configuration, making the dTDMA bus nearly 100% bandwidth efficient. Each pillar node requires a compact transceiver module to interface with the bus, as shown in Figure 5.

The dTDMA bus interface (Figure 5) consists of a transmitter and a receiver connected to the bus through a tri-state driver. The tri-state drivers on each receiver and transmitter are controlled by independently programmed fully-tapped feedback shift registers. Details of its operation can be found in [30]. The total number of wires required by the control signals from the arbiter to each layer is $3n+log_2(n)$, for n layers. Because of its very small size, the dTDMA bus interface is a minimal addition to the NoC router.

The presence of a centralized arbiter is another reason why the number of vertical buses, or pillars, in the chip should be kept low. An arbiter is required for each pillar with control signals connecting to all layers, as shown

IEEE COMPUTER SOCIETY

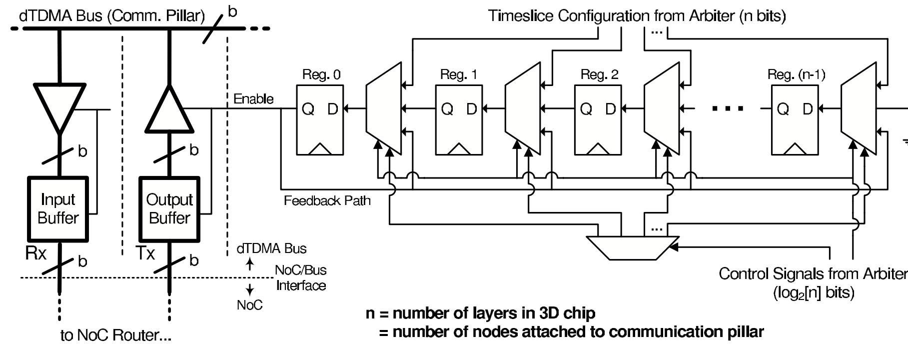

Figure 5. Transceiver module of a dTDMA bus.

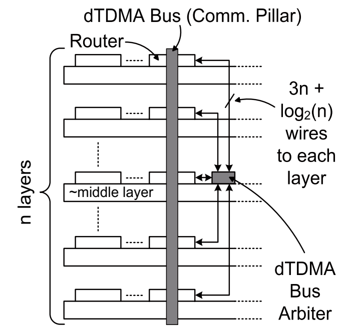

Figure 6. Side view of the 3D chip with the dTDMA bus (pillar).

Table 1. Area and power overhead of dTDMA bus.

Component	Power	Area
Generic NoC Router (5-port)	119.55 mW	0.3748 mm^2
dTDMA Bus Rx/Tx (2 per client)	97.39 μW	0.00036207 mm^2
dTDMA Bus Arbiter (1 per bus)	204.98 μW	0.00065480 mm^2

Table 2. Area overhead of inter-wafer wiring for different via pitch sizes.

Bus Width	Inter-Wafer Area (due to dTDMA Bus wiring)			
	10 μm	5 μm	1 μm	0.2 μm
128 bits(+42 control)	62500 μm^2	15625 μm^2	625 μm^2	25 μm^2

in Figure 6. The arbiter should be placed in the middle layer of the chip to keep wire distances as uniform as possible. Naturally, the number of control wires increases with the number of pillar nodes attached to the pillar, i.e., the number of layers present in the chip. The arbiter and all the other components of the dTDMA bus architecture have been implemented in Verilog HDL and synthesized using commercial 90 nm TSMC libraries. The area occupied by the arbiter and the transceivers is much smaller compared to the NoC router, thus fully justifying our decision to use this scheme as the vertical gateway between the layers. The area and power numbers of the dTDMA components and a generic 5-port (North, South, East, West, local node) NoC router (all synthesized in 90 nm technology) are shown in Table 1. Clearly, both the area and power overheads due to the addition of the dTDMA components are orders of magnitude smaller than the overall budget. Therefore, using the dTDMA bus as the vertical interconnect is of minimal area and power impact. A 7-port NoC router was considered and eliminated in the design search due to prohibitive contention issues, multi-hop communication in the vertical direction, and substantially increased area/power overhead due to an enlarged crossbar and more complicated switch arbiters. The dTDMA bus was observed to be better than an NoC for the vertical direction as long as the number of device layers was less than 9 (bus contention becomes an issue beyond that).

The length of vertical interconnect between two layers is assumed to be 10 μm. According to [10], the parasitics of inter-tier vias have a small effect on power and delay, because of their small length (i.e. low capacitance) and large cross-sectional area (i.e. low resistance).

The density of the inter-layer vias determines the number of pillars which can be employed. Table 2 illustrates the area occupied by a pillar consisting of 170 wires (128-bit bus + 3x14 control wires required in a 4-layer 3D SoC) for different via pitch sizes. In Face-To-Back 3D implementations (Figure 3), the pillars must pass through the active device layer [29], implying that the area occupied by the pillar translates into wasted device area. This is the reason why the number of inter-layer connections must be kept to a minimum. However, as via density increases, the area occupied by the pillars becomes smaller, and, at the state-of-the-art via pitch of 0.2 μm, becomes negligible compared to the area occupied by the NoC router (see Table 1 and Table 2). However, as previously mentioned, via densities are still limited by via pad sizes, which are not scaling as fast as the actual via sizes. As shown in Table 2, even at a pitch of 5 μm, a pillar induces an area overhead of around 4% to the generic 5-port NoC router, which is not overwhelming. These results indicate that, for the purposes of our 3D architecture, adding extra dTDMA bus pillars is feasible.

Via density, however, is not the only factor limiting the number of pillars. Router complexity also plays a key role. As previously mentioned, adding an extra vertical link (dTDMA bus) to an NoC router will increase the number of ports from 5 to 6, and since contention probability within each router is directly proportional to the number of competing ports, an increase in the number of ports increases the contention probability. This, in turn, will increase congestion within the router, since more flits will be arbitrating for access to the router's crossbar. Thus, arbitrarily adding vertical pillars to the NoC routers adversely affects the performance of each pillar router. Hence, the number of high-contention routers (pillar routers) in the network increases,

IEEE COMPUTER SOCIETY

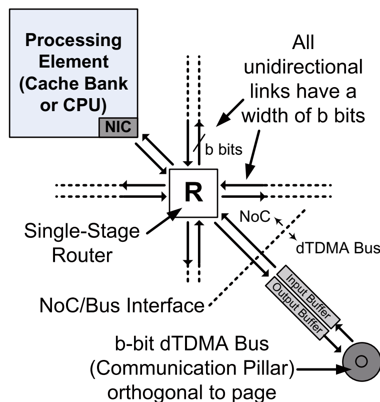

Figure 7. A high-level overview of the modified router of the pillar nodes.

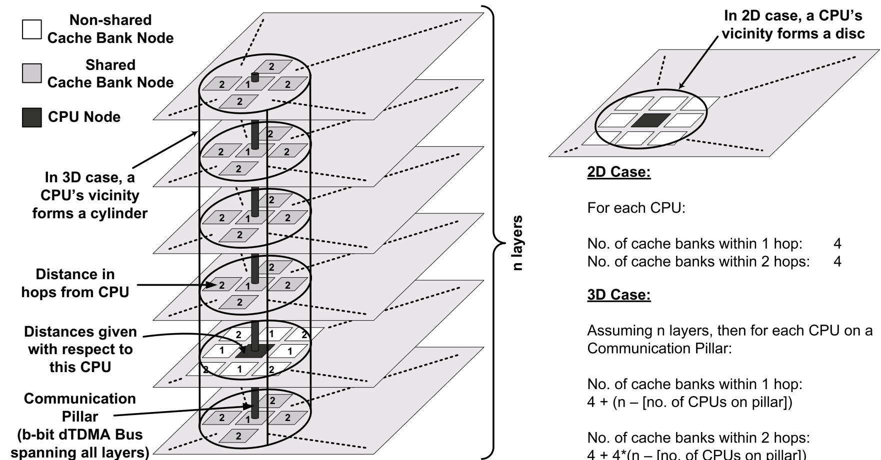

Figure 8. A CPU has more cache banks in its vicinity in the 3D architecture.

thereby increasing the latency of both intra-layer and inter-layer communication.

On the other hand, there is a minimum acceptable number of pillars. In this work, we place each CPU on its own pillar. If multiple CPUs were allowed to share the same pillar, there would be fewer pillars, but such an organization would give rise to other issues, as described in Section 3.3.

3.2 NoC Router Architecture

A generic NoC router consists of four major components: the routing unit (RT), the virtual channel allocation unit (VA), the switch allocation unit (SA), and the crossbar (XBAR). In the mesh topology, each router has five physical channels (PC): North, South, East, and West, and one for the connection with the local processing element (CPU or cache bank). Each physical unit has a number of virtual channels (VC) associated with it. These are first-in-first-out (FIFO) buffers which hold flits from different pending messages. In our implementation, we used 3 VCs per PC, each 1 message deep. Each message was chosen to be 4 flits long. The width of the router links was chosen to be 128 bits. Consequently, a 64B cache line can fit in a packet (i.e., 4 flits/packet x 128 bits/flit = 512 bits/packet = 64 B/packet).

The most basic router implementations are 4-stage ones, i.e., they require a clock cycle for each component within the router. In our L2 architecture, low network latency is of utmost importance, thereby necessitating a faster router. Lower-latency router architectures have been proposed which parallelize the RT, VA and SA using a method known as speculative allocation [28]. This method predicts the winner of the VA stage and performs SA based on that. Moreover, a method known as look-ahead routing can also be used to perform routing one step ahead (perform the routing of node i+1 at node i). These two modifications can significantly improve the performance of the router. Two stage, and even single-stage [26], routers are now possible which parallelize the various stages of operation. In our proposed architecture, we use a single-stage router to minimize latency.

Routers connected to pillar nodes are different, as an interface between the dTDMA pillar and the NoC router must be provided to enable seamless integration of the vertical links with the 2D network within the layers. The modified router is shown in Figure 7. An extra physical channel (PC) is added to the router, which corresponds to the vertical link. The extra PC has its own dedicated buffers, and is indistinguishable from the other links to the router operation. The router only sees an additional physical channel.

3.3 CPU Placement

The dTDMA pillars provide rapid communication between layers of the chip. We have a dedicated pillar associated with each processor to provide fast inter-layer access, as shown in Figure 4. Such a configuration gives each processor instant access to the pillar, additionally providing them with rapid access to all cache banks that are adjacent to the pillar. By placing each processor directly on a pillar, its memory locality (the number of banks with low access latency) is increased in the vertical direction (potentially both above and below), in addition to the pre-existing locality in the 2D plane. This is illustrated in Figure 8. Such an increase in the number of cache banks with low access latency can significantly improve the performance of applications. The relative sizing of L2 cache banks and CPU+L1 cache, as shown in Figure 8, is meant to be illustrative. Our placement approach works even when a CPU+L1 cache span the size of multiple L2 cache banks.

Stacking CPUs directly on top of each other would give

IEEE COMPUTER SOCIETY

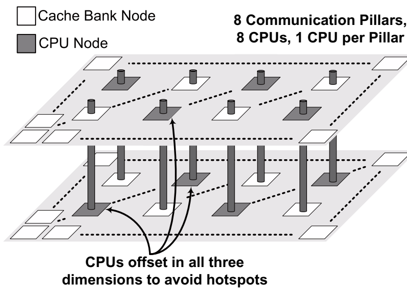

Figure 9. Hotspots can be avoided by offsetting CPUs in all three dimensions.

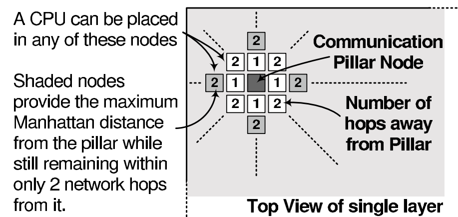

Figure 10. Placement pattern of CPUs around the pillars.

rise to thermal issues. Increased temperatures due to layer stacking is a major challenge in 3D design [4], and is often a major determining factor in component placement. Since the CPUs are expected to consume the overwhelming majority of power (they are constantly active, unlike the cache banks), it would be thermally-unwise to stack any two or more processors in the same vertical plane. Furthermore, stacking processors directly on top of each other on the same pillar would affect the performance of the network as well, as it would create high congestion on the pillar. Processors are the elements which generate most of the L2 traffic (there is also some traffic generated by the migration algorithm, as explained in Section 4.2); therefore, forcing them to share a single link would create excessive traffic. Our simulations in later sections will validate this argument. To avoid thermal and congestion problems, CPUs can be offset in all three dimensions (maximal offsetting), as shown in Figure 9.

If, however, the manufacturing technology provides only low via densities, the designer might be forced to use fewer pillars than the number of CPU cores to minimize the wasted device-layer area. In such cases, multiple CPUs would have to share a single pillar, warranting a careful CPU placement methodology, to ensure maximum performance with minimum thermal side-effects. The best way to achieve this is to offset the processors at various distances from the pillar, while keeping them within the direct vicinity. Moving the processors far away from the pillars would be detrimental to performance, since changing layers would inflict lengthy 2D traversal times to and from the pillar. In our approach, the CPUs are placed according to the pattern illustrated in Figure 10. The processors are placed at most two hops away from a pillar to minimize the adverse effect on performance.

To perform the placement of the CPUs based on the restrictions of Figure 10, a simple algorithm was developed to achieve uniform offset of all CPUs in both the vertical and horizontal directions for the configurations studied. The algorithm, shown in Algorithm 1, assumes placement of 2 or 4 CPUs per pillar per chip layer. The parameter k is the offset distance from a pillar in number of network hops. In our proposed implementation, k was chosen to be 1. If thermal issues require more mitigation, then k can be increased at the expense of performance. Parameter c is the number of CPUs assigned to each pillar on each layer. Assigning more than 4 CPUs per pillar per layer is not desirable, since it will increase bus contention dramatically, thus degrading network performance. The location of the pillars is assumed to be predetermined by the designer, and is given as a constant to the algorithm. The pillars need to be placed as far apart from each other as possible within the layer to avoid the creation of network congested areas. On the other hand, the pillars should not be placed on the edges because such placement would limit the number of cache banks which are in the vicinity of the pillar.

The placement pattern spans four layers, beyond which it is repeated. This is done because 1) increasing the factor of k beyond 2 moves the CPU too far from the pillar, and 2) thermal effects reduce as inter-layer distance between processors increases. Note that the placement is scalable to any number of layers.

To visualize how the algorithm works, Figure 11 shows an example for four CPUs per pillar per layer. The key observation is that this placement methodology provides a systematic and scalable way to place the CPUs in a thermal aware fashion.

To validate the CPU placement methodology proposed above, we simulated the thermal profile of several configurations using HS3d, a 3D thermal estimation tool [23]. HS3d provides estimates for peak, minimum and average temperatures of a chip, based on the power consumption and location of all architectural components. The tool also provides a detailed steady-state thermal profile for the entire floorplan, allowing us to identify hotspots. The power consumption of the cache banks was extracted from Cacti 3.2 [32]. Regarding the CPU cores, we assumed a simple, single-issue core, similar to recent trends in industry (STI Cell Processor, Sun UltraSPARC T1) [22, 19]. Sun's UltraSparc T1 (formerly codenamed Niagara) employs 8 processing cores with peak power consumption of 79 W [34]. Using this information as a guideline for future CMPs, we approximate the power consumption per core to be 8W in our thermal simulation (assuming that the rest of the power is consumed in peripheral circuits and L2 cache). Since our experiments focus on the relative trends between the 2D and 3D topologies, the results will still hold even if this absolute power consumption is not accurate.

IEEE COMPUTER SOCIETY

Algorithm 1: CPU placement algorithm for more than 1 CPU/pillar/layer

```
c: number of CPUs per pillar
k: distance offset from pillar in terms of the number of network hops
for i ∈ Pillars {
    for l ∈ Layers {
        //x_i,y_i of each pillar assumed predetermined
        pillar node i = (x_i, y_i, l)
        switch l modulo 4 {
        case 0: if c == 2 then
                    CPU's at (x_i ± k, y_i, l)
                else if c == 4 then
                    CPU's at (x_i ± 2k, y_i, l) and (x_i, y_i ± 2k, l)
        case 1: if c == 2 then
                    CPU's at (x_i, y_i ± k, l)
                else if c == 4 then
                    CPU's at (x_i ± k, y_i ± k, l)
        case 2: if c == 2 then
                    CPU's at (x_i ± 2k, y_i, l)
                else if c == 4 then
                    CPU's at (x_i ± k, y_i, l) and (x_i, y_i ± k, l)
        case 3: if c == 2 then
                    CPU's at (x_i, y_i ± 2k, l)
                else if c == 4 then
                    CPU's at (x_i ± 2k, y_i ± 2k, l)
        }
    }
}
```

Table 3. Temperature profile of various configurations, where L is the number of layers and k is the offset factor in Algorithm 1.

Configuration	Peak Temp ^oC	Avg Temp ^oC	Min Temp ^oC
2D, maximal offset	111.05	53.96	46.77
3D-2L, optimal offset	119.05	63.94	49.21
3D-2L, offset k=2	125.02	63.94	49.59
3D-2L, offset k=1	135.24	63.94	49.52
3D-2L, CPU stacking	173.38	63.94	50.73
3D-4L, optimal offset	158.67	86.62	64.79
3D-4L, CPU stacking	287.12	86.62	58.51

The results of the thermal simulations on a system with 256 64KB L2 cache banks (16MB) and 8 CPU cores are shown in Table 3. The L2 cache banks are clock-gated when not in use. As expected, moving from a 2D layout to a 3D layout causes the average temperature of the chip to increase. However, of critical importance is the avoidance of hotspots, i.e., places where the peak temperature is extremely high. Hotspots can substantially degrade performance and adversely affect the lifetime of the chip. Hotspots can be avoided through careful offsetting of the processor cores. Offsetting the cores in all three dimensions provides the best results, causing an increase in peak temperature of only 8°C when using two layers, as compared to the 2D case. Assuming a single CPU per pillar (row 2 of Table 3), CPUs can be optimally offset in all three dimensions, as in Figure 9. However, when processors need to share pillars, then we use the offsetting technique of Algorithm 1. As shown in Table 3, increasing the offset factor, k, can reduce peak temperature by 10°C. As predicted, stacking CPUs in both 2-layer and 4-layer configurations is detrimental to peak temperature (i.e., leads to hotspot creation).

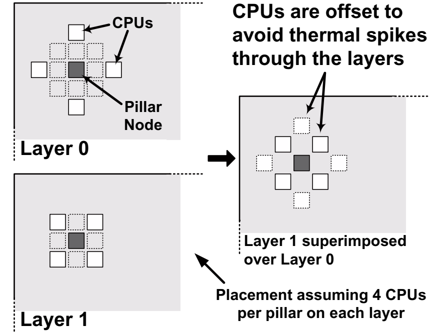

Figure 11. An example of our CPU placement algorithm.

4 3D L2 Cache Management

In this section, we present our organization of processor and L2 cache banks, and then detail our L2 cache management policies.

4.1 Processors and L2 Cache Organization

Figure 12 illustrates the organization of the processors and L2 caches in our design. Similar to CMP-DNUCA [2], we separate cache banks into multiple clusters. Each cluster contains a set of cache banks and a separate tag array for all the cache lines within the cluster. Some clusters have processors placed in the middle of them, while others do not. All the banks in a cluster are connected through a network-on-chip for data communication, while the tag array has a direct connection to the local processor in the cluster. Note that even though it is not explicitly shown in Figure 12, each processor has its own private L1 cache and an associated tag array for L2 cache banks within its local cluster. For a cluster without a local processor, the tag array is connected to a customized logic block which is responsible for receiving a cache line request, searching the tag array and forwarding the request to the target cache bank. This organization of processors and caches can be scaled by changing the size and/or number of the clusters.

4.2 Cache Management Policies

Based on the organization of processors and caches given in the previous subsection, we developed our cache management policies, consisting of a cache line search policy, a cache placement and replacement policy, and a cache line migration policy, all of which are detailed in the following subsections.

4.2.1 Search Policy

Our cache line search strategy is a two-step process. In the first step, the processor searches the local tag array in the cluster to which it belongs and also sends requests to

IEEE COMPUTER SOCIETY

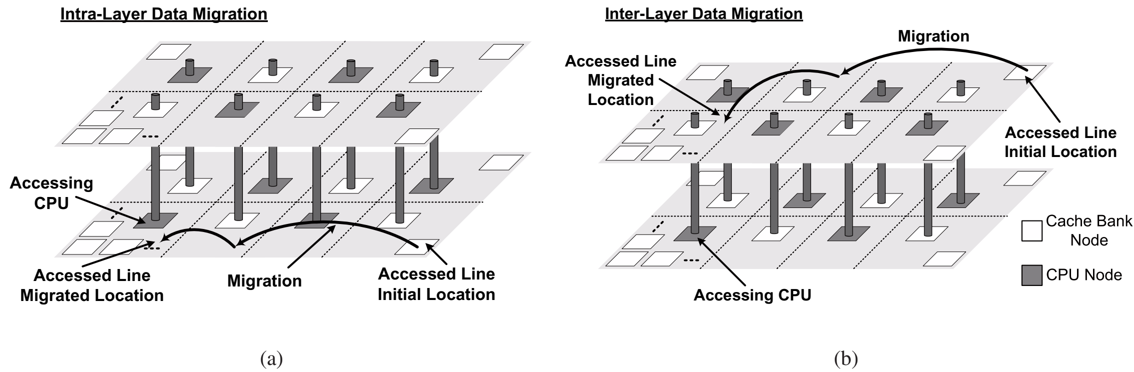

Figure 12. Intra-layer and inter-layer data migration in the 3D L2 architecture. Dotted lines denote clusters.

search the tag array of its neighboring clusters. All the vertically neighboring clusters receive the tag that is broadcast through the pillar. If the cache line is not found in either of these places, then the processor multicasts the requests to the remaining clusters. If the tag match fails in all the clusters, then it is considered as an L2 miss. On a tag match in any of the clusters, the corresponding data is routed to the requesting processor through the network on chip.

4.2.2 Placement and Replacement Policy

We use cache placement and replacement policies similar to those of CMP-DNUCA [2]. Initially a cache line is placed according to the low-order bits of its cache tag, that is, these bits determine the cluster in which the cache line will be placed initially. The low-order bits of the cache index indicate the bank in the cluster into which the cache line will be placed. The remaining bits of the cache index determine the location in the cache bank. The tag entry of the cluster is also updated when the cache line is placed. The placement policy can only be used to determine the initial location of a cache line as when cache lines start migrating, the lower order bits of the cache tag can no longer indicate the cluster location. Finally, we use a pseudo-LRU replacement policy to evict a cache line to service a cache miss.

4.2.3 Cache Line Migration Policy

Similar to prior approaches, our strategy attempts to migrate data closer to the accessing processor. However, our policy is tailored to the 3D architecture and migrations are treated differently based on whether the accessed data lies in the same or different layer as the accessing processor. For data located within the same layer, the data is migrated gradually to a cluster closer to the accessing processor. When moving the cache lines to a closer cluster, we skip clusters that have processors (other than the accessing processor) placed in them since we do not want to affect their local L2 access patterns and get the cache lines to the next closest cluster without a processor. Eventually, if the data is accessed repeatedly by only a single processor, it migrates to the local cluster of the processor. Figure 12(a) illustrates this intra-layer data migration.

For data located in a different layer, the data is migrated gradually closer to the pillar closest to the accessing processor (see Figure 12(b)). Since clusters accessible through the vertical pillar communications are considered to be in local vicinity, we never migrate the data across the layers. This decision has the benefit of reducing the frequency of cache line migrations, which in turn reduces power consumption.

To avoid false misses (misses caused by searches for data in the process of migration), we employ a lazy migration mechanism as in CMP-DNUCA [2].

5 Experimental Evaluation

5.1 Methodology

We simulated the 3D CMP architecture by using Simics [24] interfaced with a 3D NoC simulator. A full-system simulation of an 8-processor CMP architecture running Solaris 9 was performed. Each processor uses in-order issue and executes the SPARC ISA. The processors have private L1 caches and share a large L2 cache. The default configuration parameters for processors, memories and Network-in-Memory are given in Table 4. Some of the parameters in this table are modified for studying different configurations. The shown cache bank and tag array access latencies are extracted using Cacti 3.2 [32].

To model the latency of the three-dimensional, hybrid NoC/bus interconnect, we developed a cycle-accurate simulator in C, based on an existing 2D NoC simulator [21]. For this work, the 2D simulator was extended to three dimensions, and the dTDMA bus was integrated as the vertical communication channel. The 3D NoC simulator produces, as output, the communication latency for cache access.

In our cache model, private L1 caches of different processors are maintained coherent by implementing a dis-

IEEE COMPUTER SOCIETY

Table 4. Default system configuration parameters (L2 cache is organized as 16 clusters of size 16x64KB).

Processor Parameters	
Number of Processors	8
Issue Width	1
Memory Parameters	
L1 (split I/D)	64KB, 2-way, 64B line, 3-cycle, write-through
L2 (unified)	16MB (256x64KB), 16-way, 64B line, 5-cycle bank access
Tag Array (per cluster)	24KB, 4-cycle access
Memory	4GB, 260 cycle latency
Network Parameters	
Number of Layers	2
Number of Pillars	8
Routing Scheme	Dimension-Order
Switching Scheme	Wormhole
Flit Size	128 bits
Router Latency	1 cycle

tributed directory-based protocol. Each processor has a directory tracking the states of the cache lines within its L1 cache. L1 access events (such as read misses) cause state transitions and updates to directories, based on the MSI protocol. The traffic due to L1 cache coherence is taken into account in our simulation.

We simulated nine SPEC OMP benchmarks [33] with our simulation platform. These benchmarks are listed in Table 5[1]. For each benchmark, we marked an initialization phase in the source code. The cache model is not simulated until this initialization completes. This is reflected as the fastforward cycles for each benchmark shown in the second row of Table 5. After that, each application runs 500 million cycles for warming up the L2 caches. We then collected statistics for the next 2 billion cycles following the cache warm-up period. The third row in Table 5 gives the total number of L2 cache accesses (including data read, data write, and instruction fetch) within the sampling period for each benchmark. We see that the benchmarks mgrid, swim and wupwise exhibit many more L2 accesses than the others, as a result of higher L1 miss rates.

5.2 Results

We first introduce the schemes compared in our experiments. We refer to the scheme with perfect search from [2] as CMP-DNUCA. We name our 2D and 3D schemes as CMP-DNUCA-2D and CMP-DNUCA-3D, respectively. Note that our 2D scheme is just a special case of our 3D scheme discussed in the paper, with a single layer. Both of these schemes employ cache line migration. To isolate the benefits due to 3D technology, we also implemented our 3D scheme without cache line migration, which is called CMP-SNUCA-3D.

Our first set of results give the average L2 hit latency numbers under different schemes. The results are presented in Figure 13. We observe that our 2D scheme (CMP-DNUCA-2D) generates competitive results with the prior

[1] We could not run the remaining two SPEC OMP benchmarks through Simics due to a memory leak problem.

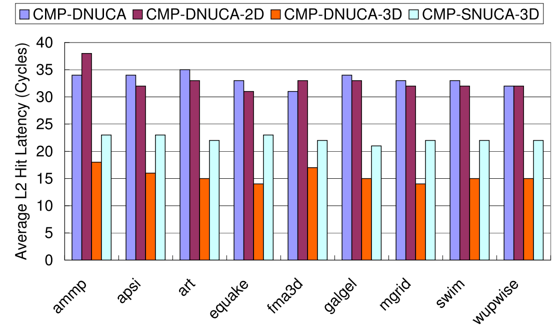

Figure 13. Average L2 hit latency values under different schemes.

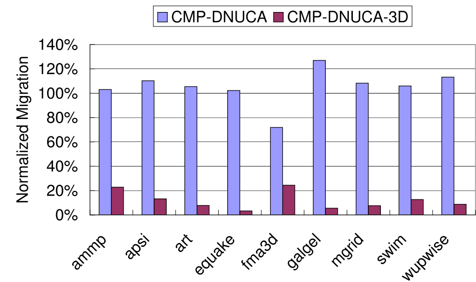

Figure 14. Number of block migrations for CMP-DNUCA and CMP-DNUCA-3D, normalized with respect to CMP-DNUCA-2D.

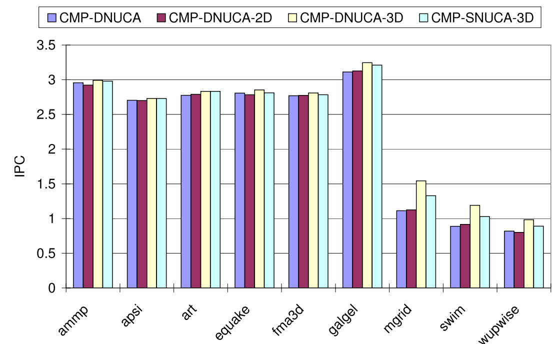

Figure 15. IPC values under different schemes.

2D approach (CMP-DNUCA [2]). Our 2D scheme shows slightly better IPC results for several benchmarks because we place processors not on the edges of the chip, as in CMP-DNUCA, but instead surround them with cache banks as shown in Figure 12. Our results with 3D schemes reiterate the expected benefits from the increase in locality. It is interesting to note that CMP-SNUCA-3D, which does not employ migration, still outperforms the 2D schemes that employ migration. On the average, L2 cache latency reduces by 10 cycles when we move from CMP-DNUCA-2D to CMP-SNUCA-3D. Further gains are also possible in the 3D topology using data migration. Specifically, CMP-DNUCA-3D reduces average L2 latency by 7 cycles as compared to the static 3D scheme. Further, we note that even when employing migration, as shown in Figure 14, 3D

IEEE COMPUTER SOCIETY

Table 5. Our benchmarks.

Benchmarks	ammp	apsi	art	equake	fma3d	galgel	mgrid	swim	wupwise
Fastforward (Million Cycles)	3,633	4,453	3,523	21,538	18,535	3,665	3,533	4,306	18,777
Number of L2 Transactions	24,508,715	27,013,447	25,638,435	27,502,906	12,599,496	38,181,613	204,815,737	164,762,040	141,499,738

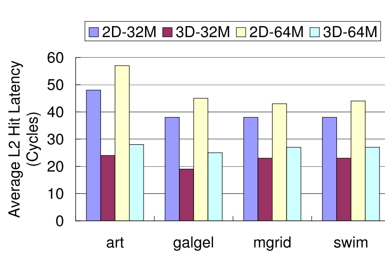

Figure 16. Average L2 hit latency values under different schemes.

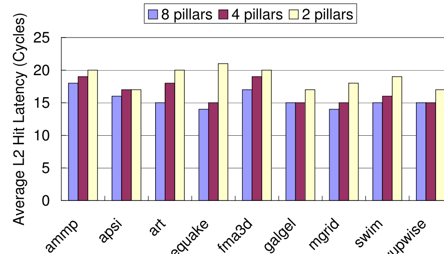

Figure 17. Impact of the number of pillars (the CMP-DNUCA-3D scheme).

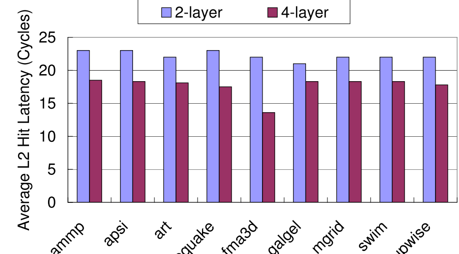

Figure 18. Impact of the number of layers (the CMP-SNUCA-3D scheme).

exercises it much less frequently compared to 2D, due to the increased locality (see Figure 8). The reduced number of migrations in turn reduces the traffic on the network and the power consumption. These L2 latency savings translate to IPC improvements commensurate with the number of L2 accesses. Figure 15 illustrates that the IPC improvements brought by CMP-DNUCA-3D (CMP-SNUCA-3D) over our 2D scheme are up to 37.1% (18.0%). The IPC improvements are higher with mgrid, swim and wupwise since these applications exhibit higher number of L2 accesses.

We next study the impact of larger cache sizes on our savings using CMP-DNUCA-2D and CMP-DNUCA-3D. When we increase the size of the L2 cache, we increase the size of each cluster, while maintaining the 16-way associativity. Figure 16 shows the average L2 latency results with 32MB and 64MB L2 caches for four representative benchmarks (art and galgel with low L1 miss rates and mgrid and swim with high L1 miss rates). We observe that L2 latencies increase with the large cache sizes albeit at a slower rate with the 3D configuration (on average 7 cycles for 2D versus 5 cycles for 3D), indicating that 3D topology is a more scalable option when we move to larger L2 sizes.

Next we make experiments by modifying some of the parameters in the underlying 3D topology. The results with the CMP-DNUCA-3D scheme using different numbers of pillars to capture the effect of the different interlayer via pitches are given in Figure 17. As the number of pillars reduces, the contention for the shared resource (pillar) increases to service interlayer communications. Consequently, average L2 latency increases by 1 to 7 cycles when we move from 8 to 2 pillars. Also, when the number of layers increases from 2 to 4, the L2 latency decreases by 3 to 8 cycles, primarily due to the reduced distances in accessing data, as illustrated in Figure 18 for the CMP-SNUCA-3D scheme. However, it needs to be recalled that the additional layers impose a higher thermal cost as has been shown in Table 3.

6 Conclusions

Three dimensional circuits and Networks-on-Chip (NoC) are two emerging trends for mitigating the growing complexity of interconnects. In this work, we have demonstrated that combining on-chip networks and 3D architectures can be a promising option for designing large L2 cache memories for chip multiprocessors. Specifically, this paper has proposed a novel hybrid bus/NoC fabric to efficiently exploit the fast vertical interconnects in 3D circuits, discussed processor placement and L2 data management issues, and presented an extensive experimental evaluation of the proposed architecture as well as its comparison to 2D L2 cache designs. Experiments have been performed using a novel simulation framework that integrates a 3D NoC simulator for L2 caches with a system level multi-CPU simulator. Our experiments have shown that the proposed 3D architecture reduces average L2 access latency significantly over 2D topologies and this in turn brings IPC benefits. In addition, our results have shown that moving from a 2D topology to a 3D topology can provide more latency reduction than incorporating sophisticated data migration strategies into the 2D topology. We have also demonstrated that the placement of processors needs to be done with care, taking into account the thermal issues. Furthermore, we have shown that the bandwidth of the vertical interconnections (captured by varying the number of pillars) has a significant impact on the L2 cache latencies. Overall, the results of this paper emphasize the importance of considering 3D technology in designing future chip multiprocessors.

Acknowledgements

This work is supported in part by NSF grants EIA-0202007, CAREER 0093085 and 0093082, and a grant from DARPA/MARCO GSRC.

IEEE COMPUTER SOCIETY

References

[1] V. Agarwal, M. Hrishikesh, S. Keckler, and D. Burger. Clock Rate Versus IPC: The End of the Road for Conventional Microarchitectures. In *Proc. the 27th International Symposium on Computer Architecture*, June 2000.

[2] B. M. Beckmann and D. A. Wood. Managing wire delay in large chip-multiprocessor caches. In *Proc. the International Symposium on Microarchitecture*, 2004.

[3] Benini and De Micheli. Networks on Chips: A New SoC Paradigm. *IEEE Computer*, 2002.

[4] B. Black et al. 3D Processing technology and Its Impact on IA32 Microprocessors. In *Proc. the International Conference on Computer Design*, 2004.

[5] Z. Chishti, M. D. Powell, and T. N. Vijaykumar. Distance associativity for high-performance energy-efficient non-uniform cache architectures. In *Proc. the 36th annual IEEE/ACM International Symposium on Microarchitecture*, 2003.

[6] Z. Chishti, M. D. Powell, and T. N. Vijaykumar. Optimization replication, communication, and capacity allocation in CMPs. In *Proc. the International Symposium on Computer Architectures*, 2005.

[7] J. Cong and Y. Zhang. Thermal-Driven Multilevel Routing for 3-D ICs. In *Proc. the Asia South Pacific Design Automation Conference*, Jan. 2005.

[8] W. Dally and B. Towles. Route Packets, Not Wires: On-Chip Inteconnection Networks. In *Proc. the 38th Conference on Design Automation*, 2001.

[9] S. Das et al. Technology, Performance, and Computer Aided Design of Three-Dimensional Integrated Circuits. In *Proc. International Symposium on Physical Design*, 2004.

[10] W. R. Davis et al. Demystifying 3d ics: The pros and cons of going vertical. *IEEE Design and Test of Computers*, 22(6), Nov. 2005.

[11] Y. Deng et al. 2.5D System Integration: A Design Driven System Implementation Schema. In *Proc. the Asia South Pacific Design Automation Conference*, 2004.

[12] L. Hammond, B. Nayfeh, and K. Olukotun. A Single-Chip Multiprocessor. *IEEE Computer Special Issue on "Billion-Transistor Processors"*, Sept. 1997.

[13] R. Ho, K. Mai, and M. Horowitz. The Future of Wires. *Proc. the IEEE*, 89(4), Apr. 2001.

[14] J. Hu and R. Marculescu. Energy- and performance-aware mapping for regular NoC architectures. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 24(4), Apr. 2005.

[15] J. Huh, C. Kim, H. Shafi, L. Zhang, D. Burger, and S. Keckler. A NUCA substrate for flexible CMP cache sharing. In *Proc. the 19th Annual International Conference on Supercomputing*, 2005.

[16] M. Ieong et al. Three Dimensional CMOS Devices and Integrated Circuits. In *Proc. IEEE Custom Integrated Circuits Conference*, 2003.

[17] J. Joyner, P. Zarkesh-Ha, and J. Meindl. A stochastic global net-length distribution for a three-dimensional system-on-a-chip (3D-SoC). In *Proc. 14th Annual IEEE International ASIC/SOC Conference*, Sept. 2001.

[18] S. Jung et al. The Revolutionary and Truly 3-Dimentional 25F2 SRAM Technology with the Smallest S3 Cell, 0.16um2 and SSTFF for Ultra High Density SRAM. In *VLSI Technology Digest of Technical Papers*. 2004.

[19] J. Kahle, M. Day, H. Hofstee, C. Johns, T. Maeurer, and D. Shippy. Introduction to the Cell Multiprocessor. *IBM Journal of Research and Development*, 49(4-5), 2005.

[20] C. Kim, D. Burger, and S. Keckler. An Adaptive, Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches. In *Proc. the 10th International Conference on Architectural Support for Programming Languages and Operating Systems*, Oct. 2002.

[21] J. Kim, D. Park, C. Nicopoulos, N. Vijaykrishnan, and C. Das. Design and analysis of an NoC architecture from performance, reliability and energy perspective. In *Proc. the Symposium on Architecture for Networking and Communications Systems*, Oct. 2005.

[22] P. Kongetira, K. Aingaran, and K. Olukotun. Niagara: A 32-Way Multithreaded SPARC Processor. *IEEE MICRO Magazine*, Apr. 2005.

[23] G. M. Link and N. Vijaykrishnan. Thermal trends in emergent technologies. In *Proc. International Symposium on Quality Electronic Design*, 2006.

[24] P. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, and G. Hallberg. Simics: A full system simulation platform. *IEEE Computer*, 35(2), Feb. 2002.

[25] P. Morrow, M. Kobrinsky, S. Ramanathan, C.-M. Park, M. Harmes, V. Ramachandrarao, H. Park, G. Kloster, S. List, and S. Kim. Wafer-Level 3D Interconnects Via Cu Bonding. In *Proc. the 21st Advanced Metallization Conference*, Oct. 2004.

[26] R. Mullins, A. West, and S. Moore. Low-latency virtual-channel routers for on-chip networks. In *Proc. the 31st Annual International Symposium on Computer Architecture*, June 2004.

[27] K. Olukotun, B. Nayfeh, L. Hammond, K. Wilson, and K.-Y. Chang. The Case for a Single-Chip Multiprocessor. In *Proc. the 7th International Symposium on Architectural Support for Programming Languages and Operating Systems*, Oct. 1996.

[28] L.-S. Peh and W. Dally. A delay model and speculative architecture for pipelined routers. In *The Seventh International Symposium on High-Performance Computer Architecture*, Jan. 2001.

[29] K. Puttaswamy and G. Loh. Implementing Caches in a 3D Technology for High Performance Processors. In *Proc. the International Conference on Computer Design*, Oct. 2005.

[30] T. Richardson, C. Nicopoulos, D. Park, V. Narayanan, Y. Xie, C. Das, and V. Degalahal. A Hybrid SoC Interconnect with Dynamic TDMA-Based Transaction-Less Buses and On-Chip Networks. In *Proc. VLSI Design*, 2006.

[31] P. Rickert. Problems or opportunities? Beyond the 90nm frontier. ICCAD Keynote Address, 2004.

[32] P. Shivakumar and N. Jouppi. Cacti 3.0: An integrated cache timing, power and area model. Technical report, Compaq Computer Corporation, Aug. 2001.

[33] Standard Performance Evaluation Corporation. SPEC OMP. http://www.spec.org/hpg/omp2001/, Dec. 2005.

[34] Sun Microsystems Inc. Sun UltraSPARC T1 Overview. http://www.sun.com/processors/UltraSPARC-T1/, Dec. 2005.

[35] Y.-F. Tsai, Y. Xie, N. Vijaykrishnan, and M. Irwin. Three-Dimensional Cache Design Exploration Using 3D Cacti. In *Proc. the International Conference on Computer Design*, Oct. 2005.

[36] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik. Orion: A power-performance simulator for interconnection networks. In *Proc. the 35th International Symposium on Microarchitecture*, Nov. 2002.

[37] A. Young. Perspectives on 3D-IC Technology. Presentation at the 2nd Annual Conference on 3D Architectures for Semiconductor Integration and Packaging, June 2005.

[38] A. Zeng, J. Lu, K. Rose, and R. Gutmann. First-Order Performance Prediction of Cache Memory with Wafer-Level3D Integration. *IEEE Design and Test of Computers*, 22(6), June 2005.

[39] A. Zhang and K. Asanovic. Victim replication: Maximizing capacity while hiding wire delay in tiled chip multiprocessors. In *Proc. the 32nd International Symposium on Computer Architecture*, 2005.

0-7695-2608-X/06 $20.00 © 2006 IEEE

IEEE COMPUTER SOCIETY