Journal of Systems Architecture 56 (2010) 429–441

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Exploiting address compression and heterogeneous interconnects for efficient message management in tiled CMPs

Antonio Flores *, Manuel E. Acacio, Juan L. Aragón

Departamento de Ingeniería y Tecnología de Computadores, University of Murcia, 30100 Murcia, Spain

ARTICLE INFO

Article history:
Received 2 December 2008
Received in revised form 27 April 2010
Accepted 7 May 2010
Available online 19 May 2010

Keywords:
Tiled chip multiprocessor
Energy-efficient architecture
Cache-coherence protocol
Heterogeneous on-chip interconnection network

ABSTRACT

High performance processor designs have evolved toward architectures that integrate multiple processing cores on the same chip. As the number of cores inside a Chip MultiProcessor (CMP) increases, the interconnection network will have significant impact on both overall performance and energy consumption as previous studies have shown. Moreover, wires used in such interconnect can be designed with varying latency, bandwidth and power characteristics. In this work, we show how messages can be efficiently managed in tiled CMP, from the point of view of both performance and energy, by combining both address compression with a heterogeneous interconnect. In particular, our proposal is based on applying an address compression scheme that dynamically compresses the addresses within coherence messages allowing for a significant area slack. The arising area is exploited for wire latency improvement by using a heterogeneous interconnection network comprised of a small set of very-low-latency wires for critical short-messages in addition to baseline wires. Detailed simulations of a 16-core CMP show that our proposal obtains average improvements of 10% in execution time and 38% in the energy-delay^2 product of the interconnect. Additionally, the sensitivity analysis shows that our proposal performs well when either OoO cores or caches with higher latencies are considered.

1. Introduction

As implementation technology scales down, high performance processor designs have evolved toward architectures that integrate multiple processing cores on a single die. Within this decade, it is expected that Chip Multiprocessors (CMP)s with 10s of cores on the die will be a reality. Additionally, future many-core CMPs with several tens of processor cores probably will be designed as arrays of replicated tiles connected over an on-chip switched direct network. Each tile contains a processing core with primary caches, a slice of the L2 cache, and a connection to the on-chip network. This type of caches is referred as Non-Uniform Cache Architecture (NUCA) as the cache access time differs depending on whether we access to the local L2 cache slice or a remote one [17]. These tiled architectures have been claimed to provide a scalable solution for managing the design complexity, and effectively using the resources available in advanced VLSI technologies. Maybe, one of the best known examples of a tiled CMP architecture nowadays is the 80-core Intel's Polaris prototype [27]. Another example is the TILE64 processor developed by Tilera [29] – a 64-core tiled multi-core architecture connected over five 2D mesh networks, each one specialized for a different use and inspired by the MIT's RAW architecture [26].

One of the greatest bottlenecks to high performance and energy-efficiency in such tiled CMP architectures is the high cost of on-chip communication through global wires [13]. Wang et al. [28] reported that the on-chip network of the MIT's RAW processor consumes 36% of the total chip power. Magen et al. [21] also attribute 50% of overall chip power to the interconnect. Most of this power is dissipated in the point-to-point links of the interconnect [28]. Thus, wires pose major performance and power dissipation problems as technology shrinks and total die area increases. This trend will be exacerbated in future many-core CMP designs. Therefore, as communication emerges as a larger power and performance constraint than computation itself, wire properties should be exposed to architects in order to enable them to find ways to exploit these properties.

One way to address the problems that wire delay and power dissipation will pose on future many-core CMPs is the use of heterogeneous on-chip interconnection networks [1], i.e., an interconnect with links that are comprised of wires with varying physical properties. By tuning wire width and spacing, it is possible to design wires with varying latency and bandwidth properties. Similarly, by tuning repeater size and spacing, it is possible to design wires with varying latency and energy properties [2]. Cheng et al. [6] apply this observation to develop a heterogeneous interconnect

* Corresponding author. Tel.: +34 868884638; fax: +34 868884151.
E-mail addresses: aflores@ditec.um.es (A. Flores), meacacio@ditec.um.es (M.E. Acacio), jlaragon@ditec.um.es (J.L. Aragón).

1383-7621/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.sysarc.2010.05.006

comprised of two new wire implementations apart from baseline wires (*B-Wires*): power optimized wires (*PW-Wires*) and bandwidth optimized wires (*L-Wires*) in order to fit the latency and bandwidth requirements of the different kinds of coherence messages.

Another approach, which is orthogonal to the previous one, to alleviate these problems is to *transcode* the transferred information in order to better exploit the interconnect bandwidth. The area slack created due to compression could then be exploited to further improve the latency of some particular wires. This paper explores such an approach by proposing the use of an address compression scheme in the context of a heterogeneous interconnect that allows most of the critical messages, used to ensure coherence between the L1 caches of a CMP, to be compressed in a few bytes and transmitted using very-low-latency wires meanwhile the rest of messages are transmitted using baseline wires. It is important to note that we are not aimed at proposing a particular compression scheme but at exploiting the area slack due to compression for improving critical messages latency by means of using a heterogeneous interconnect. Detailed simulations of a 16-core CMP show average improvements of 10% in execution time and 38% in the energy-delay^2 product of the interconnect (28% for the full CMP) when such a heterogeneous interconnection network is used in conjunction with an address compression scheme.

In [10] we presented a preliminary version of this work. The major new contributions that we make in this extended manuscript are the following:

- A more up-to-date configuration of the CMP is evaluated. Particularly, we consider a 32 nm process technology in order to use current technology in the industry.
- The experimental results' section has been extended to evaluate the improvement in both execution time and ED^2P metric when the compressed messages are sent through *VL-Wires* using fragmentation. In this way, a better understanding of the benefits of our proposal is gained.
- A new section with a sensitivity analysis has been included in order to evaluate the effect of our proposal when considering the following: out-of-order processing cores, narrower links, relative latency of the interconnect with respect to the second-level cache and, finally, different sizes for the data that partial replies carry on.

The rest of the paper is organized as follows: Section 2 reviews some related work, traditional address compression schemes and presents a background on techniques that enable different wire implementations and the design of a heterogeneous interconnect. Our proposal for optimizing the on-chip interconnection network energy and performance in tiled CMPs is presented in Section 3. Section 4 describes the evaluation methodology and presents the results of the proposed mechanism. A sensitivity analysis is presented in Section 5. Finally, Section 6 summarizes the main conclusions of the work.

2. Preliminaries

2.1. Related work

The on-chip interconnection network is a critical design element in a multi-core architecture and, consequently, it is the subject of several recent works. Among others, Kumar et al. [18] analyze several on-chip interconnection mechanisms and topologies, and quantify their area, power, and latency overheads. The study concludes that the design choices for the interconnect have a significant effect on the rest of the chip, potentially consuming a significant fraction of the real estate and power budget.

Several address compression schemes have been proposed to reduce their energy and/or area cost. Address compression schemes based on using a small compression cache were first proposed in [9] and were used to reduce off-chip bus widths and pin counts. Recent attempts at using compression to reduce on-chip wire delay and/or energy consumption have been presented in the last years. In [7], Criton proposes to exploit low entropy in order to reduce wire delay. However, in that work, performance impact results are based on estimations instead of using simulations. Liu et al. [20] propose an evolution of the mechanism described in [9]. They describe a partial match compression (PMC) scheme for the address bus between the L1 and L2 caches in a single-core processor executing sequential applications in order to reduce wire delay by increasing inter-wire spacing. Performance improvement of up to 16% is obtained for some configurations, however, they do not estimate the effect of their proposal on energy consumption.

Basu et al. [3] propose placing a value cache at both ends of a communication channel. Upon a *hit*, the system sends the index to the cache entry instead of the whole word, to reduce bit transitions. Parcerisa and Gonzalez [23] applied value prediction to inter-cluster communication on clustered microarchitectures in order to reduce long wire delays. In [19], the authors propose a communication value cache (CVC) to reduce the number of bit transfers between processors inside a bus-based CMP architecture. Differently from previous works, we show how address compression could be used in conjunction with a heterogeneous network to improve performance and reduce energy consumption in tiled CMP architectures.

In addition to address compression schemes, some authors have recently proposed to compress NoC traffic through exploiting frequent data patterns/values. In [8], the authors discovered that some patterns occur with very high frequencies in on-chip traffic. They then propose to take advantage of this observation by using a cache compression mechanism that allows to increase the effective cache capacity along with network compression in order to reduce network latency with additional power savings. Zhou et al. propose to exploit frequent values instead of value patterns [34], using a very small table for end-to-end communications where frequent values are stored. When sending a data message, values are compared with the values stored in that table and on a hit the index of the table is sent. Otherwise the original value is used. In this way, they obtain average reductions of 24% in the data message length with reductions of up to 16.7% in the router power. In [15], it is proposed a similar data compression scheme where indexes are stored in a shared table instead of per flow basis. An additional management protocol ensures that in-order delivery is not required. Both proposals can be applied simultaneously with our proposal of using a heterogeneous interconnection network comprised of a small set of very-low-latency wires (*VL-Wires*) for critical short-messages in addition to baseline wires. But, with average reductions of 24% in the length of data message [34], the use of *VL-Wires* for compressed data messages has to be discarded because of the very limited number of wires of this type.

On the other hand, a reduced number of works have attempted to exploit the properties of a heterogeneous interconnection network at the microarchitecture level in order to reduce the interconnect energy share. Beckmann and Wood [4] propose the use of transmission lines to access large L2 on-chip caches in order to reduce the required cache area and the dynamic power consumption of the interconnection network. In [1], Balasubramonian et al. make the first proposal of wire management at the microarchitecture level. They introduce the concept of a heterogeneous interconnect that is comprised of wires with varying area, latency, bandwidth, and energy characteristics, and they apply it to register

communication within a clustered architecture. In particular, cache accesses are accelerated by sending a subset of the address bits on low-latency wires to prefetch data out of the L1 D-cache, while non-critical register values are transmitted on low-power wires. They extend this proposal in [22] with techniques aimed at accelerating cache accesses in large L2/L3 split caches (L2/L3 NUCA architectures [17]) by taking advantage of a lower-bandwidth, lower-latency network.

Recently, Cheng et al. [6] applied the heterogeneous network concept to the cache-coherence traffic problem in CMPs. In particular, they propose an interconnection network composed of three sets of wires with varying latency, bandwidth and energy characteristics, and map coherence messages to the appropriate set taking into account their latency and bandwidth needs. They report significant performance improvement and interconnect energy reduction when a two-level tree interconnect is used to connect the cores and the L2 cache. Unfortunately, insignificant performance improvements are reported for direct topologies (such as the 2D mesh typically employed in tiled CMPs [27]).

More recently, we have proposed in [11] *Reply Partitioning*, a technique that allows all coherence messages to be classified into two groups: critical and short, and non-critical and long. In particular, *Reply Partitioning* focuses on replies that carry data and split them into a critical and short *Partial Reply* message that carries the word requested by the processor, in addition to a non-critical *Ordinary Reply* with the full cache block. *Reply Partitioning* aims to use a heterogeneous interconnection network comprised of low-latency wires for critical messages and low-energy wires for non-critical ones, which also allows for a more balanced workload. Note that the proposal presented in the current paper is orthogonal to that, and could be used to accelerate even more the low-latency wires.

Finally, other works have proposed to expose the interconnection network properties directly to the software. Thus, the MIT's RAW architecture uses two static (routes are specified at compile time) and two dynamic (routes are specified at run time) 32-bit full-duplex on-chip networks [26]. These on-chip networks are exposed to the software through the RAW ISA as network hops, thereby giving the programmer or compiler the ability to carefully orchestrate the transfer of data values between tiles in a similar way to the routing in an ASIC. A similar approach is used in the TILE64 processor architecture with one static and four dynamic on-chip networks [29]. Differently from these works, our proposal is not only focused on performance but also on energy reduction by exploiting the properties of a heterogeneous interconnection network at the microarchitecture level.

2.2. Address compression schemes

As mentioned before, address buses have been widely studied in the literature and several strategies have been adopted in order to eliminate redundant and/or useless information. Dynamic compression schemes were first proposed by Farrens et al. [9]. Their Dynamic Base Register Caching (DBRC) scheme consists of a small compression cache at the sending end, and register files at the receiving end of a processor-memory address bus. When a *hit* happens in the sender compression cache, the system sends the index to the cache entry instead of the whole address, thus reducing the number of wires needed to implement the address bus. In [20] the authors present Partial Match Compression Scheme (PMCS) that exploits the spatial and temporal locality of addresses to dynamically compress them depending on the extent to which they match the higher-order portions of recently-occurring addresses saved in a small *compression cache*. When a maximal match occurs, the address is compressed to the maximum extent and is transmitted on a narrow bus in one cycle. When a partial match occurs, one or more extra cycles are required. Fig. 1 shows PMCS when three candidates for partial match are considered. For a complete hit/miss the behavior is the same than in DBRC, but when a partial hit happens, the additional bits codify which low-order portion (T_{miss}) is sent through the bus.

Fig. 2 (left) shows how to adapt the DBRC scheme to a tiled CMP architecture in which the global bus is replaced by a switched direct network (similar adaption can be done for the PMCS). An alternative compression scheme is shown in Fig. 2 (right). In this case, the compression cache at the sending end is replaced with a base register that stores the last non-compressed address sent by the source core to that destination. At the receiving end, a similar base register also stores that address. When the *difference* between the address stored at the sending end and a subsequent address can be represented using less than a fixed number of bits, the system sends just the difference instead of the whole address, updating the information stored in the base register in both the sending and the receiving cores. This simple scheme is based on the fact that many memory references, beyond affine access functions, can be easily predicted as they follow a stride pattern [24].

In order to guarantee a proper synchronization of address values in both ends of the communication channel when using a switched direct network, all theses hardware structures are maintained per a flow [34]. Although it might be possible to use a shared table scheme as in [15], the complexity of synchronization may offset the benefits in area savings. For a 16-core processor the

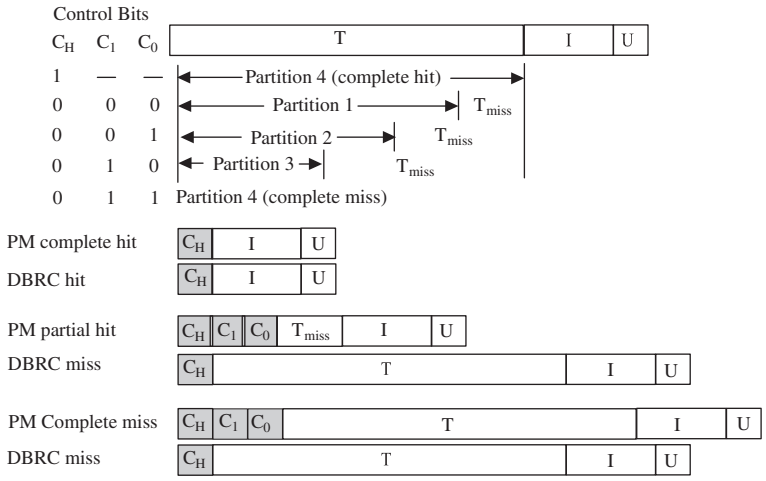

Fig. 1. Partial match address compression scheme (from Ref. [20]).

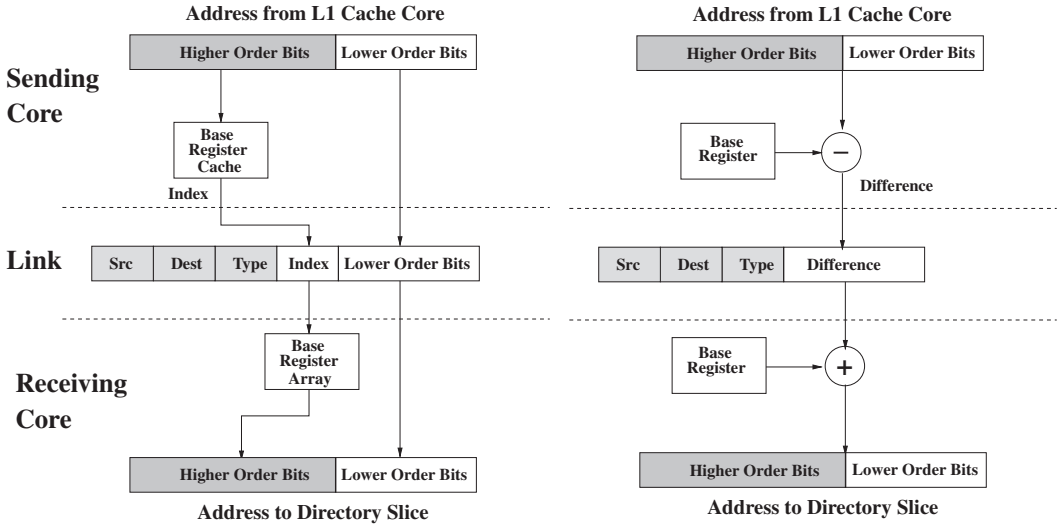

Fig. 2. Organization of the DBRC (left) and Stride (right) address compression schemes for tiled CMP architectures.

Table 1
Overall area and power characteristics of different address compression schemes for a 16-core tiled CMP. For comparison purposes, in parenthesis we show the area and power relative to one of the cores of the CMP.

Compression scheme	Size (bytes)	Area (mm^2)	Max. dyn. power (W)	Static power (mW)
4-Entry DBRC	1088	0.0175 (0.29%)	0.0762 (0.70%)	3.38 (0.05%)
16-Entry DBRC	4352	0.0649 (1.07%)	0.2753 (2.49%)	13.48 (0.19%)
64-Entry DBRC	17,408	0.1996 (3.30%)	0.5064 (4.58%)	41.80 (0.60%)
2-Byte Stride	272	0.0062 (0.1%)	0.0401 (0.36%)	1.61 (0.02%)

overall area overhead is less than 5% of the area needed for a core (see Table 1).

We have evaluated the effectiveness of the compression schemes described before in the context of a tiled CMP running parallel applications. Fig. 3 shows the fraction of compressed addresses using different configurations for the DBRC and Stride compression schemes (see Section 4 for further details about the 16-core CMP configuration and working sets evaluated). Results for PMCS are not presented because its compression coverage is similar to the simpler and more cost-effective DBRC scheme. Some interesting results can be pointed out. First, if we try to keep address compression size to 1 byte, a low compression coverage is obtained for both the Stride compression scheme, and the DBRC scheme when a small compression cache is considered (1-byte Stride and 4-entry DBRC (1B LO) bars). In order to obtain acceptable compression coverages (over 80%), we need either to implement a compression cache with, at least, 16 entries (16-entry DBRC (1B LO) bar); or to use an address compression size of 2 bytes (2-byte Stride or 4-entry DBRC (2B LO) bars). In the last case, the Stride compression scheme obtains similar results to an equivalent DBRC configuration eliminating the need of using an adder. These results show that, in this case, compression schemes based on the use of stride patterns perform badly in comparison with other proposals. Finally, DBRC with a compression size of 2 bytes shows

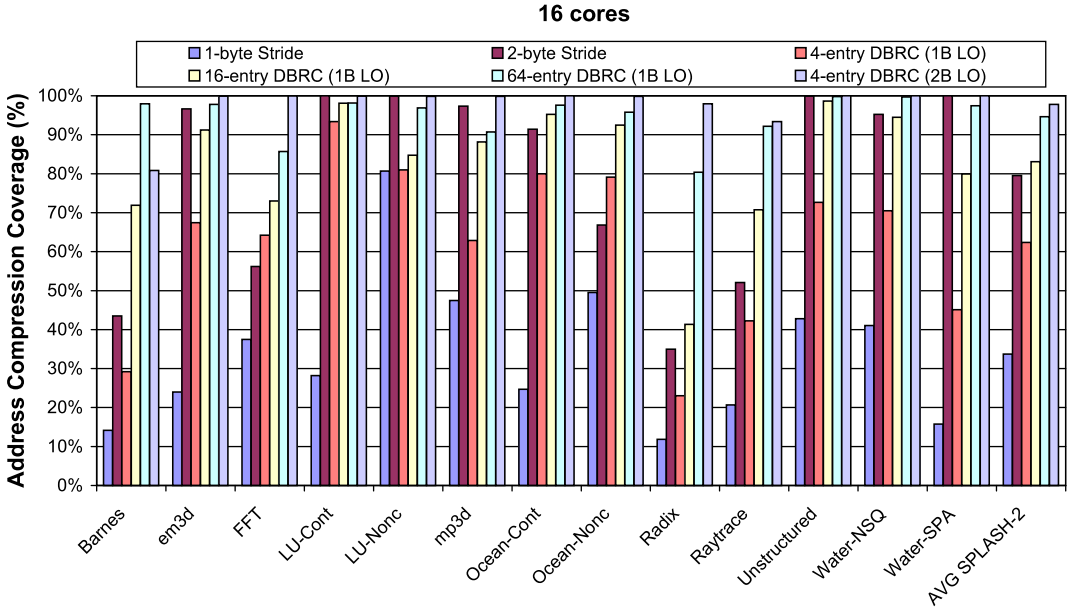

Fig. 3. Address compression coverage for a 16-core tiled CMP.

average address compression coverages of around 98%. These results show that traditional compression schemes provide significant coverage that can be exploited.

It is important to note that this paper is not aimed at proposing a particular compression scheme but at exploiting the area slack due to compression for improving critical messages latency and ensuring efficient message management in CMPs by means of using a heterogeneous interconnect.

In order to evaluate the hardware cost of the address compression schemes for a tiled CMP architecture, Table 1 shows the area and power characteristics of different configurations for these two address compression schemes along with the percentage relative to one of the cores of the CMP. The measurements have been carried out using CACTI 4.1 [25] for a 32 nm process technology. In each core, we need to implement one sending structure and as many receiving structures as the number of cores. Moreover, requests and coherence commands use their own hardware structures to avoid destructive interferences between both address streams. It can be seen that the relative area cost of implementing these compression schemes for the full CMP with respect to a core ranges from 0.1% for the 2-byte Stride scheme to 3.3% for the 64-entry DBRC one (from 0.46% to 4.58% when the dynamic power consumption is considered).

2.3. Wire implementation for heterogeneous interconnects

The delay of a wire can be modeled as a first-order RC circuit [13]. In that model, a CMOS driver is seen as a simple resistor, R_{gate}, with a parasitic load, C_{diff} as shown in Eq. (1). The CMOS receiver at the other end of the wire presents a capacitive load C_{gate}. C_{wire} and R_{wire} are the wire resistance and capacitance, respectively.

$$\text{Delay} \propto R_{gate}(C_{diff} + C_{wire} + C_{gate}) + R_{wire}\left(\frac{1}{2}C_{wire} + C_{gate}\right) \quad (1)$$

The resistance per unit length of the wire, R_{wire}, depends on the geometrical dimensions of the wire cross-section. Increasing the width of the wire can significantly decrease its resistance although a modest increase in C_{wire} is produced. Similarly, increasing the spacing between adjacent wires results in a C_{wire} drop. Combining both factors, we can design wires with lower delays.

Furthermore, the delay of an uninterrupted wire grows quadratically with its length. Therefore, for long wires, designers must insert repeaters periodically along the wire to break this quadratic dependence of wire delay on the wire length. As repeaters divide a long wire into multiple shorter segments of length l, the total wire delay is the number of segments multiplied by the individual segment delay. Each segment's delay is still quadratically dependent on its segment length, but the total wire delay is now linear with total length. Overall wire delay can be minimized by selecting optimal repeater sizes and spacing between repeaters, being a commonly employed technique nowadays.

For a global interconnect of length L, the total power dissipation is

$$P_{line} = nP_{repeater} = n(P_{switching} + P_{leakage}) \quad (2)$$

where $n = L/l$ is the number of repeaters for that line.

The dynamic power dissipated driving the wire segment with activity factor α is

$$P_{switching} = \alpha(s(C_{gate} + C_{diff}) + lC_{wire})fV_{DD}^2 \quad (3)$$

where V_{DD} is the power supply voltage; f is the clock frequency and s is the size of the repeaters.

The average leakage power of a repeater is given by

$$P_{leakage} = V_{DD}I_{leakage} = V_{DD}\frac{1}{2}(I_{off_N}W_{N_{min}} + I_{off_P}W_{P_{min}})s \quad (4)$$

Table 2
Area, delay, and power characteristics of wire implementations for a 32 nm process technology (derived from Ref. [6]).

Wire type	Relative latency	Relative area	Dynamic power (W/m) α = switching factor	Static power (W/m)
B-Wire (8X plane)	1*x*	1*x*	1.69α	3.72
B-Wire (4X plane)	1.6*x*	0.5*x*	1.85α	4.21
L-Wire (8X plane)	0.5*x*	4*x*	0.77α	1.81
PW-Wire (4X plane)	3.2*x*	0.5*x*	0.56α	1.09

where I_{off_N} (I_{off_P}) is the leakage current per unit NMOS (PMOS) transistor width and $W_{N_{min}}$ ($W_{N_{min}}$) is the width of the NMOS (PMOS) transistor in minimum size inverter.

Eqs. (3) and (4) show that the dissipated power can be reduced by employing smaller repeaters and by increasing their spacing. Banerjee et al. [2] developed a methodology to estimate repeater size and spacing that minimizes power consumption for a fixed wire delay.

In summary, by varying some physical properties such as wire width/spacing and repeater size/spacing, we can implement wires with different latency, bandwidth and power properties. As previously mentioned, in [6], the authors apply this observation to develop a heterogeneous interconnect. They propose to use two wire implementations apart from baseline wires (*B-Wires*): power optimized wires (*PW-Wires*) that have fewer and smaller repeaters, and bandwidth optimized wires (*L-Wires*) with higher widths and spacing. Then, coherence messages are mapped to the appropriate set of wires taking into account, among others, their latency and bandwidth requirements.

Table 2 shows the relative delay, area, and power characteristics of *L-* and *PW-Wires* compared to baseline wires (*B-Wires*) when a 32 nm process technology is considered. Data have been derived from the original 65 nm values in [6] using Orion 2.0 [16]. We assume 10 metal layers: four layers in 1X plane, and two layers in each 2X, 4X, and 8X planes [18]. 4X and 8X Metal planes are used for global inter-core wires. It can be seen that *L-Wires* yield a two-fold latency improvement at a fourfold area cost. On the other hand, *PW-Wires* are designed to reduce power consumption with twice the delay of baseline wires (and the same area cost). As in [18], it is assumed that 4X and 8X wires are routed over memory arrays.

3. A proposal for efficient message management in tiled CMPs

In this section we present our proposal for reduced energy consumption in tiled CMPs. As introduced before, there are two main components in our proposal. The first is the use of an address compression scheme that allows for a significant area slack. The second is the use of a heterogeneous interconnect that exploits that area slack for wire latency improvement by using two different set of wires. Messages are sent through the appropriate set, according to their latency and bandwidth requirements. This section starts with a description of the tiled CMP architecture assumed in this paper, followed by a classification of the messages in terms of both their criticality and size and, finally, the description of the proposed mechanism.

3.1. Tiled CMP architectures

A tiled CMP architecture consists of a number of replicated *tiles* connected over a switched direct network (Fig. 4). Each tile contains a processing core with primary caches (both instruction and data caches), a slice of the L2 cache, and a connection to the

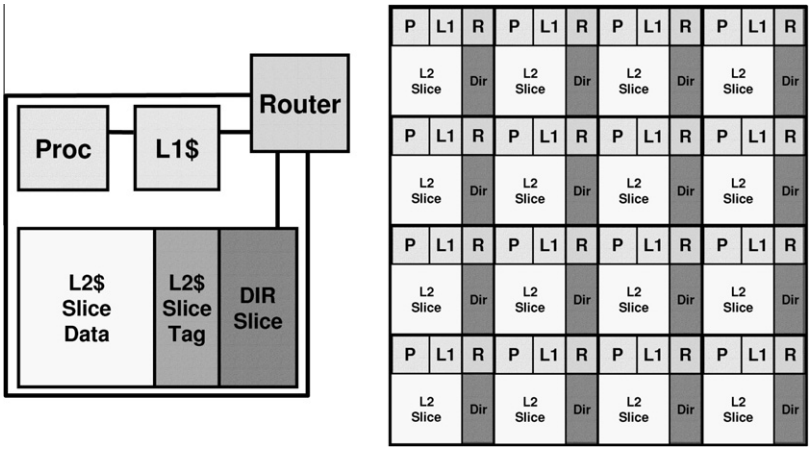

Fig. 4. Tiled CMP architecture overview.

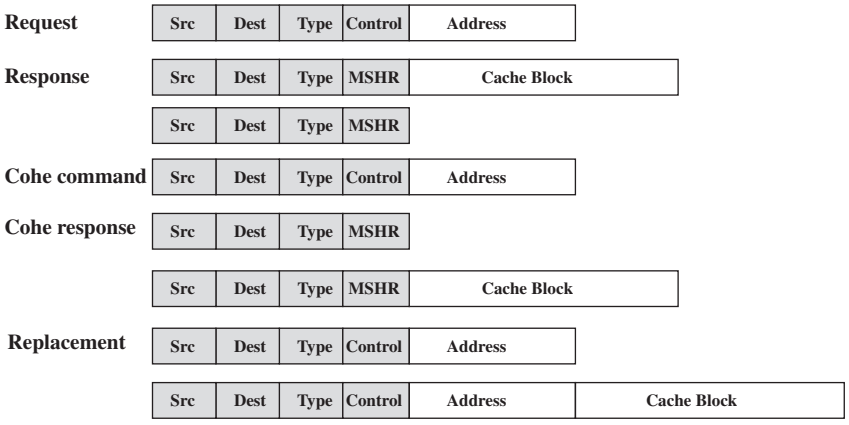

Fig. 5. Classification of messages that travel on the interconnection network of a tiled CMP architecture.

on-chip network. The L2 cache is shared among the different processing cores, but it is physically distributed between them. Therefore, some accesses to the L2 cache will be sent to the local slice while the rest will be serviced by remote slices (L2 NUCA architecture [17]). In addition, the L2 cache stores (in the tags' part of the local L2 slice) the directory information needed to ensure coherence between the L1 caches. On a L1 cache miss, a request is sent down to the appropriate tile where further protocol actions are initiated based on that block's directory state, such as invalidation messages, intervention messages, data writeback, data block transfers, etc. In this paper, we assume a process technology of 32 nm, a tile area of approximately 12 mm^2, and a die size in the order of 200 mm^2 [31,33]. Note that, due to manufacturing costs and form factor limitations, it would be desirable to keep die size as low as possible [33]. Further details about the evaluation methodology and the simulated CMP configuration can be found in Section 4.

There are a variety of message types traveling on the interconnect of a CMP, each one with properties that are clearly distinct. In general, we can classify messages into the following groups (see Fig. 5):

- *Request messages*, that are generated by cache controllers in response to L1 cache misses and are sent to the corresponding home L2 cache to demand privileges over a memory line.
- *Response messages* to these requests, generated by the home L2 cache controller or, alternatively, by the remote L1 cache that has the single valid copy of the data, and that can carry the memory line or not.
- *Coherence commands*, that are sent by the home L2 cache controller to the corresponding L1 caches to ensure coherence.
- *Coherence responses*, sent by the L1 caches back to the corresponding home L2 in response to coherence commands.
- *Replacement messages*, that the L1 caches generate in case of exclusive or modified lines being replaced (replacement hints are not sent for lines in shared state).

3.2. Classification of messages in tiled CMP architectures

Messages involved in the L1 cache coherence protocol (shown in Fig. 5) can be classified according to their criticality into critical and non-critical messages. We say that a message is critical when it is in the critical path of the L1 cache miss. In other case, we call the message as non-critical.

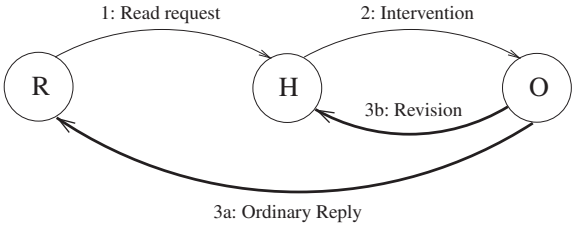

Fig. 6. Protocol actions taken in response to a read request that misses for a block in modified state (messages with data are represented using thicker lines).

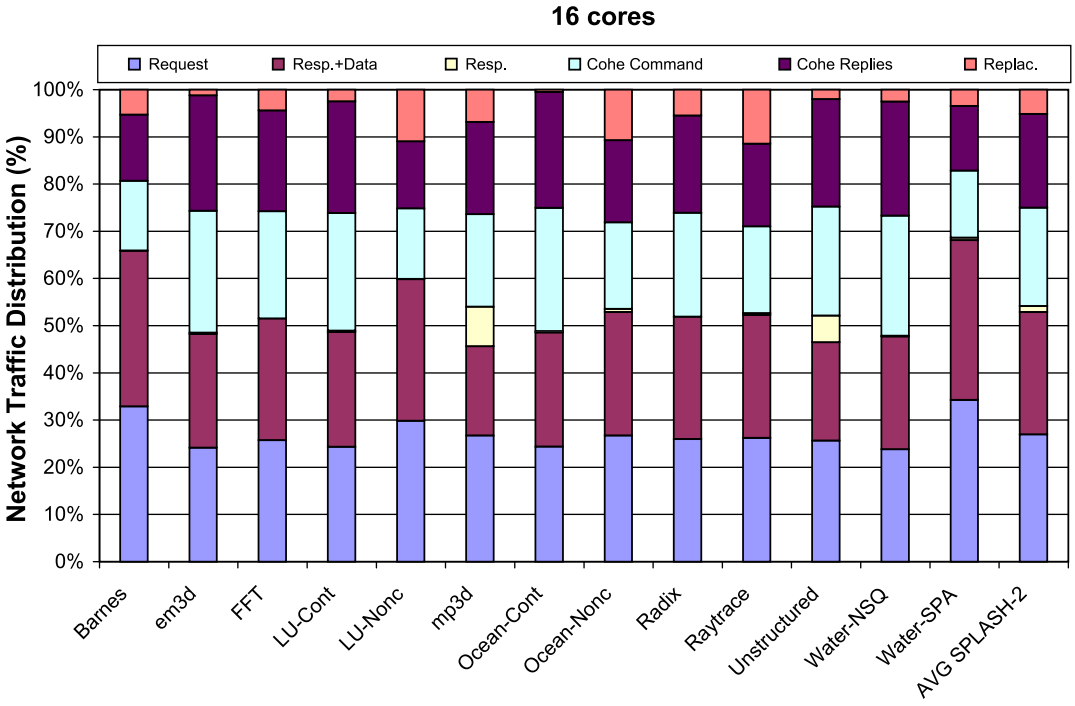

Fig. 7. Breakdown of the messages that travel on the interconnection network for a 16-core CMP.

As an example, Fig. 6 shows the coherence actions involved in a L1 read miss for a line in modified state in other tile. These actions consist of: (1) a request message that is sent down to the appropriate directory tile (the home L2 cache); (2) an intervention message sent to the owner tile that (3a) sends back the line to the requestor and (3b) to the directory tile. Whereas messages (1), (2) and (3a) are critical because they belong to the critical path between the processor request and the memory system response, (3b) is non-critical. Using this criterion, we can observe that all message types but replacement messages and some coherence replies (such as revision messages) are critical. It is clear that performance is increased if critical messages are sent through low-latency *L-Wires* (assuming that there are enough wires).

On the other hand, coherence messages can also be classified according to their size into short and long messages. Coherence responses do not include the address or the data block and just contain control information (source/destination, message type, MSHR id, etc.). Therefore, they are classified as short messages. Other message types, in particular requests, responses without data and coherence commands, also contain address block information but they are still narrow enough to be classified as short messages. They are the focus of our proposal. Finally, replacements with data and data block transfers also carry a cache line and, therefore, they are classified as long messages.

Fig. 7 plots the fraction of each message type on the total number of messages for a 16-core CMP configuration for the applications used in our evaluation (see Section 4.1 for evaluation details). As it can be seen, on average, more than 60% of the messages are related to memory accesses (a request and its corresponding reply), whereas the rest has to do with coherence enforcement (25%) and block replacement (15%). It is important to remember that more than 50% of the messages are short messages containing address block information that can be compressed.

Previous work [6,11] has shown that the use of a heterogeneous interconnect comprised of low-latency *L-Wires* and power-efficient *PW-Wires* allows for more energy-efficient interconnect utilization. However, since the number of *L-Wires* is small because of their fourfold area cost (relative to baseline wires) only short messages can take full advantage of them.

3.3. Interconnect design for efficient message management

In this work, we use the same main parameters for the interconnect as in [6,11]. In particular, message sizes and the width of the original links of the interconnect are the same. Short messages can take up to 11 bytes. Requests, coherence commands are 11-byte long since beside control information (3 bytes) they also carry address information. On the other hand, coherence replies and replacements without data are just 3-byte long. Finally, *Ordinary Reply* messages are 67-byte long since they carry control information (3 bytes) and a cache line (64 bytes).

As discussed in the previous section, *L-Wires* have a fourfold area cost compared to baseline wires and, therefore, the number of *L-Wires* is quite limited. Considering that they will be used for sending short, critical messages, previous proposals have fixed their amount according to the typical size of short messages (e.g., 11 bytes). The remaining area have been employed for sending long messages. However, by using an address compression scheme the amount of *L-Wires* can be reduced dramatically, from 11 bytes to 4–5 bytes depending on the size of the uncompressed low-order bits used by the underlying compression scheme, arising an area slack than can be exploited for further improving wire latency, i.e., having thicker but faster wires. We denote this new kind of wires as *VL-Wires*.

Table 3 shows the relative delay, area, and power characteristics of the new *VL-Wires* for different wire widths compared to baseline wires (*B-Wires*) when a single metal plane is considered (8X plane). In order to match the metal area of the baseline configuration, in our proposal, each original 75-byte unidirectional link is designed to be made up of 24–40 *VL-Wires* (3–5 bytes) with different relative latencies and area cost, as shown in Table 3, and 272 *B-Wires* (34 bytes).[1] *VL-Wires* will be used for sending already short, critical

[1] Note that the heterogeneous interconnect (*VL-Wires* + *B-Wires*) always matches the metal area of the baseline interconnect (75-byte or 600 *B-Wires*).

Table 3
Relative delay, area, and power characteristics of *VL-Wires* (8X plane) related to baseline wires (*B-Wires*) for different widths.

Wire width	Relative latency	Relative area	Dynamic power (W/m) α = switching factor	Static power (W/m)
3 Bytes	0.27*x*	14*x*	0.459α	0.978
4 Bytes	0.31*x*	10*x*	0.527α	1.248
5 Bytes	0.35*x*	8*x*	0.596α	1.403

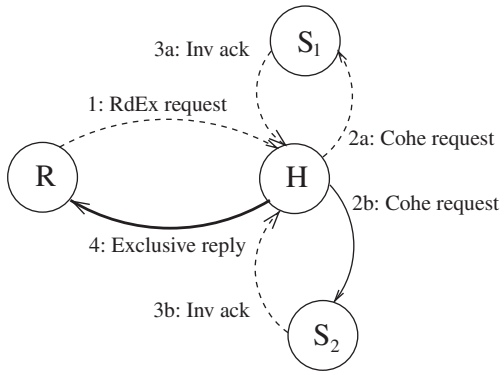

Fig. 8. New behavior for the protocol actions taken in response to a write request for a block in shared state (messages with data are represented using thicker lines). The use of *VL-Wires* is represented by dashed lines.

messages (e.g., coherence replies) as well as *compressed* requests and *compressed* coherence commands. Uncompressed and long messages are sent using the original *B-Wires*. For a discussion regarding the implementation complexity of heterogeneous interconnects refer to [6].

Fig. 8 shows the new behavior when our proposal is used. It shows the coherence actions involved in a L1 write miss for a line in shared state in two tiles. These actions consist of: (1) a request message that is sent down to the appropriate directory tile (the home L2 cache); (2a, 2b) two coherence messages sent to the sharer tiles that send acknowledgments back (3a, 3b) to the directory tile; and (4) a reply message to the requestor tile. Acknowledgment messages (3a) and (3b) are 3-byte long and can always be sent using the new *VL-Wires*. The reply message (4) is carrying a data line, so it will be sent through the original *B-Wires*. The request and coherence messages are short messages containing address block information that can be compressed. Assuming an average address compression coverage over 80% we can consider that, on average, two of the messages will be compressed and sent using the new *VL-Wires*. In this example, the request (1) and the first coherence message (2a) are compressed.

Our proposal introduces additional complexity in the routing logic. When using a heterogeneous interconnect comprised of two kind of wires: *VL-Wires* and *B-Wires*, two different buffers are required at each port to store *VL-* and *B*-messages separately as opposed to a single larger buffer employed in the base case. The size of each buffer is proportional to the flit size of the corresponding set of wires. For example, a set of 32 *VL-Wires* (4 bytes) employs a message buffer with a word size of 32 bits. For a further analysis about the implications of using heterogeneous networks in the on-chip interconnect of a tiled CMP architecture, please refer to [6].

4. Experimental results

This section shows the results that are obtained for our proposal under different scenarios and compares them against those achieved with the configuration that employs just *B-Wires*, which is taken as baseline. For comparison purposes, we also show the behavior assuming perfect address compression, which is represented in the figures using lines instead of barlines.

4.1. Evaluation methodology

The results presented in this work have been obtained through detailed simulations of a full CMP. We have employed a cycle-accurate CMP power-performance simulation tool, *Sim-PowerCMP* [12], that estimates both dynamic and leakage power and is based on RSIM [14]. In particular, *Sim-PowerCMP* employs as performance simulator a modified version of RSIM that models the architecture of the tiled CMP presented in Section 3. *Sim-PowerCMP* also implements already proposed and validated power models for both dynamic power (from Wattch [5], CACTI [25]) and leakage power (from HotLeakage [32]) of each processing core, as well as the interconnection network (from Orion [28,16]).

Table 4 (top) shows the architecture configuration used across this paper. It describes a 16-core CMP built in 32 nm technology. The tile area has been fixed to 12 mm^2, including a portion of the second-level cache [31]. With this configuration, links that interconnect routers configuring the 2D mesh topology measure around 2.5 mm. Reply messages are 67-byte long since they carry both control information (3 bytes) and a cache line (64 bytes). On the contrary, request, coherence and coherence reply messages that do not contain data are, at most, 11-byte long (3 bytes for header, 8 bytes for the memory address), just 3-byte long for coherence replies. Table 4 (bottom) shows the applications used in our experiments. *MP3D* is from the SPLASH benchmark suite; *Barnes-Hut*, *FFT*, *LU-cont*, *LU-noncont*, *Ocean-cont*, *Ocean-noncont*, *Radix*, *Raytrace* and *Water-nsq* are from the SPLASH-2 benchmark suite; Berkeley *EM3D* simulates the propagation of electro-magnetic waves through objects in three dimensions; and *Unstructured* is a computational fluid dynamics application that uses an Unstructured mesh. Problem sizes have been chosen commensurate with the size of the L1 caches and the number of cores used in our simula-

Table 4
Configuration of the evaluated baseline CMP architecture and applications.

CMP configuration	
Parameter	
Process technology	32 nm
Tile area	12 mm^2
Number of tiles	16
Cache line size	64 Bytes
Core	4 GHz, in-order 2-way model
L1 I/D-cache	32 kB, 4-way
L2 cache (per core)	256 kB, 4-way, 6 + 2 cycles
Memory access time	400 cycles
Network configuration	2D mesh
Network bandwidth	75 GB/s
Link width	75 bytes (*8X-B-Wires*)
Link length	2.5 mm
Application	Problem size
Barnes-Hut	16K bodies, 4 timesteps
EM3D	9600 nodes, 5% remote links, 4 timesteps
FFT	256K complex doubles
LU-cont	256 $\times$ 256, *B* = 8
LU-noncont	256 $\times$ 256, *B* = 8
MP3D	50,000 nodes, 2 timesteps
Ocean-cont	258 $\times$ 258 grid
Ocean-noncont	258 $\times$ 258 grid
Radix	2M keys
Raytrace	car.env
Unstructured	mesh.2K, 5 timesteps
Water-nsq	512 Molecules, 4 timesteps
Water-spa	512 Molecules, 4 timesteps

tions, following the recommendations given in [30]. All experimental results reported in this work are for the parallel phase of these applications.

4.2. Simulation results and analysis

In this section we analyze, first, the impact of our proposal on the execution time and on the energy-delay^2 product metric for the inter-core links. Then, we evaluate the energy-delay^2 product metric for the full CMP. All results have been normalized with respect to the baseline configuration where only *B-Wire*, unidirectional 75-byte wide links are considered and they include the additional cost due to the extra hardware structures needed to implement the different address compression schemes evaluated.

Fig. 9 (top) depicts the normalized execution time with respect to that obtained for the baseline configuration for a 16-core CMP. Barlines show the normalized execution time for several Stride and DBRC address compression schemes, in particular those with a compression coverage over 80% as reported in Fig. 3. The number of bytes used to send the low-order bits (1 or 2 bytes) determines the number of *VL-Wires* in the heterogeneous network (4 or 5 bytes). For comparison purposes, and in order to show the potential improvement in the execution time, three additional solid lines have been added to show the execution time for the different heterogeneous network configurations when a perfect compression coverage is considered. It can be observed that a 4-entry DBRC compression scheme with 2 low-order bytes is enough to achieve an average performance improvement of 8% (close to 10% of

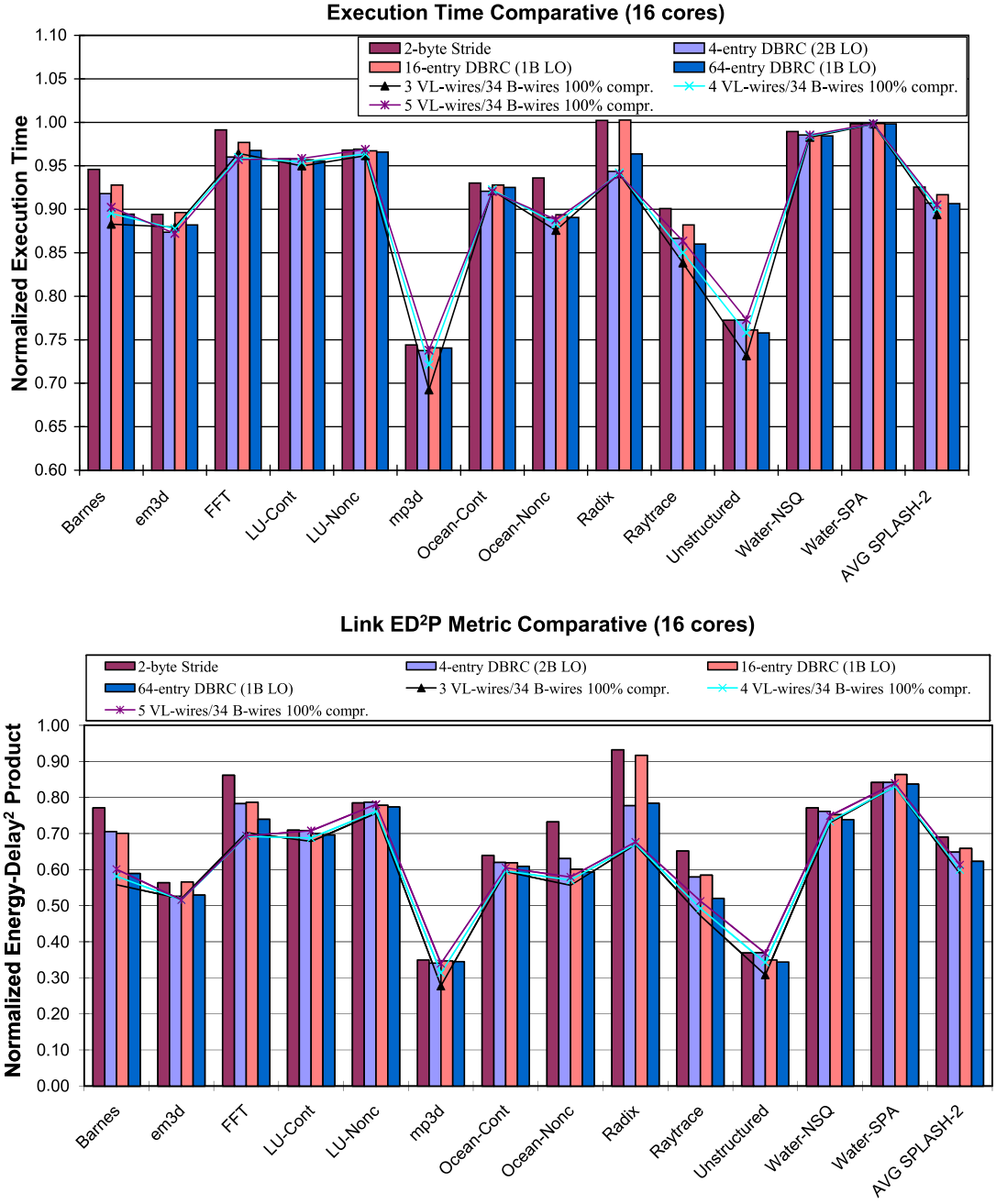

Fig. 9. Normalized execution time (top) and link ED^2P metric (bottom) for different address compression schemes over heterogeneous links (*VL-Wires* width matches address compression size).

maximum potential performance improvement). This improvement has high variability, ranging from almost 1–2% for Water and LU to 22–25% for mp3d and Unstructured. This variability is due to two factors. First, some applications, as Water or LU, present low inter-core data sharing patterns [30]. In these cases, the coherence traffic is small and our proposal has little impact in the execution time. Other applications, such as mp3d or Unstructured, present better traffic patterns and can take advantage of a faster interconnect. The second factor that explains this variability is related with the address compression scheme coverage. As it was shown in Fig. 3, applications such as Barnes or Radix exhibit low address compression coverage for most of the proposed configurations. For these applications, the reduction in the execution time does not match the maximum potential even when a 64-entry configuration is used.

Fig. 9 (bottom) plots the normalized energy-delay^2 product (ED^2P) metric. Average reductions close to 40% are obtained, although again, a high variability among applications is observed. Some applications, such as Water and LU, show reductions of around 20% due mainly to the lower power dissipation of the proposed heterogeneous interconnection network; others, such as mp3d and Unstructured, present a reduction of 65% in the ED^2P metric due to bigger emphasis on the execution time that the ED^2P metric does.

On the other hand, Fig. 10 shows the normalized execution time (top) and ED^2P metric (bottom) when the number of *VL-Wires* is restricted to 3 bytes, the minimum size of the messages flowing through the interconnect, independently of the address compression scheme used. In this case, the compressed messages are sent through *VL-Wires* using fragmentation. This configuration allows

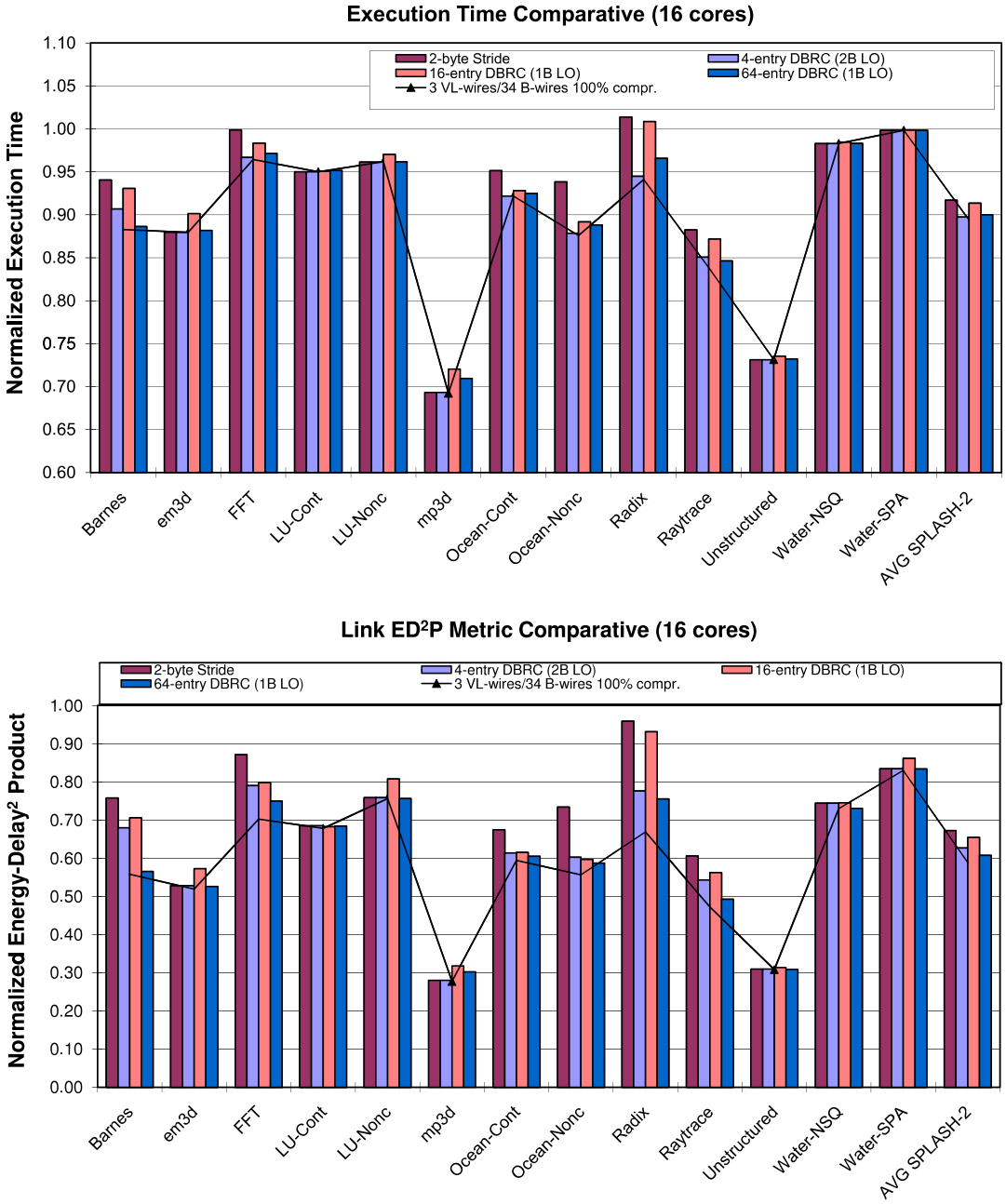

Fig. 10. Normalized execution time (top) and ED^2P metric (bottom) for different address compression schemes over heterogeneous links with fixed 3-byte *VL-Wires* width and message fragmentation.

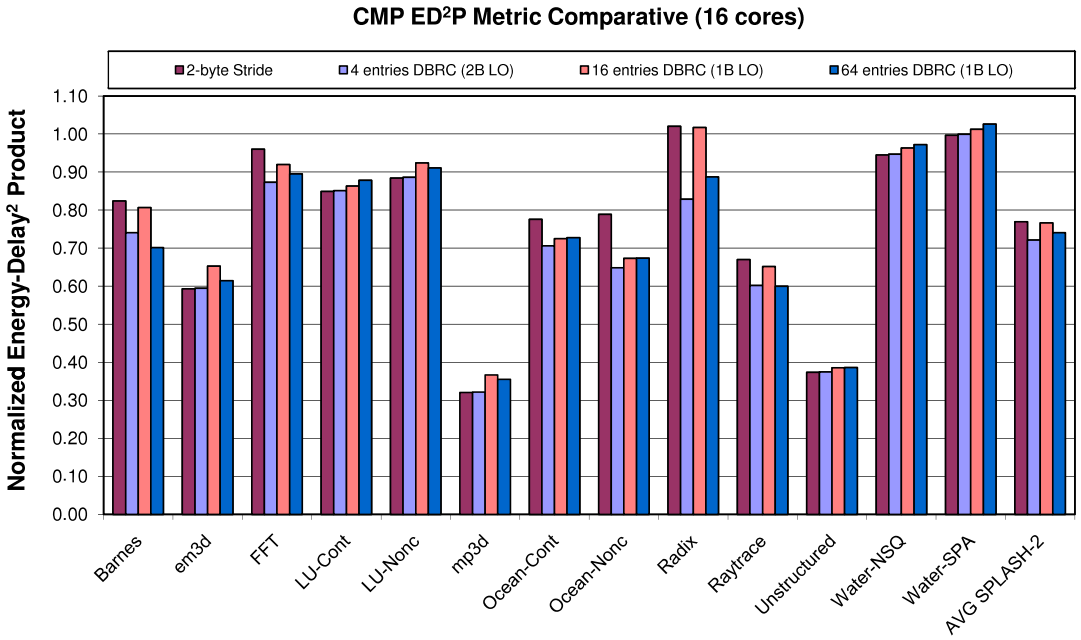

Fig. 11. Normalized energy-delay^2 product (ED^2P) for the full CMP (3-byte *VL-Wires* configuration).

for a minimum latency for responses without data and coherence responses (3-byte long) with little penalty for compressed messages (4- or 5-byte long). In this case, an additional reduction in both execution time and the ED^2P metric is observed. For the 4-entry DBRC compression scheme with 2 low-order bytes, an average performance improvement of 10% is obtained (two additional points with respect to the results obtained when *VL-Wires* width is designed depending on the address compression size). When the ED^2P metric is considered, average reductions over 35% are obtained for the 4-entry DBRC compression scheme going up to 38% for the 64-entry configuration.

Finally, Fig. 11 presents the normalized ED^2P metric for the full CMP assuming the 3-byte *VL-Wires* configuration. Average improvements range from 23% for the 2-byte Stride configuration to 28% for the 4-entry DBRC one. As it can be seen, when the number of entries of the DBRC compression scheme is increased, worse ED^2P metrics are obtained for the full CMP. This is due to the bigger impact that the extra hardware structures have, which is not compensated with a significant reduction in the execution time.

5. Sensitivity analysis

In this section, we discuss the impact of using out-of-order processors cores, link bandwidth and relative latency between the second-level cache and the interconnect on our proposed address compression technique with a heterogeneous interconnect.

5.1. Out-of-order/in-order processors

Along this paper we have considered in-order processing cores. In this section we evaluate the effect of using out-of-order (OoO) cores in order to show how even with this kind of cores, which allow for potential overlapping of several cache misses, our proposal still shows important performance improvements. Note that the use of OoO cores in future many-core CMP designs seems unlikely due to the increased area and energy consumption that this kind of processor cores would entail.

We have configured *SimPower-CMP* to model a CMP with OoO cores and the same parameters reported in Table 4. Non-blocking first level caches have been considered, which allows overlapping cache some cache misses, and a ROB with 64 entries is used. Fig. 12 shows the performance speedup of our proposal over an out-of-order baseline configuration for a 16-core CMP. All benchmarks except Ocean-cont show degrees of performance improvement that range from 25% to 27% for mp3d and Unstructured to almost negligible for Water-nsq. Although still noticable, the average reduction in the execution time is lower than what we observe in a CMP using in-order cores (an improvement of 7% versus a 9% when in-order cores were considered). This behavior is due to the greater tolerance to long instruction latencies that an out-of-order processor has.

5.2. Link bandwidth

As discussed in Section 2.3, *L-Wires* have a fourfold area cost compared to baseline wires and, therefore, the number of *L-Wires* is quite limited. In a bandwidth-constrained system this restriction is exacerbated and, therefore, our proposal is likely not to perform very well. To verify this, we consider the same base case that Cheng et al. propose in [6] where every link has only 80 *B-Wires* at the 8X-metal layer. As in [6], we consider twice the metal area of the new baseline interconnect to implement a heterogeneous interconnect where links are designed to be made up of 24 *L-Wires* (3 bytes) and 64 *B-Wires* (8 bytes). Benchmarks such as Water, Barnes, Unstructured or LU that exhibit lower network utilizations still show marginal performance improvements. The rest of the benchmarks suffer significant performance losses. Overall, the heterogeneous model performed 15% worse than the base case.

5.3. Relative latency of the interconnect with respect to the L2 cache

Across this paper we have considered a more up-to-date access time to the L2 cache slices (6 + 2 cycles) that in previous works [6,11] where L2 caches present higher latencies (10 + 20 cycles).On average, our proposal performs 3% better when a more up-to-date access time is considered (6 + 2 cycles configuration). An improvement of 6% versus 9% is obtained when L2 caches slices that present higher latencies (10 + 20 cycles) are considered.

Fig. 12. Normalized execution time for different address compression schemes over heterogeneous links when OoO cores are used for a 16-core CMP.

For most of the applications the use of L2 caches with higher latencies has little impact on performance (performance improvement is 0–2% worse than the one obtained with the initial cache configuration). For some applications for which the execution time is highly dependent on the memory hierarchy latencies (with increases in execution time that ranges from 10% to 15% for Raytrace or Unstructured to almost 30% for mp3d), our proposal suffers significant degradation (over 10% worse) for the 10 + 20 cycles configuration. In these applications the bottleneck is not the interconnection network but the access time to the L2 cache slices.

6. Conclusions

In this work we have proposed an energy and performance aware message management mechanism for tiled CMPs that consists of two components. The first one is the use of an address compression scheme that allows for a significant area slack. The second approach is a heterogeneous interconnection network that exploits the arising area slack for improving wire latency and that is comprised of only two different types of wires: *VL-Wires* for critical short messages and baseline wires for the rest of messages.

Results obtained through detailed simulations of a 16-core CMP show that the proposed on-chip message management mechanism can reduce the ED^2P metric for the links of the interconnection network about 38% with an additional reduction in execution time of 10%. Finally, these reductions translate into overall CMP savings of 28% when the ED^2P metric is considered.

The sensitivity analysis shows that our proposal performs better when tiles are implemented using in-order processors (an improvement in the execution time of 7% versus a 9% is obtained when out-of-order processors were considered). This behavior is due to the greater tolerance to long load/store latencies that an out-of-order processor has. Moreover, the relative latency between the second-level cache and the interconnect has little impact in the performance of our proposal (an improvement of 6% versus a 9%). Finally, when a bandwidth-constrained system is considered, the fourfold area cost of the *L-Wires* becomes a problem and, on average, the heterogeneous model performs 15% worse than the base case.

All these results reveal that correctly organizing the interconnection network and properly managing the different types of messages through it have significant impact on the energy consumed by CMPs, especially for next-generation dense CMP architectures.

Acknowledgments

This work has been jointly supported by the Spanish MEC and European Commission FEDER funds under grants "Consolider Ingenio-2010 CSD2006-00046" and "TIN2006-15516-C4-03", and also by the Fundación Séneca (Agencia Regional de Ciencia y Tecnología, Región de Murcia) under grant 05831/PI/07.

References

[1] R. Balasubramonian, N. Muralimanohar, K. Ramani, V. Venkatachalapathy, Microarchitectural wire management for performance and power in partitioned architectures, in: Proceedings of the 11th International Symposium on High-Performance Computer Architecture (HPCA-11), IEEE Computer Society, San Francisco, CA, USA, 2005, pp. 28–39.
[2] K. Banerjee, A. Mehrotra, A power-optimal repeater insertion methodology for global interconnects in nanometer designs, IEEE Transactions on Electron Devices 49 (11) (2002) 2001–2007.
[3] K. Basu, A.N. Choudhary, J. Pisharath, M.T. Kandemir, Power protocol: reducing power dissipation on off-chip data buses, in: Proceedings of the 35th International Symposium on Microarchitecture (MICRO-35), IEEE Computer Society, San Diego, CA, USA, 2002, pp. 345–355.
[4] B.M. Beckmann, D.A. Wood, TLC: Transmission Line Caches, in: Proceedings of the 36th International Symposium on Microarchitecture (MICRO-36), IEEE Computer Society, San Diego, CA, USA, 2003, pp. 43–54.
[5] D. Brooks, V. Tiwari, M. Martonosi, Wattch: a framework for architectural-level power analysis and optimizations, in: Proceedings of the 27th International Symposium on Computer Architecture (ISCA-27), ACM Press, Vancouver, Canada, 2000, pp. 83–94.
[6] L. Cheng, N. Muralimanohar, K. Ramani, R. Balasubramonian, J. Carter, Interconnect-aware coherence protocols for chip multiprocessors, in: Proceedings of the 33rd International Symposium on Computer Architecture (ISCA-33), Boston, MA, USA, 2006, pp. 339–351.
[7] D. Citron, Exploiting low entropy to reduce wire delay, Computer Architecture Letters 3 (2004) 1.
[8] R. Das, A.K. Mishra, C. Nicopoulos, D. Park, V. Narayanan, R. Iyer, M.S. Yousif, C.R. Das, Performance and power optimization through data compression in network-on-chip architectures, in: Proceedings of the 14th International Conference on High-Performance Computer Architecture (HPCA'08), 2008, pp. 215–225.

[9] M. Farrens, A. Park, Dynamic base register caching: a technique for reducing address bus width, Proceedings of the 18th International Symposium on Computer Architecture (ISCA), vol. 19, ACM Press, New York, NY, 1991, pp. 128–137.
[10] A. Flores, M.E. Acacio, J.L. Aragón, Address compression and heterogeneous interconnects for energy-efficient high-performance in tiled cmps, in: International Conference on Parallel Processing, IEEE Computer Society, Los Alamitos, CA, USA, 2008, pp. 295–303.
[11] A. Flores, J.L. Aragón, M.E. Acacio, Efficient message management in tiled cmp architectures using a heterogeneous interconnection network, in: Proceedings of the 14th International Conference on High Performance Computing (HiPC'07), Lecture Notes in Computer Science, vol. 4873, Springer, 2007, pp. 133–146.
[12] A. Flores, J.L. Aragón, M.E. Acacio, An energy consumption characterization of on-chip interconnection networks for tiled cmp architectures, Journal of Supercomputing 45 (3) (2008) 341–364.
[13] R. Ho, K. Mai, M. Horowitz, The future of wires, Proceedings of the IEEE 89 (4) (2001) 490–504.
[14] C.J. Hughes, V.S. Pai, P. Ranganathan, S.V. Adve, RSIM: simulating shared-memory multiprocessors with ilp processors, IEEE Computer 35 (2) (2002) 40–49.
[15] Y. Jin, K.H. Yum, E.J. Kim, Adaptive data compression for high-performance low-power on-chip networks, in: Proceedings of the 41st International Symposium on Microarchitecture (MICRO'08), 2008, pp. 354–363.
[16] A.B. Kahng, B. Li, L.-S. Peh, K. Samadi, Orion 2.0: a fast and accurate noc power and area model for early-stage design space exploration, in: Proceedings of the Design Automation and Test in Europe (DATE'09), 2009, pp. 423–428.
[17] C. Kim, D. Burger, S.W. Keckler, An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches, in: Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-10), ACM Press, San Jose, CA, USA, 2002, pp. 211–222.
[18] R. Kumar, V. Zyuban, D.M. Tullsen, Interconnections in multi-core architectures: understanding mechanisms, overheads and scaling, in: Proceedings of the 32nd International Symposium on Computer Architecture (ISCA-32), IEEE Computer Society, Madison, WI, USA, 2005, pp. 408–419.
[19] C. Liu, A. Sivasubramaniam, M. Kandemir, Optimizing bus energy consumption of on-chip multiprocessors using frequent values, Journal of Systems Architecture 52 (2) (2006) 129–142.
[20] J. Liu, K. Sundaresan, N. Mahapatra, Fast, performance-optimized partial match address compression for low-latency on-chip address buses, in: 24th International Conference on Computer Design (ICCD'06), 2006, pp. 17–24.
[21] N. Magen, A. Kolodny, U. Weiser, N. Shamir, Interconnect-power dissipation in a microprocessor, in: Proceedings of the 6th International Workshop on System Level Interconnect Prediction (SLIP-6), ACM Press, Paris, France, 2004, pp. 7–13.
[22] N. Muralimanohar, R. Balasubramonian, The effect of interconnect design on the performance of large L2 caches, in: 3rd IBM Watson Conference on Interaction between Architecture, Circuits, and Compilers (P = ac2), Yorktown Heights, 2006.
[23] J.-M. Parcerisa, A. Gonzalez, Reducing wire delay penalty through value prediction, in: MICRO 33: Proceedings of the 33rd International Symposium on Microarchitecture, ACM Press, 2000, pp. 317–326.
[24] Y. Sazeides, J.E. Smith, The predictability of data values, in: MICRO 30: Proceedings of the 30th International Symposium on Microarchitecture, IEEE Computer Society, 1997, pp. 248–258.
[25] D. Tarjan, S. Thoziyoor, N.P. Jouppi, Cacti 4.0, Tech. Rep., HP Laboratories, Palo Alto, 2006.
[26] M.B. Taylor, W. Lee, J. Miller, D. Wentzlaff, I. Bratt, B. Greenwald, H. Hoffmann, P. Johnson, J. Kim, J. Psota, A. Saraf, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, A. Agarwal, Evaluation of the raw microprocessor: an exposed-wire-delay architecture for ILP and streams, in: Proceedings of the International Symposium on Computer Architecture, Munchen, Germany (ISCA-31), 2004, pp. 2–13.
[27] S. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, P. Iyer, A. Singh, T. Jacob, S. Jain, S. Venkataraman, Y. Hoskote, N. Borkar, An 80-tile 1.28 tflops network-on-chip in 65 nm cmos, in: Solid-State Circuits Conference (ISSCC'07), 2007, pp. 98–99.
[28] H.-S. Wang, X. Zhu, L.-S. Peh, S. Malik, Orion: a power-performance simulator for interconnection networks, in: Proceedings of the 35th International Symposium on Microarchitecture (MICRO-35), IEEE Computer Society, Istanbul, Turkey, 2002, pp. 294–305.
[29] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J.F. Brown III, A. Agarwal, On-chip interconnection architecture of the tile processor, IEEE Micro 27 (5) (2007) 15–31.
[30] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, A. Gupta, The SPLASH-2 programs: characterization and methodological considerations, in: Proceedings of the 22nd International Symposium on Computer Architecture (ISCA-22), IEEE Computer Society, Los Alamitos, CA, USA, 1995, pp. 24–36.
[31] M. Zhang, K. Asanovic, Victim replication: maximizing capacity while hiding wire delay in tiled chip multiprocessors, in: Proceedings of the 32nd International Symposium on Computer Architecture (ISCA-32), IEEE Computer Society, Madison, WI, USA, 2005, pp. 336–345.
[32] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, M. Stan, HotLeakage: A Temperature-Aware Model of Subthreshold and Gate Leakage for Architects, Tech. Rep., University of Virginia, 2003.
[33] L. Zhao, R. Iyer, S. Makineni, J. Moses, R. Illikkal, D. Newell, Performance, area and bandwidth implications on large-scale CMP cache design, in: Proceedings of the 1st Workshop on Chip Multiprocessor Memory Systems and Interconnects (CMP-MSI'07). In Conjunction with (HPCA-13), IEEE Computer Society, Phoenix, AZ, USA, 2007.
[34] P. Zhou, B. Zhao, Y. Du, Y. Xu, Y. Zhang, J. Yang, L. Zhao, Frequent value compression in packet-based noc architectures, in: Proceedings of the 2009 Asia and South Pacific Design Automation Conference (ASP-DAC'09), IEEE Press, Piscataway, NJ, USA, 2009, pp. 13–18.

Antonio Flores received his M.S. degree in Computer Science in 1994 from the Universidad de Murcia, Spain. He is now a Ph.D. student and Assistant Professor in the Computer Engineering Department at the Universidad de Murcia. His research interests include CMP architectures, processor microarchitecture, and power-aware cache-coherence protocol design.

Manuel E. Acacio received the M.S. and Ph.D. degrees in computer science from the Universidad de Murcia, Spain, in 1998 and 2003, respectively. He joined the Computer Engineering Department at the Universidad de Murcia, in 1998, where he is currently an Associate Professor of computer architecture and technology. His research interests include prediction and speculation in multiprocessor-memory systems, hardware transactional memory, multiprocessor-on-a-chip architectures, and power-aware and fault-tolerant cache-coherence protocol design.

Juan L. Aragón received his M.S. degree in Computer Science in 1996 and his Ph.D. degree in Computer Engineering in 2003, both from the Universidad de Murcia, Spain, followed by a 1-year postdoctoral stay as a Visiting Assistant Professor and Researcher in the Computer Science Department at the University of California, Irvine. In 1999 he joined the Computer Engineering Department at the Universidad de Murcia, where he currently is an Associate Professor. His research interests are focused on CMP architectures, processor microarchitecture, and energy-efficient and reliable systems. He is a member of the IEEE Computer Society.