Chapter 8
Three-Dimensional Network-on-Chip Architecture

Yuan Xie, Narayanan Vijaykrishnan, and Chita Das

Abstract On-chip interconnects are predicted to be a fundamental issue in designing multi-core chip multiprocessors (CMPs) and system-on-chip (SoC) architectures with numerous homogeneous and heterogeneous cores and functional blocks. To mitigate the interconnect crisis, one promising option is network-on-chip (NoC), where a general purpose on-chip interconnection network replaces the traditional design-specific global on-chip wiring by using switching fabrics or routers to connect IP cores or processing elements. Such packet-based communication networks have been gaining wide acceptance due to their scalability and have been proposed for future CMPs and SoC design. In this chapter, we study the combination of both three-dimensional integrated circuits and NoCs, since both are proposed as solutions to mitigate the interconnect scaling challenges. This chapter will start with a brief introduction on network-on-chip architecture and then discuss design space exploration for various network topologies in 3D NoC design, as well as different techniques on 3D on-chip router design. Finally, it describes a design example of using 3D NoC with memory stacked on multi-core CMPs.

Y. Xie (✉)
Pennsylvania State University, University Park, PA 16801, USA
e-mail: yuanxie@cse.psu.edu

This chapter includes portions reprinted with permission from the following publications: **(a)** F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir, Design and management of 3D chip multiprocessors using network-in-memory, *Proceedings of International Symposium on Computer Architecture* (2006). Copyright 2006 IEEE. **(b)** J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, and C. Das, A novel dimensionally-decomposed router for on-chip communication in 3D architectures, *Proceedings of International Symposium on Computer Architecture* (2007). Copyright 2007 IEEE. **(c)** P. Dongkook, S. Eachempati, R. Das, A. K. Mishra, Y. Xie, N. Vijaykrishnan, and C. R. Das, Mira: A multi-layered on-chip interconnect router architecture, *Proceedings of International Symposium on Computer Architecture* (2008). Copyright 2008 IEEE.

Y. Xie et al. (eds.), *Three-Dimensional Integrated Circuit Design*,
Integrated Circuits and Systems, DOI 10.1007/978-1-4419-0784-4_8,

8.1 Introduction

As technology scales, integrating billions of transistors on a chip is now becoming a reality. For example, the latest Intel Xeon processor consists of 2.3 billion transistors [25]. At such integration levels, it is imperative to employ parallelism to effectively utilize the transistors. Consequently, modern superscalar microprocessors incorporate many sophisticated micro-architectural features, such as multiple instruction issue, dynamic scheduling, out-of-order execution, speculative execution, and dynamic branch prediction [26]. However, in order to sustain performance growth, future superscalar microprocessors must rely on even more complex architectural innovations. Circuit limitations and limited instruction-level parallelism will diminish the benefits afforded to the superscalar model by increased architectural complexity [26]. 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. The 9-core cell processor [6], the 8-core Sun UltraSPARC T1 processor [13], the 8-core Intel Xeon processor [25], and the 64-core TILEPro64 embedded processor [1] all signal the growing popularity of such systems.

A fundamental issue in designing multi-core chip multiprocessor (CMP) architectures with numerous homogeneous and heterogeneous cores and functional blocks is the design of the on-chip communication fabric. It is projected that on-chip interconnects will be the prominent bottleneck [7] in terms of performance, energy consumption, and reliability as technology scales down further into the nano-scale regime, as discussed in Chapter 1. This is primarily because the scaling of wires increases resistance, and thus the wire delay and energy consumption, while tighter spacing affects signal integrity, and thus reliability. Therefore, design of scalable, high-performance, reliable, and energy-efficient on-chip interconnects is crucial to the success of multi-core/SoC design paradigm and has become an important research thrust.

Traditionally, bus-based interconnection has been widely used for networks with a small number of cores. However, bus-based interconnects will be performance bottlenecks as the number of cores increases. Consequently, they are not considered appropriate for future multi-core systems that have many cores. To overcome these limitations, one promising option is network-on-chip (NoC) [8, 4], where a general purpose on-chip interconnection network replaces the traditional design-specific global on-chip wiring, by using switching fabrics or routers to connect IP cores or processing elements (PEs). Typically, processor cores communicate with each other using a packet-switched protocol, which packetizes data and transmits it through an on-chip network. Much like traditional macro networks, NoC

is very scalable. Figure 8.1 shows a conceptual view of the NoC idea, where many cores are connected by the on-chip network routers, rather than by an on-chip bus.

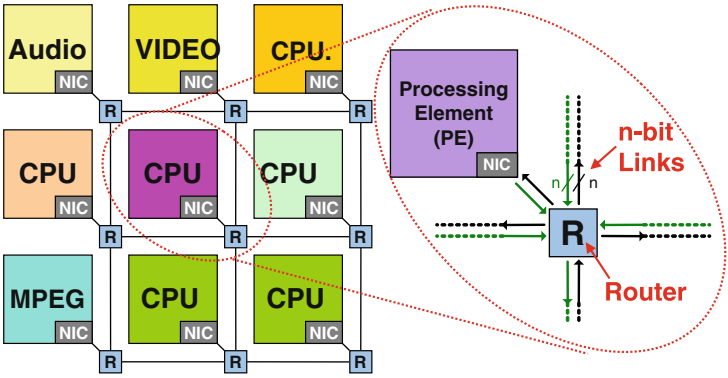

Fig. 8.1 A conceptual network-on-chip architecture: cores are connected to the on-chip network routers (R) via the network interface controllers (NICs)

Even though both 3D integrated circuits [32, 15, 29, 30] and NoCs [8, 4, 23] are proposed as alternatives for the interconnect scaling demands, the challenges of combining both approaches to design three-dimensional NOCs have not been addressed until recently [14, 10, 5, 34, 33, 16, 17, 21].

Chapter 7 presented the design of a single-core microprocessor using 3D integration (via splitting caches or function units into multiple layers) as well as dual-core processor designs using SRAM/DRAM memory stacking. However, all the designs discussed in Chapter 7 are not involved with network-on-chip architecture. In this chapter, we will focus on how to combine 3D integration with network-on-chip as the communication fabric among processor cores and memory banks.

In the following sections, we first give a brief introduction on NoC and then discuss various approaches to design 3D on-chip network topology and 3D router design. An example of memory stacking on top of chip multiprocessors (CMPs) using 3D NoC architecture will be presented.

8.2 A Brief Introduction on Network-on-Chip

Network-on-chip architecture has been proposed as a potential solution for the interconnect demands that arise with the nanometer era [8, 4]. In the network-on-chip

architecture, a general purpose on-chip interconnection network replaces the traditional design-specific global on-chip wiring by the use of switching fabric or routers to connect IP cores or processing elements (PEs). The PEs communicate with each other by sending messages in packets through the routers. This is usually called packet-based interconnect.

A typical 2D NoC consists of a number of processing elements (PE) arranged in a grid-like mesh structure, much like a Manhattan grid. The PEs are interconnected through an underlying packet-based network fabric. Each PE interfaces to a network router through a network interface controller (NIC). Each router is, in turn, connected to four adjacent routers, one in each cardinal direction. The number of ports for a router is defined as the *radix* of the router.

8.2.1 NoC Topology

Network topology is a vital aspect of on-chip network design since it determines the power-performance metrics. For example, NoC topology determines zero-load latency, bisection bandwidth, router micro-architecture, routing complexity, channel lengths, and overall network power consumption.

The mesh topologies [4] (as shown in Fig. 8.1) have been popular for tiled CMPs because of the low complexity and planar 2D-layout properties. Such a simple and regular topology has a compact 2D layout. Other topologies such as concentrated meshes and the flattened butterfly could also be employed in the NoC design for various advantages.

For example, a concentrated mesh (Cmesh) [2] preserves the advantages of a mesh and works around the scalability problem by sharing the router between multiple processing elements. The number of nodes sharing a router is called the concentration degree of the network. Figure 8.2 shows the layout for Cmesh for 64

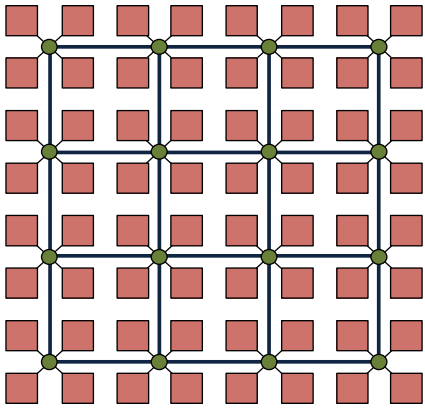

Fig. 8.2 A concentrated mesh network-on-chip topology

nodes. Such topology reduces the number of routers, resulting in reduced hop counts and thus yielding excellent latency savings over mesh. Cmesh has a radix (number of ports) of 8. It can also afford to have very wide channels (512+ bits) due to a slowly growing bisection.

Another example is the flattened butterfly topology [9], which reduces the hop count by employing both concentration and rich connectivity by using longer links to nonadjacent neighbors. The higher connectivity increases the bisection bandwidth and requires a larger number of ports (higher radix) in the router. This increased bisection bandwidth results in narrower channels. Figure 8.3 shows a possible layout for a flattened butterfly with 64 nodes. The rich connectivity trades off serialization latency for reducing the hop count. Such topology has a radix between 7 and 13, depending on the network size and small channel widths (128+ bits).

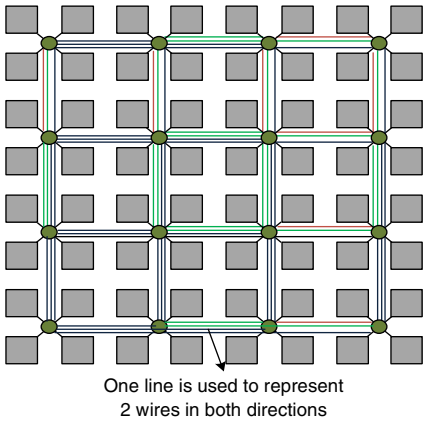

Fig. 8.3 A flattened butterfly topology with 64 nodes

8.2.2 NoC Router Design

A generic NoC router architecture is illustrated in Fig. 8.4. The router has P input and P output channels/ports (The number of ports is defined as radix). When P=5 (or radix is 5), it is a typical 2D NoC router for a mesh network, resulting in a 5 $\times$ 5 crossbar. When the network topology changes, the complexity of the router also changes. For example, a Cmesh network topology requires a router design with a radix of 8. The routing computation unit, RC, operates on the header *flit* (a flit is the smallest unit of flow control; one packet is composed of a number of flits) of an incoming packet and, based on the packet's destination, dictates the appropriate output physical channel/port (PC) and/or valid virtual channels (VC) within the selected output PC. The routing can be deterministic or adaptive. The virtual channel allocation unit (VA) arbitrates between all packets competing for access to

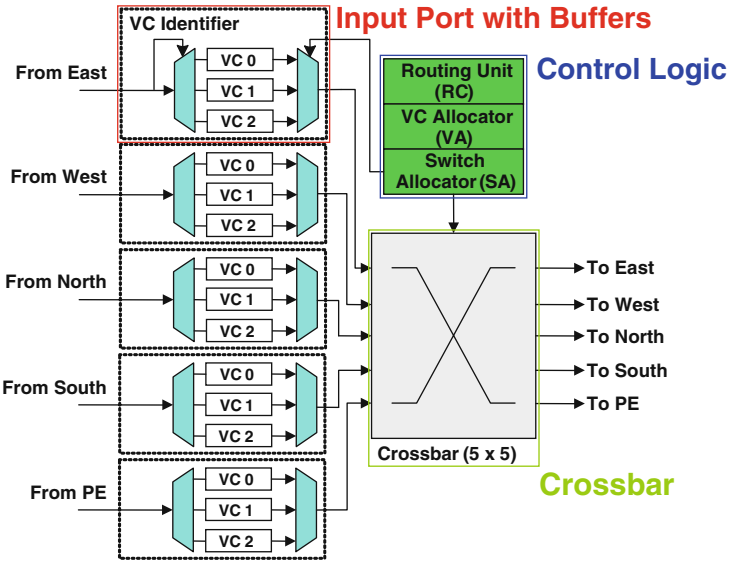

Fig. 8.4 A generic 2D network-on-chip router with five input ports and five output ports

the same output VCs and chooses a winner. The switch allocation unit (SA) arbitrates between all VCs requesting access to the crossbar. The winning flits can then traverse the crossbar and move on to their respective output links.

8.2.3 More Information on NoC Design

Network-on-chip design methodologies have gained a lot of industrial interest. For example, Tilera Corporation has built a 64-core embedded multi-core processor called TILE64 [1], which contains 64 full-featured, programmable cores that are connected by mesh-based NoC architecture. The Intel 80-core TeraFLOPS processor [31] comprises a network-on-chip architecture. The 80-core chip is arranged as an 8×10 array of PE cores and packet-switched routers, connecting with a mesh topology (similar to Fig. 8.1). Figure 8.5 shows the NoC block diagram for the processor. Each PE core contains two pipelined floating-point multiply accumulators (FPMAC), connecting to the router through an interface block (RIB). The router is a 5-port crossbar-based design with mesochronous interface (MSINT). The mesh NoC network provides a bisection bandwidth of 2 Terabits/s.

To learn more about the general background of network-on-chip architecture, one can refer to books [8, 4] and a few survey papers such as [19, 23, 12].

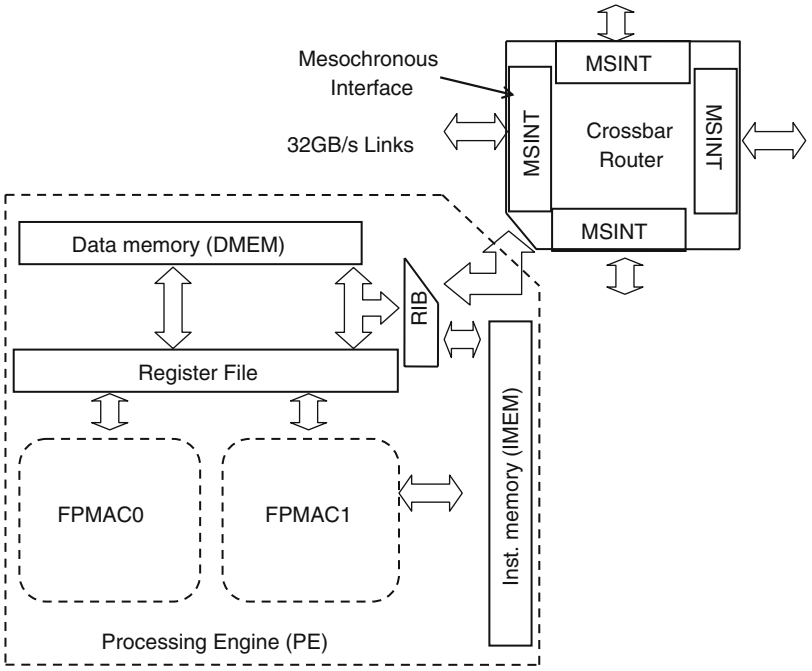

Fig. 8.5 NoC block diagram for Intel's 80-core TeraFLOPS processor

8.3 Three-Dimensional NoC Architectures

This section delves into the exploration of possible architectural designs for 3D NoC architectures. Expanding this 2D paradigm into the third dimension poses interesting design challenges. Given that on-chip networks are severely constrained in terms of area and power resources, while at the same time they are expected to provide ultra-low latency, the key issue is to identify a reasonable tradeoff between these contradictory design threads. In this section, we explore the extension of a baseline 2D NoC implementation into the third dimension while considering the aforementioned constraints.

8.3.1 Symmetric NoC Router Design

The natural and simplest extension to the baseline NoC router to facilitate a 3D layout is simply adding two additional physical ports to each router; one for Up and one for Down, along with the associated buffers, arbiters (VC arbiters and switch arbiters), and crossbar extension. We can extend a traditional NoC fabric to the third dimension by simply adding such routers at each layer (called a symmetric NoC, due

to symmetry of routing in all directions). We call this architecture a 3D symmetric NoC, since both intra- and inter-layer movements bear identical characteristics: hop-by-hop traversal, as illustrated in Fig. 8.6. For example, moving from the bottom layer of a 4-layer chip to the top layer requires three network hops.

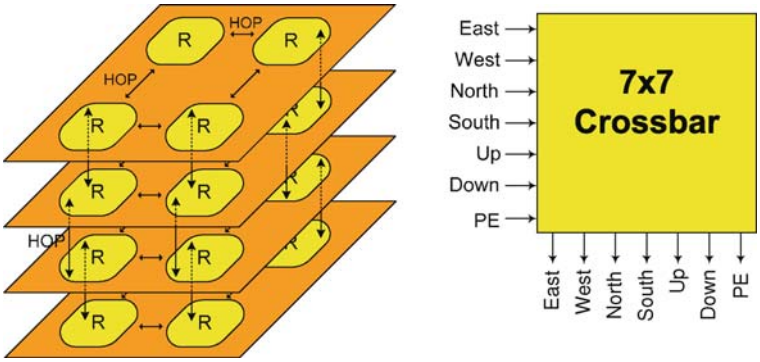

Fig. 8.6 A symmetric 3D network-on-chip router with two additional input/output ports (up and down), totally together needs seven input ports and seven output ports

This architecture, while simple to implement, has a few major inherent drawbacks.

- It wastes the beneficial attribute of a negligible inter-wafer distance in 3D chips (for example, in Chapter 2, we have seen that the thickness of a die could be as small as tens of microns). Since traveling in the vertical dimension is multi-hop, it takes as much time as moving within each layer. Of course, the average number of hops between a source and a destination does decrease as a result of folding a 2D design into multiple stacked layers, but inter-layer and intra-layer hops are indistinguishable. Furthermore, each flit must undergo buffering and arbitration at every hop, adding to the overall delay in moving up/down the layers.
- The addition of two extra ports necessitates a larger 7×7 crossbar, as shown in Fig. 8.6b. Crossbars scale upward very inefficiently, as illustrated in Table 8.1. This table includes the area and power budgets of all crossbar types investigated in this section based on synthesized implementations in 90-nm technology. Clearly, a 7×7 crossbar incurs significant area and power overhead over all other architectures. Therefore, the 3D symmetric NoC implementation is a somewhat naive extension to the baseline 2D network.
- Due to the asymmetry between vertical and horizontal links in a 3D architecture, there are several aspects, such as link bandwidth and buffer allocation, that will need to be customized along different directions in a 3D chip. Further, temperature gradients or process variation across the different layers of the 3D chip can cause identical router components to have different delays in different layers. As

Table 8.1 Area and power comparison of the crossbar switches implemented in 90-nm technology

Crossbar type	Area	Power (500 Mhz)
5×5 Crossbar	8523 μm^2	4.21 mW
6×6 Crossbar	11579 μm^2	5.06 mW
7×7 Crossbar	17289 μm^2	9.41 mW

an example, components operating at the chip layer farthest away from the heat sink will limit the highest frequency of the entire network.

8.3.2 Three-Dimensional NoC–Bus Hybrid Router Design

There is an inherent asymmetry in the delays of a 3D architecture between the fast vertical interconnects and the horizontal interconnects that connect neighboring cores due to differences in wire lengths (a few tens of microns in the vertical direction as compared to a few thousand of microns in the horizontal direction). The previous section argues that a symmetric NoC architecture with multi-hop communication in the vertical (inter-layer) dimension is not desirable.

Given the very small inter-layer distance, single-hop communication is, in fact, feasible. This technique revolves around the fact that vertical distance is negligible compared to intra-layer distances; a shared medium can provide single-hop traversal between any two layers. This realization opens the door to a very popular shared medium interconnect, the bus. The NoC router can be hybridized with a bus link in the vertical dimension to create a 3D NoC–bus hybrid structure as shown in Fig. 8.7. This hybrid system provides both performance and area benefits. Instead of an unwieldy 7 × 7 crossbar, it requires a 6 × 6 crossbar (Fig. 8.7), since the bus

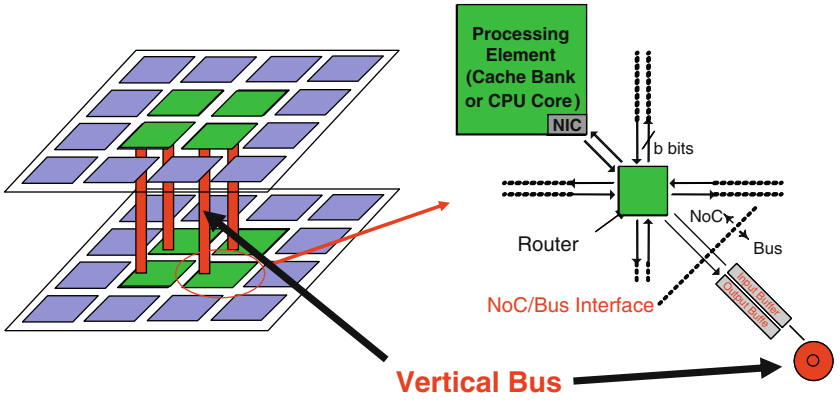

Fig. 8.7 A hybrid 3D NoC/bus architecture. The router has one additional input/output ports to connect with the vertical bus

adds a single additional port to the generic 2D 5 × 5 crossbar. The additional link forms the interface between the NoC domain and the bus (vertical) domain. The bus link has its own dedicated queue, which is controlled by a central arbiter. Flits from different layers wishing to move up/down should arbitrate for access to the shared medium. Figure 8.8 illustrates the view of the vertical via structure. This schematic depicts the usefulness of the large via pads between the different layers; they are deliberately oversized to cope with misalignment issues during the fabrication process. Consequently, it is the large via pads which ultimately limit vertical via density in 3D chips.

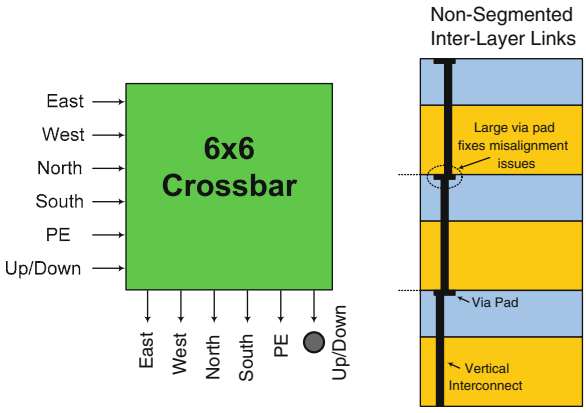

Fig. 8.8 The router has one additional input/output port to connect with the vertical bus, and therefore it needs six input ports and six output ports. The bus is formed by 3D vias connecting multiple layers

Despite the marked benefits over the 3D symmetric NoC router, the bus approach also suffers from a major drawback: it does not allow concurrent communication in the third dimension. Since the bus is a shared medium, it can only be used by a single flit at any given time. This severely increases contention and blocking probability under high network load. Therefore, while single-hop vertical communication does improve performance in terms of overall latency, inter-layer bandwidth suffers. More details on the 3D NoC–bus hybrid architecture can be found in [14].

8.3.3 True 3D Router Design

Moving beyond the previous options, we can envision a true 3D crossbar implementation, which enables seamless integration of the vertical links in the overall router operation. Figure 8.9 illustrates such a 3D crossbar layout. It should be noted at this point that the traditional definition of a crossbar – in the context of a 2D physical layout – is a switch in which each input is connected to each output through a single connection point. However, extending this definition to a physical 3D structure would imply a switch of enormous complexity and size (given

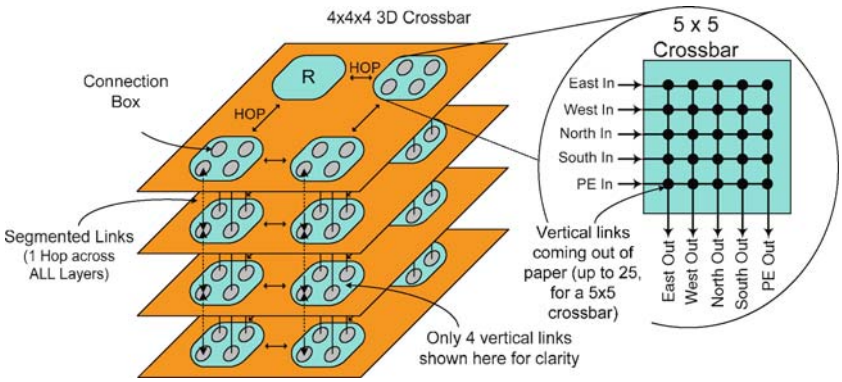

Fig. 8.9 The true 3D router design

the increased numbers of input- and output-port pairs associated with the various layers). Therefore, we chose a simpler structure which can accommodate the interconnection of an input to an output port through more than one connection points. While such a configuration can be viewed as a multi-stage switching network, we still call this structure a crossbar for the sake of simplicity. The vertical links are now embedded in the crossbar and extend to all layers. This implies the use of a 5×5 crossbar, since no additional physical channels need to be dedicated for inter-layer communication.

As shown in Table 8.1, a 5×5 crossbar is significantly smaller and less power-hungry than the 6×6 crossbar of the 3D NoC–bus hybrid and the 7×7 crossbar of the 3D symmetric NoC. Interconnection between the various links in a 3D crossbar would have to be provided by dedicated connection boxes at each layer. These connecting points can facilitate linkage between vertical and horizontal channels, allowing flexible flit traversal within the 3D crossbar. The internal configuration of such a connection box (CB) is shown in Fig. 8.10. The vertical link segmentation also affects the via layout as illustrated in Fig. 8.10. While this layout is more complex than that shown in Fig. 8.8, the area between the offset vertical vias can still be utilized by other circuitry, as shown by the dotted ellipse in Fig. 8.10. Hence, the 2D crossbars of all layers are physically fused into one single three-dimensional crossbar. Multiple internal paths are present, and a traveling flit goes through a number of switching points and links between the input and output ports. Moreover, flits reentering another layer do not go through an intermediate buffer; instead, they directly connect to the output port of the destination layer. For example, a flit can move from the western input port of layer 2 to the northern output port of layer 4 in a single hop.

However, despite this encouraging result, there is an opposite side to the coin which paints a rather bleak picture. Adding a large number of vertical links in a 3D crossbar to increase NoC connectivity results in increased path diversity. This translates into multiple possible paths between source and destination pairs. While this

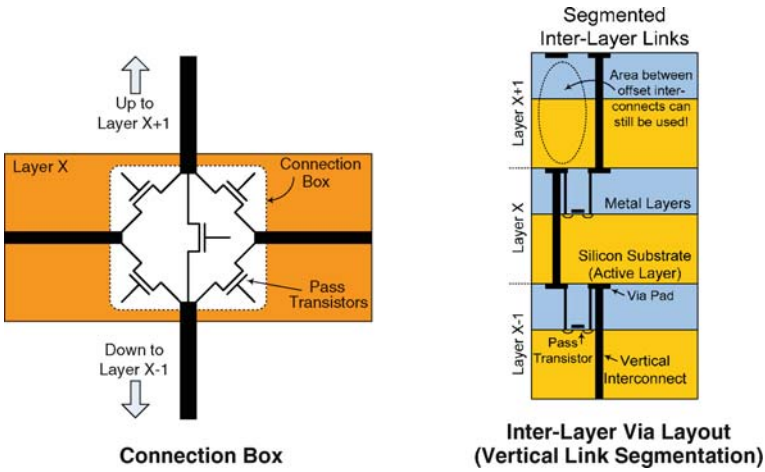

Fig. 8.10 Side view of the inter-layer via structure in 3D crossbar for the true 3D router design

increased diversity may initially look like a positive attribute, it actually leads to a dramatic increase in the complexity of the central arbiter, which coordinates inter-layer communication in the 3D crossbar. The arbiter now needs to decide between a multitude of possible interconnections and requires an excessive number of control signals to enable all these interconnections. Even if the arbiter functionality can be distributed to multiple smaller arbiters, then the coordination between these arbiters becomes complex and time consuming. Alternatively, if dynamism is sacrificed in favor of static path assignments, the exploration space is still daunting in deciding how to efficiently assign those paths to each source–destination pair. Furthermore, a full 3D crossbar implies 25 (i.e., 5×5) connection boxes per layer. A four-layer design would therefore require 100 CBs! Given that each CB consists of six transistors, the whole crossbar structure would need 600 control signals for the pass transistors alone! Such control and wiring complexity would most certainly dominate the whole operation of the NoC router. Pre-programming static control sequences for all possible input–output combinations would result in an oversize table/index; searching through such a table would incur significant delays, as well as area and power overhead. The vast number of possible connections hinders the otherwise streamlined functionality of the switch. Note that the prevailing tendency in NoC router design is to minimize operational complexity in order to facilitate very short pipeline lengths and very high frequency. A full crossbar with its overwhelming control and coordination complexity poses a stark contrast to this frugal and highly efficient design methodology. Moreover, the redundancy offered by the full connectivity is rarely utilized by real-world workloads, and is, in fact, design overkill [10].

8.3.4 3D Dimensionally-Decomposed NoC Router Design

Given the tight latency and area constraints in NoC routers, vertical (inter-layer) arbitration should be kept as simple as possible. Consequently, a true 3D router design, as described in the previous section, is not a realistic option. The design complexity can be reduced by using a limited amount of inter-layer links. This section describes a modular 3D decomposable router (called row–column–vertical (RoCoVe) Router) [10].

In a typical 2D NoC router, the 5×5 crossbar has five inputs/outputs that correspond to the four cardinal directions and the connection from the local PE. The crossbar is the major contributor to the latency and area of a router. It has been shown [11] that through the use of a preliminary switching process known as guided flit queuing, incoming traffic can be decomposed into two independent streams: (a) East–West traffic (i.e., packet movement in the X dimension) and (b) North–South traffic (i.e., packet movement in the Y dimension). Such segregation of traffic flow allows the use of smaller crossbars and the isolation of the two flows in two independent router submodules, which are called *Row Module* and *Column Module* [11].

With the same idea of traffic decomposition, the traffic flow in 3D NoC can be decomposed into three independent streams, with a third traffic flow in the Z dimension (i.e., inter-layer communication). An additional module is required to handle all traffic in the third dimension, and this module is called *Vertical Module*. In addition, there must be links between vertical module and row/column modules to allow the movement of packets from the vertical module to the row module and the column module. Consequently, such a dimensionally decomposed approach allows for a much smaller crossbar design (4×2) resulting in a much faster and power-efficient 3D NoC router design. The architectural view of this 3D dimensionally-decomposed NoC router design is shown in Fig. 8.11. More details can be found in [10].

8.3.5 Multi-layer 3D NoC Router Design

All the 3D router design options discussed earlier (symmetric 3D router, 3D NoC–bus hybrid router, true 3D router, and 3D dimensionally decomposed router) are based on the assumption that the processing element (PE) (which could be a processor core or a cache bank) itself is still a 2D design. In Section 7.4, a fine-granularity design of a microprocessor is introduced such that one can split a PE across multiple layers. For example, 3D cache design and 3D functional units are described in Section 7.4. Three-dimensional block design and the floorplanning algorithms are also discussed in Chapter 4. Consequently, a PE in the NoC architecture can be implemented with such a fine-granularity approach. Although such a multi-layer stacking of a PE is considered aggressive in the current technology, it could

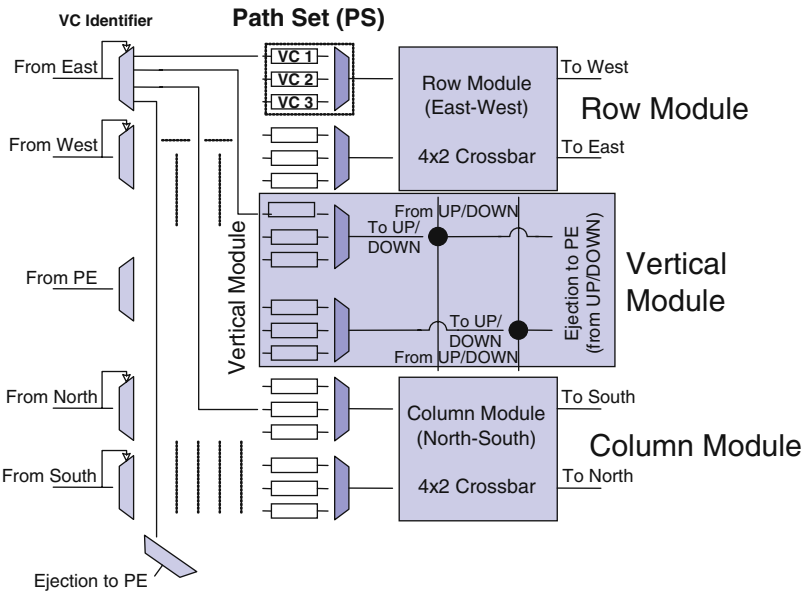

Fig. 8.11 Architectural detail of the 3D-dimensionally decomposed NoC router NoC router design

be possible as 3D technology matures with smaller TSV pitches (as discussed in Section 7.4).

With such a multi-layer stacking of processing elements in the NoC architecture, it is necessary to design a multi-layer 3D router that will span across multiple layers of a 3D chip. Logically, such a NoC architecture with multi-layer PEs and multi-layer routers is identical to the traditional 2D NoC case with the same number of nodes, albeit the smaller area of each PE and router and the shorter distance between routers. Consequently, the design of a multi-layer router requires no additional functionality as compared to a 2D router, and only requires distribution of the functionality across multiple layers.

The router modules can be classified into two categories – separable and non-separable based on the ability to systematically split the module into smaller sub-modules across layers with the inter-layer wiring constraints and the need to balance areas across layers [5]. Input buffers, crossbar, and inter-router links are classified as separable modules, while arbitration logic and routing logic are classified as nonseparable since they cannot be systematically broken into subsets. The saving in chip area can be used for enhancing the router capability, for example, adding express paths between nonadjacent PEs to reduce the average hop count, and help to boost performance and reduce power. Furthermore, because a large portion of the communication traffic consists of short flits and frequent patterns, it is possible to dynamically shut down some layers of the multi-layer router to reduce the power consumption.

8.3.6 3D NoC Topology Design

Until now, all the router designs discussed so far are based on the mesh-based NoC topology. As described in Section 8.2, there exists various NoC topologies, such as the concentrated mesh or the flattened butterfly topologies, all of which have advantages and disadvantages. By employing different topologies rather than the mesh topology, the router designs discussed above could also have different variants. For example, in 2D concentrated mesh topology, the router itself has a radix of 8 (i.e., an 8-port router, with four to local PEs and the others to four cardinal directions). With such topology, the 3D NoC–bus hybrid approach would result in a 9-port router design. Such high-radix router designs are power-hungry with degraded performance, even though the hop count between PEs is reduced. Consequently, a topology-router co-design method for 3D NoC is desirable, so that the hop count between any two PEs and the radix of the 3D router design is as small as possible. Xu et al. [33] proposed a 3D NoC topology with a low diameter and low radix router design. The level 2D mesh is replaced with a network of long links connecting nodes that are at least m mesh-hops away, where m is a design parameter. In such a topology, long-distance communications can leverage the long physical wire and vertical links to reach their destination, achieving low total hop count while the radix of the router is kept low. For application-specific NoC architecture, Yan and Lin [34] also proposed a 3D NoC synthesis algorithm called ripup-reroute-and-router-merging (RRRM) that is based on a rip-up and reroute formulation for routing flows and a router merging procedure for network optimization to reduce the hop count.

8.3.7 Impact of 3D Technology on NoC Designs

Chapter 2 discussed the 3D integration technology options. In this section, the impact of various 3D integration approaches on NoC design is discussed.

Since TSV vias contend with active device area, they impose constraints on the number of such vias per unit area. Consequently, the NoC design should be performed holistically in conjunction with other system components such as the power supply and the clock network that will contend for the same interconnect resources.

The 3D integration using TSV (through silicon via) can be classified into one of the two following categories: (1) *monolithic approach* and (2) *stacking approach*. The first approach involves a sequential device process, where the front-end processing (to build the device layer) is repeated on a single wafer to build multiple active device layers before the back-end processing builds interconnects among devices. The second approach (which could be wafer-to-wafer, die-to-wafer, or die-to-die stacking) processes each active device layer separately using conventional fabrication techniques. These multiple device layers are then assembled to build up 3D ICs using bonding technology. Dies can be bonded face-to-face (F2F) or face-to-back (F2B). The microbump in face-to-face wafer bonding does not go through a thick

buried Si layer and can be fabricated with a higher pitch density. In stacking bonding, the dimension of the TSVs is not expected to scale at the same rate as the feature size because alignment tolerance and thinned die/wafer height during bonding pose limitation on the scaling of the vias.

The TSV (or micropad) size, length, and the pitch density, as well as the bonding method (face-to-face or face-to-back bonding, SOI-based 3D or bulk CMOS-based 3D) can have a significant impact on the 3D NoC topology design. For example, the relatively large size of TSVs can hinder partitioning a design at very fine granularity across multiple device layers and make the true 3D router design less possible. On the other hand, the monolithic 3D integration provides more flexibility in the vertical 3D connection because the vertical 3D via can potentially scale down with feature size due to the use of local wires for connection. Availability of such technologies makes it possible to partition the design at a very fine granularity. Furthermore, face-to-face bonding or SOI-based 3D integration may have a smaller via pitch size and higher via density than face-to-back bonding or bulk CMOS-based integration. This influence of the 3D technology parameters on the NoC topology design will be thoroughly studied, and suitable NoC topologies for different 3D technologies will be identified with respect to the performance, power, thermal, and reliability optimizations.

8.4 Chip Multiprocessor Design with 3D NoC Architecture

In the previous section, various router designs and topology explorations for 3D NoC architecture are discussed. In this section, we use the 3D NoC–bus hybrid architecture as an example to study the chip multiprocessor design with memory stacking that employs 3D NoC architecture and to evaluate the benefits of such an architecture design [14].

The integration of multiple cores on a single die is expected to accentuate the already daunting memory bandwidth problem. Supplying enough data to a chip with a massive number of on-die cores will become a major challenge for performance scalability. Traditional on-chip memory will not suffice due to the I/O pin limitation. According to the ITRS projection, the number of pins on a package will not continue to grow rapidly enough for the next decade to overcome this problem. Consequently, it is anticipated that memory stacking on top of multi-core would be one of the early commercial uses of 3D technology.

The adoption of CMPs and other multi-core systems is expected to increase the sizes 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. 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 nonuniform cache architectures (NUCA) [3] 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 locations 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 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 provides an opportunity to reduce wire lengths and increase memory bandwidth. Consequently, this technology can be useful in reducing the access latencies to the remote cache banks of a NUCA architecture.

Section 7.2 discusses the design of stacking SRAM or DRAM L2 cache for dual-core processors without the need of using NoC architecture or the concept of NUCA architecture. In this section, 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. This section provides new insights on network topology design for 3D NoCs and addresses issues related to data management in L2, taking into account network traffic and thermal issues.

8.4.1 The 3D L2 Cache Stacking on CMP Architecture

As previously mentioned in Chapter 1, one of the advantages of 3D chips is the very small distance between the layers. In Chapter 2, we have seen that the distance between two layers is on the order of tens of microns, which is negligible compared to the distance travelled between two network-on-chip routers in 2D network-on-chip architecture. (For example, 1500 μm on average for a 64 KB cache bank implemented in 65-nm technology.) This characteristic makes communication in the vertical (inter-layer) direction very fast as compared to the horizontal (intra-layer). In this section, we introduce the architecture of stacking a large L2 cache on top of a CMP processor, where a fast access to the stacked L2 cache from CMP processor is enabled by the 3D technology.

As discussed in Section 8.3, a straightforward 3D NoC router design is the symmetric 3D NoC router, which increases the design complexity (using a 7×7 crossbar) and results in multi-hop communication between nonadjacent layers. A NoC–bus hybrid design not only reduces the design complexity (using a 6×6 crossbar), but also results in single-hop communication among the layers because of the short distance between them. In this section, a NoC–bus hybrid architecture is

described, which uses dynamic time-division multiple access (dTDMA) buses as "Communication Pillars" between the wafers, as shown in Fig. 8.7. 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. Due to technological limitations and router complexity issues (to be discussed later), 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.

8.4.2 The dTDMA Bus as a Communication Pillar

The dTDMA bus architecture [24] eliminates the transactional character commonly associated with buses and instead employs a bus arbiter which dynamically grows and shrinks the number of time slots 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 time slot 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 Fig. 8.12.

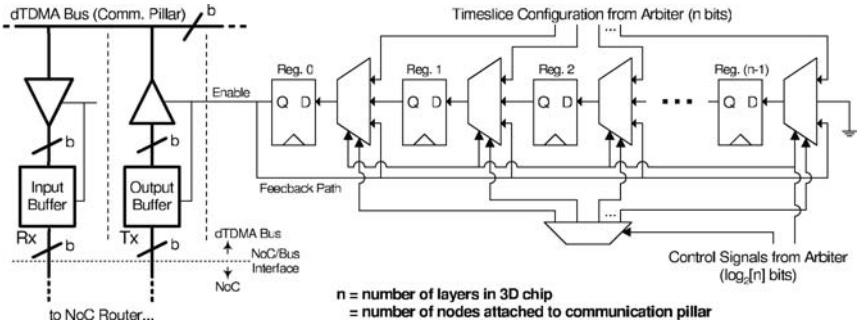

Fig. 8.12 Transceiver module of a dTDMA bus

The dTDMA bus interface (Fig. 8.12) 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. 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 all layers. 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 the 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 8.2. 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. The dTDMA bus is observed to be better than a symmetric 3D router design for the vertical direction as long as the number of device layers is < 9 (bus contention becomes an issue beyond that).

Table 8.2 Area and power overhead of dTDMA bus

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

Table 8.3 Area overhead of inter-wafer wiring for different via pitch sizes

	Inter-wafer area (due to dTDMA bus wiring)			
Bus width	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

As discussed in Chapter 2, the parasitics of TSVs have a small effect on power and delay, because of their small size. The density of the inter-layer vias determines the number of pillars which can be employed. Table 8.3 illustrates the area occupied by a pillar consisting of 170 wires (128-bit bus + 3 $\times$ 14 control wires required in a 4-layer 3D SoC) for different via pitch sizes. In face-to-back 3D implementations, the pillars must pass through the active device layer, 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 negligible compared to the area occupied by the NoC router (see Tables 8.2 and 8.3). However, as previously mentioned in Chapter 2, via densities are still limited by via pad sizes, which are not scaling as fast as the actual via sizes. As shown in Table 8.3, 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, 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 such as contentions.

8.4.3 3D NoC–Bus Hybrid Router Architecture

Section 8.2 provided a brief description of the 3D NoC–bus hybrid router design. A detailed design is described in this section.

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 $\times$ 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 [22]. 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 [20], 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 Fig. 8.7. 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.

8.4.4 Processors and L2 Cache Organization

Figure 8.13 illustrates the organization of the processors and L2 caches in our design. Similar to CMP-DNUCA [3], 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 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.

8.4.5 Cache Management Policies

Based on the organization of processors and caches given in the previous section, 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 sections.

8.4.5.1 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 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 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.

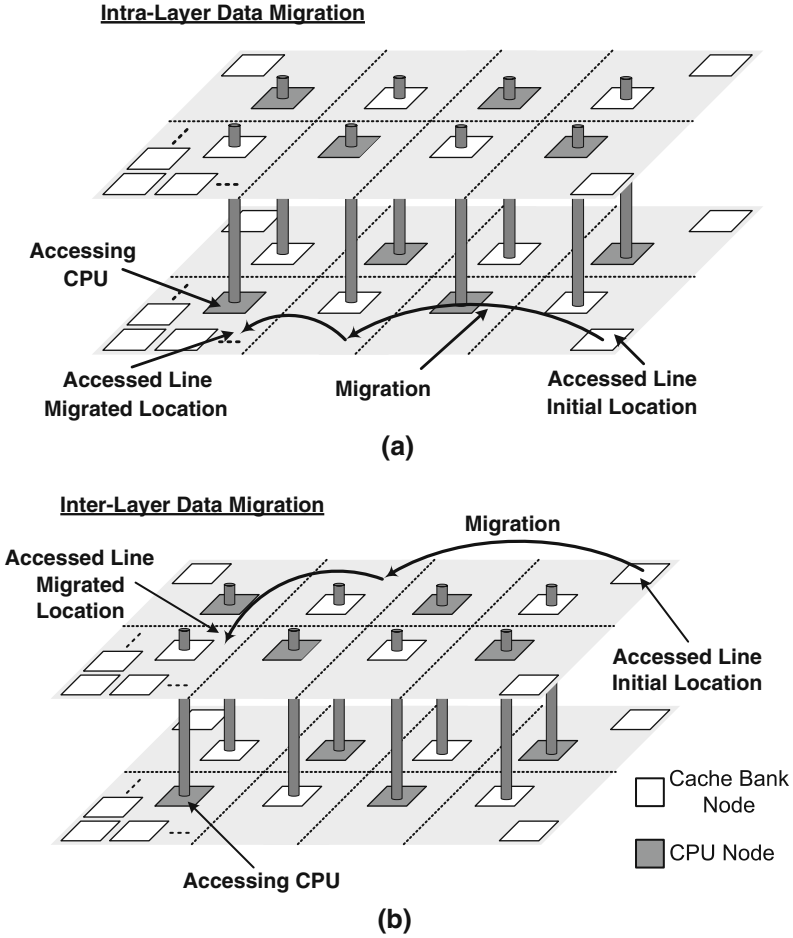

Fig. 8.13 Intra-layer and inter-layer data migration in the 3D L2 architecture. *Dotted lines* denote clusters

8.4.5.2 Placement and Replacement Policy

We use cache placement and replacement policies similar to those of CMP-DNUCA [3]. 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, because 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.

8.4.5.3 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 lie 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 move 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 8.13a 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 Fig. 8.13b). 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 [3].

8.4.6 Methodology

We simulated the 3D CMP architecture by using Simics [18] 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 8.4. 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 the well-known cache simulator Cacti [27].

To model the latency of the 3D, hybrid NoC/bus interconnect, we developed a cycle-accurate simulator in C based on an existing 2D NoC simulator [12]. 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 distributed 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 MESI protocol. The traffic due to L1 cache coherence is taken into account in our simulation.

We simulated nine SPEC OMP benchmarks [28] with our simulation platform. For each benchmark, we marked an initialization phase in the source code. The

Table 8.4 Default system configuration parameters (L2 cache is organized as 16 clusters of size 16 × 64 KB)

Processor parameters	
Number of processors	8
Issue width	1
Memory parameters	
L1 (split I/D)	64 KB, 2-way, 64 B line, 3-cycle, write-through
L2 (unified)	16 MB (256×64 KB), 16-way, 64 B line, 5-cycle bank access
Tag array (per cluster)	24 KB, 4-cycle access
Memory	4 GB, 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

cache model is not simulated until this initialization completes. 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.

8.4.7 Results

We first introduce the schemes compared in our experiments. We refer to the scheme with perfect search from [3] 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 Fig. 8.14. We observe that our 2D scheme (CMP-DNUCA-2D) generates competitive results with the prior 2D approach (CMP-DNUCA [3]). 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 Fig. 8.13. 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

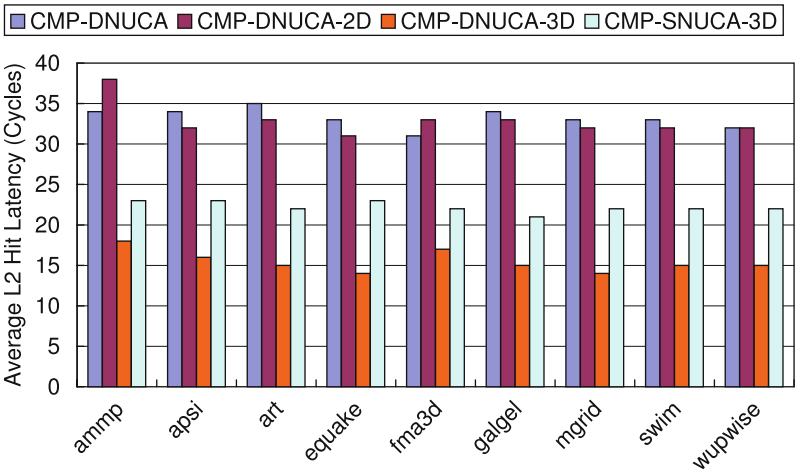

Fig. 8.14 Average L2 hit latency values under different schemes

migration. Specifically, CMP-DNUCA-3D reduces average L2 latency by seven cycles as compared to the static 3D scheme. Further, we note that even when employing migration, as shown in Fig. 8.15, 3D exercises it much less frequently compared to 2D, due to the increased locality. 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. Fig. 8.16 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

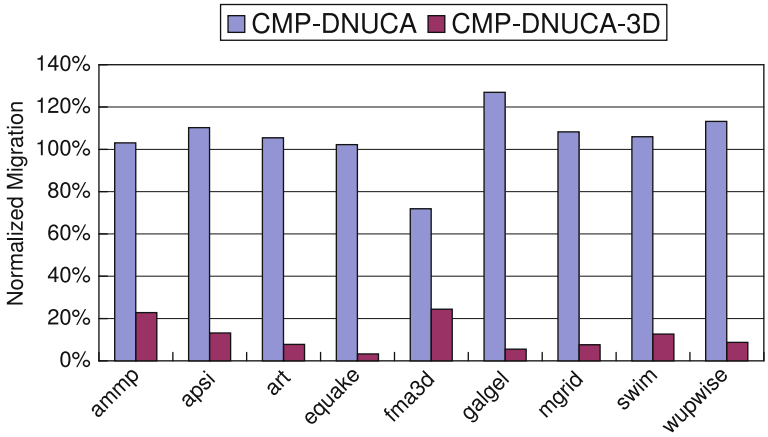

Fig. 8.15 Number of block migrations for CMP-DNUCA and CMP-DNUCA-3D, normalized with respect to CMP-DNUCA-2D

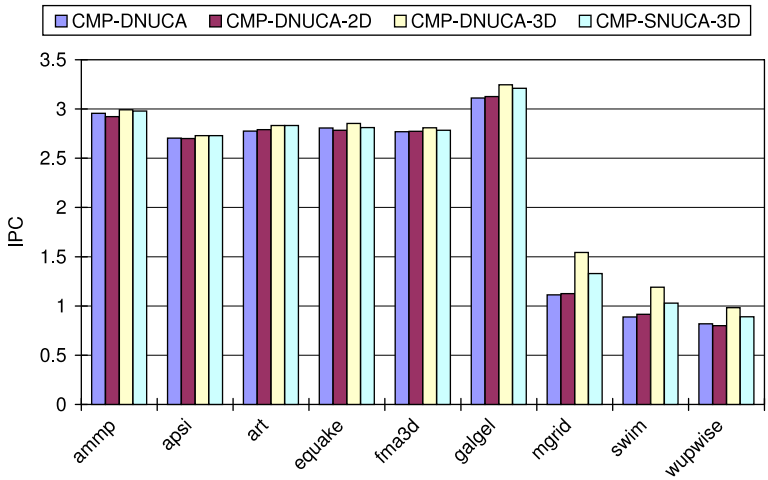

Fig. 8.16 IPC values under different schemes

improvements are higher with mgrid, swim, and wupwise since these applications exhibit a 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 8.17 shows the average L2 latency results with 32 MB and 64 MB 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 seven cycles for 2D vs. five cycles for 3D), indicating that 3D topology is a more scalable option when we move to larger L2 sizes.

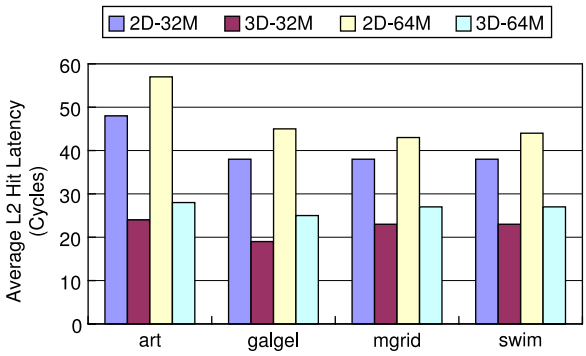

Fig. 8.17 Average L2 hit latency values under different schemes

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 inter-layer via pitches are given in Fig. 8.18. As the number of pillars reduces, the contention for the shared resource (pillar) increases to service inter-layer 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 Fig. 8.19 for the CMP-SNUCA-3D scheme.

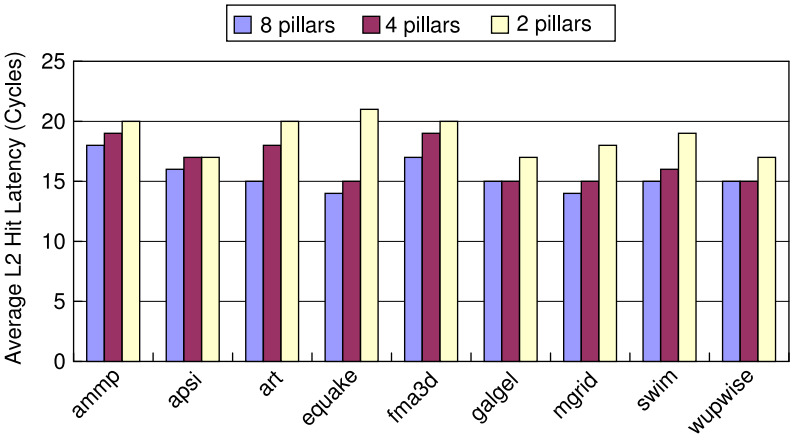

Fig. 8.18 Impact of the number of pillars (the CMP-DNUCA-3D scheme)

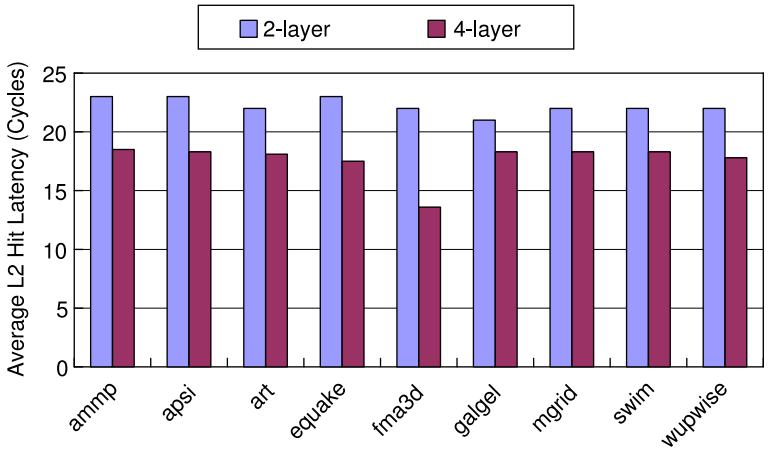

Fig. 8.19 Impact of the number of layers (the CMP-SNUCA-3D scheme)

8.5 Conclusion

Three-dimensional circuits and networks-on-chip (NoC) are two emerging trends for mitigating the growing complexity of interconnects. In this chapter, we describe various approaches to designing 3D NoC architecture and demonstrate that combining on-chip networks and 3D architectures can be a promising option for designing future chip multiprocessors.

Acknowledgments Much of the work and ideas presented in this chapter have evolved over several years of working with our colleagues and graduate students, in particular Professor Mahmut Kandemir, Dr. Mazin Yousif from Intel, Chrysostomos Nicopoulos, Thomas Richardson, Feihui Li, Jongman Kim, Dongkook Park, Reetuparna Das, Asit Mishra, and Soumya Eachempati. The research was supported in part by NSF grants, EIA-0202007, CCF-0429631, CNS-0509251, CCF-0702617, CAREER 0093085, and a grant from DARPA/MARCO GSRC.

References

1. A. Agarwal, L. Bao, J. Brown, B. Edwards, M. Mattina, C. Miao, C. Ramey, and D. Wentzlaff. Tile processor: Embedded multicore for networking and multimedia. In *Proceedings of Hot Chips Symposium*, 2007.
2. J. Balfour and W. J. Dally. Design tradeoffs for tiled CMP on-chip networks. In *Proceedings of International conference on Supercomputing*, pp. 187–198, 2006.
3. B. M. Beckmann and D. A. Wood. Managing wire delay in large chip-multiprocessor caches. In *Proceedings of International Symposium on Microarchitecture*, pp. 319–330, 2004.
4. G. De Micheli and L. Benini. *Networks on Chips*. Morgan Kaupmann, San Francisco, CA, 2006.
5. P. Dongkook, S. Eachempati, R. Das, A. K. Mishra, Y. Xie, N. Vijaykrishnan, and C. R. Das. Mira: A multi-layered on-chip interconnect router architecture. In *Proceedings of International Symposium on Computer Architecture*, pp. 251–261, 2008.
6. M. Gschwind, P. Hofstee, B. Flachs, M. Hopkins, Y. Watanabe, and T. Yamazaki. A compiler enabling and exploiting the cell broadband processor architecture. *IBM Systems Journal Special Issue on Online Game Technology*, 45(1), 2006.
7. R. Ho, K. Mai, and M. Horowitz. The future of wires. *Proceedings of the IEEE*, 89(4):490–504, April 2001.
8. A. Jantsch and H. Tenhunen. *Networks on Chip*. Kluwer Academic Publishers, Boston, 2003.
9. J.Kim, J. Balfour, and W. J. Dally. Flattened butterfly topology for onchip networks. In *Proceedings of International Symposium on Microarchitecture*, pp. 172–182, 2007.
10. J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, and C. Das. A novel dimensionally-decomposed router for on-chip communication in 3D architectures. In *Proceedings of International Symposium on Computer Architecture*, pp. 138–149, 2007.
11. J. Kim, C. Nicopoulos, D. Park, V. Narayanan, M. S. Yousif, and C. Das. A gracefully degrading and energy-efficient modular router architecture for on-chip networks. In *Proceedings of International Symposium on Computer Architecture*, pp. 4–15, 2006.
12. 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 *Proceedings of Symposium on Architecture for Networking and Communications Systems*, pp. 173–182, October 2005.
13. P. Kongetira, K. Aingaran, and K. Olukotun. Niagara: A 32-way multithreaded SPARC processor. *IEEE MICRO*, 25(2):21–29, 2005.
14. F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir. Design and management of 3D chip multiprocessors using network-in-memory. In *Proceedings of International Symposium on Computer Architecture*, pp. 130–141, 2006.

15. G. H. Loh. 3D-stacked memory architectures for multi-core processors. In *Proceedings of International Symposium on Computer Architecture*, pp. 453–464, 2008.
16. I. Loi, F. Angiolini, and L. Benini. Developing mesochronous synchronizers to enable 3D NoCs. In *Proceedings of Design, Automation and Test in Europe Conference*, pp. 1414–1419, 2008.
17. I. Loi, S. Mitra, T. H. Lee, S. Fujita, and L. Benini. A low-overhead fault tolerance scheme for tsv-based 3D network on chip links. In *Proceedings of International Conference on Computer-Aided Design*, pp. 598–602, 2008.
18. P. Magnusson, M. Christensson, J. Eskilson, D. Forsgren, and G. Hallberg. Simics: A full system simulation platform. *IEEE Computer*, 35(2):50–58, February 2002.
19. R. Marculescu, U. Y. Ogras, L. S. Peh, N. E. Jerger, and Y. Hoskote. Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(1):3–21, January 2009.
20. R. Mullins, A. West, and S. Moore. Low-latency virtual-channel routers for on-chip networks. In *Proceedings of International Symposium on Computer Architecture*, p. 188, June 2004.
21. V. F. Pavlidis and E. G. Friedman. 3-D topologies for networks-on-chip. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 15(10):1081–1090, 2007.
22. L. Peh and W. Dally. A delay model and speculative architecture for pipelined routers. In *Proceedings of International Symposium on High Performance Computer Architecture*, pp. 255–266, January 2001.
23. A. Pullini, F. Angiolini, S. Murali, D. Atienza, G. De Micheli, and L. Benini. Bringing NoCs to 65 nm. *IEEE Micro*, 27(5):75–85, 2007.
24. 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 *Proceedings of International Symposium on VLSI Design*, pp. 657–664, 2006.
25. S. Rusu, S. Tam, H. Muljono, J. Stinson, D. Ayers, J. Chang, R. Varada, M. Ratta, and S. Kottapalli. A 45 nm 8-core enterprise xeo processor. In *Proceedings of International Solid-State Circuits Conference*, February 2009.
26. J. Shen and M. Lipasti. *Modern Processor Design: Fundamentals of Superscalar Processors*. McGraw-Hill, Boston, 2005.
27. P. Shivakumar and N. Jouppi. Cacti 3.0: An integrated cache timing, power and area model. In *Technical Report, Compaq Computer Corporation*, August 2001.
28. Standard Performance Evaluation Corporation. SPEC OMP. http://www.spec.org.
29. G. Sun, X. Dong, Y. Xie, J. Li, and Y. Chen. A novel architecture of the 3D stacked mram l2 cache for CMPs. In *Proceedings of International Symposium on High Performance Computer Architecture*, pp. 239–249, 2009.
30. B. Vaidyanathan, W. Hung, F. Wang, Y. Xie, N. Vijaykrishnan, and M. Irwin. Architecting microprocessor components in 3D design space. In *Proceedings of International Conference on VLSI Design*, pp. 103–108, 2007.
31. S. R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar. An 80-tile sub-100-w teraFLOPS processor in 65-nm cmos. *IEEE Journal of Solid-State Circuits*, 43(1):29–41, 2008.
32. Y. Xie, G. H. Loh, B. Black, and K. Bernstein. Design space exploration for 3D architectures. *ACM Journal of Emerging Technology of Computer Systems*, 2(2):65–103, 2006.
33. Y. Xu, Y. Du, B. Zhao, X. Zhou, Y. Zhang, and J. Yang. A low-radix and low-diameter 3D interconnection network design. In *Proceedings of International Symoposium on High Performance Computer Architecture*, pp. 30–41, 2009.
34. S. Yan and B. Lin. Design of application-specific 3D networks-on-chip architectures. In *Proceedings of International Conference of Computer Design*, pp. 142–149, 2008.