Heterogeneous Interconnects for Energy-Efficient Message Management in CMPs

Antonio Flores, Juan L. Aragón, *Member*, *IEEE Computer Society*, and Manuel E. Acacio

Abstract—Continuous improvements in integration scale have made major microprocessor vendors to move to designs that integrate several processing cores on the same chip. Chip multiprocessors (CMPs) constitute a good alternative to traditional monolithic designs for several reasons, among others, better levels of performance, scalability, and performance/energy ratio. On the other hand, higher clock frequencies and the increasing transistor density have revealed power dissipation and temperature as critical design issues in current and future architectures. Previous studies have shown that the interconnection network of a Chip Multiprocessor (CMP) has significant impact on both overall performance and energy consumption. 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, from the point of view of both performance and energy, in tiled CMPs using a heterogeneous interconnect. Our proposal consists of two approaches. The first is *Reply Partitioning*, a technique that splits replies with data into a short *Partial Reply* message that carries a subblock of the cache line that includes the word requested by the processor plus an *Ordinary Reply* with the full cache line. This technique allows all messages used to ensure coherence between the L1 caches of a CMP to be classified into two groups: critical and short, and noncritical and long. The second approach is the use of a heterogeneous interconnection network composed of low-latency wires for critical messages and low-energy wires for noncritical ones. Detailed simulations of 8 and 16-core CMPs show that our proposal obtains average savings of 7 percent in execution time and 70 percent in the Energy-Delay squared Product (ED^2P) metric of the interconnect over previous works (from 24 to 30 percent average ED^2P improvement for the full CMP). Additionally, the sensitivity analysis shows that although the execution time is minimized for subblocks of 16 bytes, the best choice from the point of view of the ED^2P metric is the 4-byte subblock configuration with an additional improvement of 2 percent over the 16-byte one for the ED^2P metric of the full CMP.

Index Terms—Tiled chip multiprocessor, energy-efficient architectures, cache coherence protocol, heterogeneous on-chip interconnection network, parallel scientific applications.

1 Introduction

High-performance processor designs have evolved toward architectures that implement multiple processing cores on a single die (CMPs) as implementation technology scales down. On the other hand, tiled architectures provide a scalable solution for supporting families of products with varying computational power, managing the design complexity, and effectively using the resources available in advanced VLSI technologies. Therefore, it is expected that future CMPs will be designed as arrays of replicated tiles connected over an on-chip switched direct network [1], [2].

In CMP architectures, the design of the on-chip interconnection network has been shown to have a significant impact on overall system performance and energy consumption, since it is implemented using global wires that exhibit long delays and high capacitance properties. Wang et al. [3] reported that the on-chip network of the Raw processor consumes 36 percent of total chip power. Magen et al. [4] also attribute 50 percent of overall chip power to the interconnect. Finally, in a previous work [5], we showed that, depending on the number of processing cores, the contribution of the interconnection network to the total CMP power ranges from 15 percent (8-core tiled CMP) to 30 percent (16-core tiled CMP), on average, with some applications reaching up to 50 percent. Additionally, we found that most of this power is dissipated in the point-to-point links of the interconnect (about 70 percent) [6]. Similar results have been reported also in [7], [8].

It is important to note that it is possible to design wires with varying latency and bandwidth properties by tuning wire's characteristics such as wire width and spacing. Similarly, it is possible to design wires with varying latency and energy properties by tuning repeater size and spacing [9]. Using links that are composed of wires with different physical properties, a heterogeneous on-chip interconnection network is obtained. Cheng et al. [10] show that with such a heterogeneous interconnect, a reduction in both execution time and energy consumption is obtained for a CMP with a two-level tree interconnect topology. Unfortunately, they report insignificant performance improvements when direct network topologies are employed in tiled CMPs.

In this work, we present a proposal for efficient message management, from the point of view of both performance and energy, in tiled CMPs. Our proposal consists of two approaches. The first one is *Reply Partitioning*, a technique that allows all messages used to ensure coherence between

• *The authors are with the Facultad de Informática, University of Murcia, Campus de Espinardo s/n, 30100 Espinardo (Murcia), Spain. E-mail: aflores@um.es, {jlaragon, meacacio}@ditec.um.es.*

Manuscript received 30 Sept. 2008; revised 2 Feb. 2009; accepted 23 Feb. 2009; published online 3 Sept. 2009.
Recommended for acceptance by R. Marculescu.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-2008-09-0486.
Digital Object Identifier no. 10.1109/TC.2009.129.

the L1 caches of a CMP to be classified into two groups: 1) critical and short and 2) noncritical and long messages. The second approach is the use of a heterogeneous interconnection network composed of only two types of wires: low-latency wires for critical messages and low-energy wires for noncritical ones.

The main contribution of our proposal is the partitioning of reply messages that carry data into a short critical message containing the subblock of the cache requested by the core as well as a long noncritical message with the whole cache line. This partitioning allows for a more energy-efficient use of the heterogeneous interconnect since now *all* short messages have been made critical whereas *all* long messages have been made noncritical. The former can be sent through the low-latency wires whereas the latter can be sent through the power-efficient wires. Different from the proposals in [10], our proposed partitioning approach first eliminates a complex logic for choosing the correct set of wires (just based on one bit in the message length field instead of checking the directory state or the congestion level of the network), and second, it obtains significant energy-delay improvements when using direct topologies. Additionally, our proposed approach allows for a more balanced workload across the heterogeneous interconnects. Detailed simulations of 8 and 16-core CMP processors show average savings of 7 percent in execution time and 70 percent in the Energy-Delay squared Product (ED^2P) metric of the interconnect (from 24 to 30 percent when the full CMP is considered). Finally, our sensitivity analysis shows that although the execution time is minimized for subblocks of 16 bytes, the best choice from the point of view of the ED^2P metric is the 4-byte subblock configuration with an additional improvement of 2 percent over the 16-byte one for the ED^2P metric of the full CMP.

We presented a preliminary version of this paper in [11]. The major contributions we present in this extended work are the following:

- A new section that reviews wire implementation and the design of a heterogeneous interconnect has been added.
- The section that explains our proposal for efficient message management in tiled CMPs has been extended to evaluate the utilization of cache lines and the available time gap for decoupling data messages. In this way, a better understanding of the benefits of our proposal is obtained.
- A new sensitivity analysis section has been added in order to evaluate the effect of our proposal when considering out-of-order processing cores, narrower links, the relative latency of the interconnect with respect to the second-level cache, and finally, different sizes for the data that partial replies carry on.
- A more up-to-date configuration of the CMP is evaluated. We have changed the L2 access time in order to model a faster distributed L2 cache than the one evaluated in [10], [11].

The rest of the paper is organized as follows: Section 2 provides some background on the techniques that enable different wire implementations and the design of a heterogeneous interconnect, and reviews some related work. Our proposal for efficient message management 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.

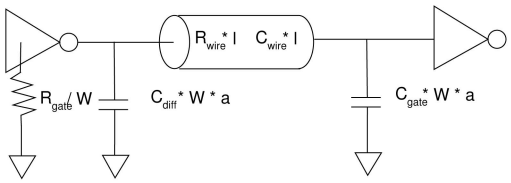

Fig. 1. First-order repeater model.

2 Background and Related Work

2.1 Background: Wire Implementation

The delay of a wire can be modeled as a first-order RC circuit [12] (see Fig. 1). In this model, a CMOS driver is seen as a simple resistor R_{gate}, with a parasitic load C_{diff}, as shown in (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. Other parameters of importance are the transistor width w, the wire length l, and the normalized sum of NMOS+PMOS gate widths a.

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

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 allows for wire designs 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 times 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

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

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

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

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

TABLE 1
Area, Delay, and Power Characteristics of Wire Implementations (Extracted from [10])

Wire Type	Relative Latency	Relative Area	Dynamic Power (W/m) α=Switching Factor	Static Power W/m
B-Wire (8X plane)	$1x$	$1x$	2.65α	1.0246
B-Wire (4X plane)	$1.6x$	$0.5x$	2.9α	1.1578
L-Wire (8X plane)	$0.5x$	$4x$	1.46α	0.5670
PW-Wire (4X plane)	$3.2x$	$0.5x$	0.87α	0.3074

The average leakage power of a repeater is given by

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

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 a minimum size inverter.

Equations (3) and (4) show that the dissipated power can be reduced by employing smaller repeaters and increasing their spacing. Banerjee and Mehrotra [9] 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 mentioned before, in [10], 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 bigger 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 1 shows the relative delay, area, and power characteristics of *L* and *PW-Wires* compared to baseline wires (*B-Wires*), as reported in [10]. A $65\,\mathrm{nm}$ process technology is considered assuming 10 metal layers: 4 layers in the 1X plane, and 2 layers in each 2X, 4X, and 8X planes [13]. 4X and 8X metal planes are used for global intercore wires. It can be seen that *L-Wires* yield a twofold 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 [13], it is assumed that 4X and 8X wires are routed over memory arrays.

2.2 Related Work

The on-chip interconnection network is a critical design element in a multicore architecture, and consequently, it has been the subject of several recent works. Among others, Kumar et al. [13] 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.

A reduced number of works have attempted to exploit the properties of a heterogeneous interconnection network at the microarchitectural level in order to reduce the interconnect energy share. Beckmann and Wood [14], [15] 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. Nelson et al. [16] propose the use of silicon-based on-chip optical interconnects for minimizing the performance gap that the separation of the processing functions creates in a clustered architecture in an effort to alleviate power density. In [17], Balasubramonian et al. make the first proposal of wire management at the microarchitectural level. They introduce the concept of a heterogeneous interconnect that is composed 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 noncritical register values are transmitted on low-power wires. Subsequently, they extend the proposal in [18] with new techniques aimed at accelerating cache accesses in large L2/L3 split caches (L2/L3 NUCA architectures [19]) by taking advantage of a lower bandwidth, lower latency network.

Cheng et al. [10] 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 a D-torus).

More recently, Walter et al. [20] explore the benefits of adding a low-latency, customized shared bus as an integral part of the NoC architecture. This bus is used for certain transactions such as broadcast of queries, fast delivery of control signals, and quick exchange of small data items. Different from our proposal, they do not consider a directory-based CMP architecture when exploring the benefits of their proposal over a traditional NoC architecture. In [21], the authors propose a priority-based NoC, which differentiates between short control signals and long data messages to achieve a significant reduction in cache access delay. They also propose to use more efficient multicast and broadcast schemes instead of multiple unicast messages in order to implement the invalidation procedure and provide support for synchronization and mutual exclusion.

Finally, Balfour and Dally [22] evaluate a variety of on-chip networks designed for 64-core tiled CMPs, and compare them in terms of performance, area, and energy efficiency. They conclude that a concentrated 4×4 mesh architecture (each router is shared by four cores to reduce the hop count), replicated subnetworks, and express channels are the best option. Different from our work, the authors focus on the interconnection network design and obviate the cache coherence protocol (they assume an abstract communication protocol).

3 Efficient Message Management in Tiled CMPs

As introduced before, there are two main components in our proposal. The first approach is *Reply Partitioning* that

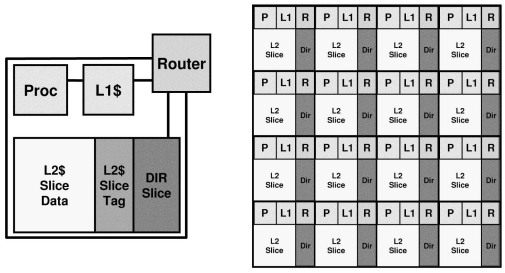

Fig. 2. Tiled CMP architecture overview.

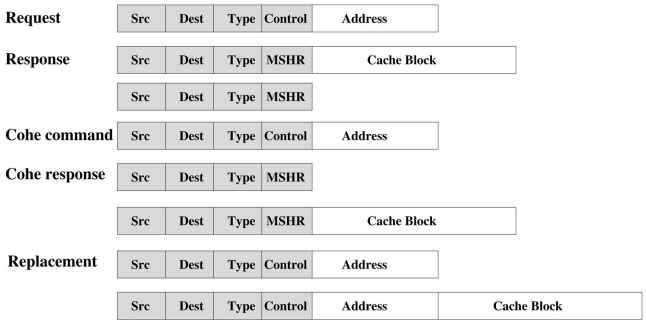

Fig. 3. Classification of messages that travel on the interconnection network of a tiled CMP architecture.

allows to classify all messages that the cores of a CMP exchange into two distinct groups. The second approach is a heterogeneous interconnect that uses separate wires, with different physical properties, for sending the messages through each group. 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 *Reply Partitioning* mechanism.

3.1 Tiled CMP Architectures

A tiled CMP architecture consists of a number of replicated *tiles* connected over a switched direct network (Fig. 2). 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 on-chip network. The L2 cache is shared among the different processing cores, but it is physically distributed between them.[1] 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 [19]). In addition, the L2 cache stores (in the tags' part of the local L2 slice) the directory information needed to ensure coherence between L1 caches. On an 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 $65\,\mathrm{nm}$, a tile area of approximately $25\,\mathrm{mm}^2$, and a die size of $400\,\mathrm{mm}^2$ [2], [23]. Note that this area is similar to the largest die in production today (Itanium 2 processor—around $432\,\mathrm{mm}^2$ [24]). Note also that, due to the manufacturing costs and form factor limitations, it would be desirable to keep the die size as low as possible [23]. Further details about the evaluation methodology and the simulated CMP configuration can be found in Section 4.1.

3.2 Classification of Messages in Tiled CMP Architectures

There is a variety of message types traveling on the interconnect of a CMP, each one with properties that are clearly distinct. In general, messages can be classified into the following groups (see Fig. 3):

1. Alternatively, each L2 slice could have been treated as a *private* L2 cache for the local processor. In this case, cache coherence has to be ensured at the L2 cache level (instead of L1). In any case, our proposal would be equally applicable in this configuration.

1. **Request messages.** These are generated by cache controllers in response to L1 cache misses. Requests are sent to the corresponding home L2 cache to demand privileges (read-only or read/write) over a memory line.
2. **Response messages.** These are sent in response to requests. These messages can be generated by the home L2 cache controller, or alternatively, by the remote L1 cache that has the single valid copy of the data, and they can carry the memory line or not. The latter is due to the presence of upgrade requests used to demand ownership for a line already kept in the local L1 cache.
3. **Coherence commands.** These are sent by the home L2 cache controller to the corresponding L1 caches to ensure coherence (for example, invalidation messages).
4. **Coherence responses.** These are sent by the L1 caches back to the corresponding home L2 in response to coherence commands.
5. **Replacement messages.** These are generated by L1 caches in case of exclusive or modified lines being replaced.

Fig. 4 plots the fraction of each message type over the total number of messages for a 16-core CMP configuration for the applications used in our evaluation (see Section 4.1 for configuration details). As it can be seen, on average, more than 60 percent of the messages are related to memory

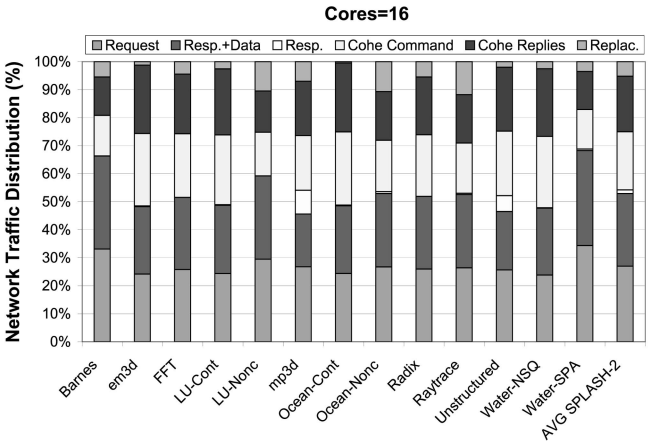

Fig. 4. Breakdown of the messages that travel on the interconnection network for a 16-core CMP.

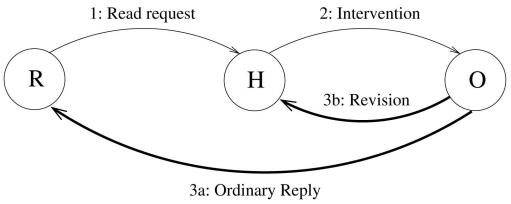

Fig. 5. Protocol actions taken in response to a read request that misses for a block in modified state (messages with data are represented using thicker lines).

accesses (a request and its corresponding reply), whereas the rest has to do with coherence enforcement (25 percent) and block replacement (15 percent). It is interesting to note that almost all replies imply the transfer of memory blocks ($Resp.+Data$).

Alternatively, messages involved in the L1 cache coherence protocol, shown in Fig. 4, can be classified according to their criticality into critical and noncritical messages. We say that a message is critical when it is in the critical path of the L1 cache miss. In other cases, we call the message noncritical. As expected, delaying a critical message will result in longer L1 cache miss latencies. On the other hand, slight slowdowns in the delivery of noncritical messages will not cause any performance degradation.

As an example, Fig. 5 shows the coherence actions involved in an L1 read miss for a line in modified state in other tiles. 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 sends back the line (3a) to the requestor and to the directory tile (3b). Whereas messages 1, 2, and 3a are critical because they belong to the critical path between the processor request and the memory system response, message 3b is noncritical. 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). At the same time, energy is saved, without affecting performance, when noncritical messages travel on slower, power-efficient *PW-Wires*.

On the other hand, coherence messages can also be classified according to their size into either short or 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.). In this way, we can say that they are short messages. Other message types, in particular, requests and coherence commands, also contain address block information but they are still narrow enough to be classified as short messages. Finally, replacements with data and data block transfers also carry a cache line, and therefore, they are classified as long messages.

Regarding the power dissipated by each message type, Fig. 6 plots their power breakdown for the baseline configuration using only *B-Wires*. As it can be seen, most of the power in the interconnect is associated to reply messages that carry L2 cache lines (55-65 percent). As previously commented, most of this power is dissipated in the point-to-point links, and therefore, message size plays a major role.

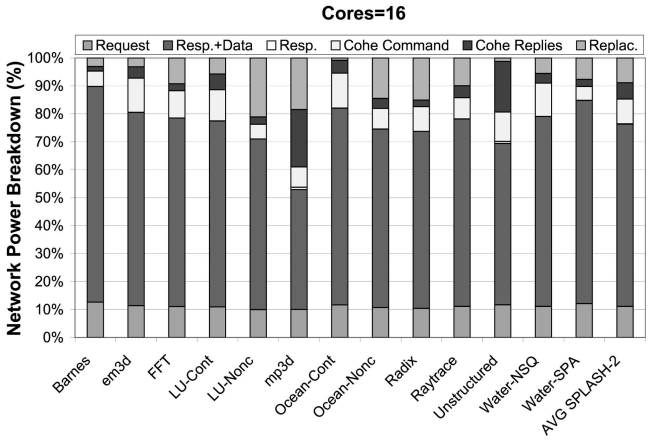

Fig. 6. Percentage of the power dissipated in the interconnection network by each message type for a 16-core CMP.

The use of a heterogeneous interconnect composed of low-latency *L-Wires* and power-efficient *PW-Wires* allows for a more energy-efficient interconnect utilization. However, as the number of *L-Wires* is smaller because of their fourfold area cost (relative to baseline wires), only short messages can take full advantage of them. On the other hand, since message size has direct impact on the power dissipated in the interconnect, significant energy savings can be obtained when long messages are sent through *PW-Wires*.

Table 2 summarizes the characteristics of each message type (in terms of criticality and length) and points out the links that would be preferred in every case. In general, short messages are critical, and therefore, should be sent through *L-Wires*. The exceptions are coherence replies that are not in the critical path of L1 cache misses (i.e., revision messages). On the other hand, long messages can be critical (responses with data) or noncritical (replacements with data), and the choice of wires to be used is not so clear in this case. If performance is what matters the most (criticality), then *L-Wires* might be the best choice. On the contrary, if energy is more important (length), then *PW-Wires* should be utilized. In this way, the policy of sending critical messages on *L-Wires* and noncritical on *PW-Wires* leaves the latter links underutilized since only replacements would make use of them, and small energy savings would be obtained. On the other hand, the policy of sending long messages on *PW-Wires* and short ones on *L-Wires* causes important delays to responses with data, which would finally translate into intolerable performance degradation. Different from the policies proposed in [10], which are mainly based on

TABLE 2
Classification of the Messages That Travel on the Interconnection Network According to Their Criticality and Length

Message Type	Critical?	Length	Preferred Wires (assuming unbounded number)
Request	Yes	Short	L-Wires
Response	Yes	Short/Long	L-Wires (Performance) PW-Wires (Energy)
Cohe. Command	Yes	Short	L-Wires
Cohe. Replies	Yes/No	Short	L-Wires (Critical) PW-Wires (Non-critical)
Replacements	No	Short/Long	PW-Wires

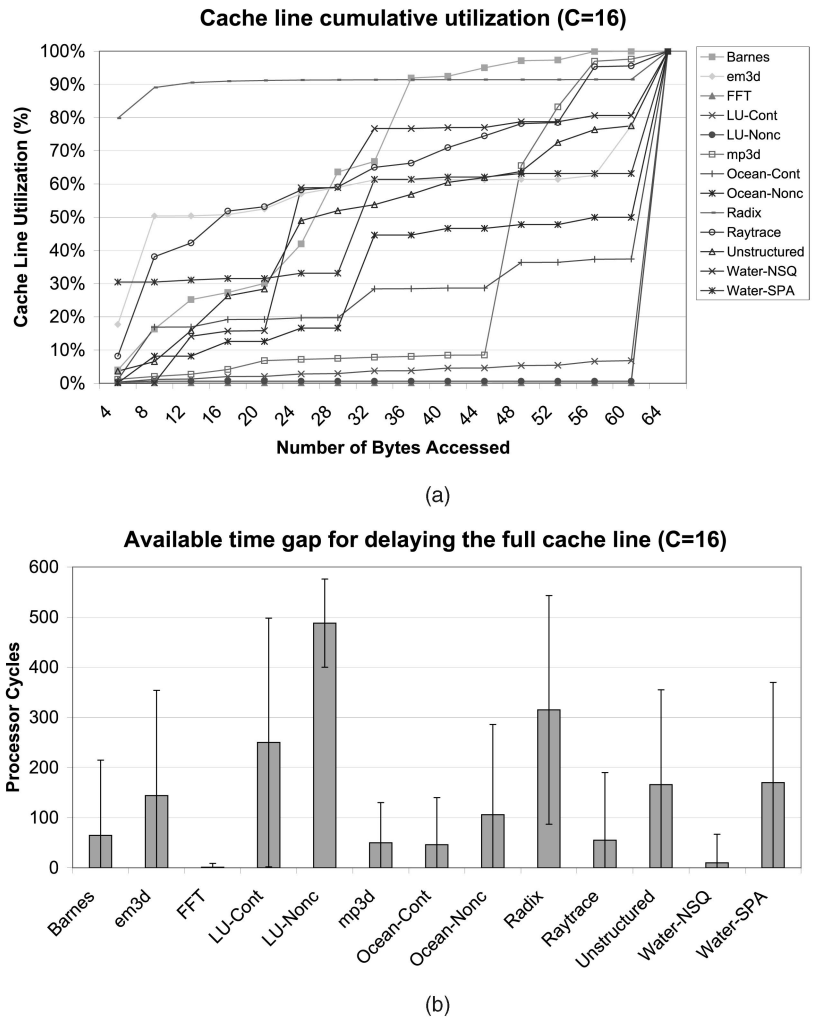

Fig. 7. (a) Cumulative utilization of cache lines and (b) available time gap for decoupling data messages for a 16-core CMP.

message criticality, in this work, we present a mechanism for efficient message management based on simultaneously taking advantage of both criticality and length properties by means of *Reply Partitioning* as we will describe next.

3.3 Reply Partitioning for Decoupling Data Messages into Critical and Noncritical Parts

In this work, we propose *Reply Partitioning*, a technique aimed at dealing with reply messages that carry data. *Reply Partitioning* is based on the observation that when the processor requests data that are not found in the L1 cache (L1 cache miss), the full line could not be necessary in that moment but only a small subset of it. In this way, our proposal splits replies with data into two messages. The first is a short *Partial Reply* (PR) message that carries a subblock of the cache line that includes the word requested by the processor. And the second message, called *Ordinary Reply*, is the original message and includes the full cache line. Our proposal is inspired by the critical-word-first scheme for uniprocessors described in [25] that requests the missed word first from memory and sends it to the processor as soon as it arrives, letting the processor continue execution while filling the remainder of the cache line.

Fig. 7a shows the cumulative utilization of cache lines for a CMP with 16 cores. We can observe a high variability among applications, ranging from applications where the common case is the access to a single word of the cache line before an invalidation or replacement takes place (80 percent of the cases for the Radix application) to applications as FFT and LU where the utilization of the full line is the common case (over 90 percent). More interesting is Fig. 7b, which measures the time between the first access to a cache line and a subsequent access to a different word in the same cache line, as a good indicator of the available time gap that we can take advantage of for decoupling replies with data. Again, we observe a high variability among the applications. For applications such as LU, Radix, or Water-SPA, the number of cycles between the first access to a cache line and a subsequent one to a different word is quite high, so a good behavior is expected for these applications with our *Reply Partitioning* proposal. On the other hand, benchmarks such as FFT or Water-NSQ that show a very low available time gap between subsequent cache accesses are expected to obtain poor performance gains with our proposal.

It is important to note that the division of replies with data into *Partial Replies* and *Ordinary Replies* makes *all* critical messages to be short (note that *Partial Replies* are critical since they contain the word requested by the processor), and therefore, they can be sent using the low-latency *L-Wires*. At

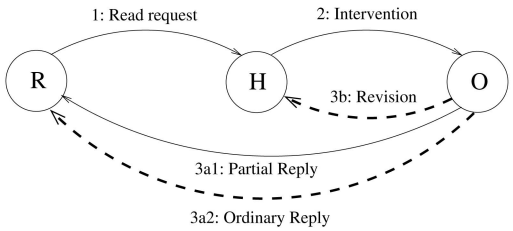

Fig. 8. New behavior for the case discussed in Fig. 5. The use of *L-Wires* is represented by solid lines whereas dashed lines mean that power-efficient *PW-Wires* are employed.

the same time, *all* long messages become now noncritical (note that *Ordinary Replies* are noncritical since the requested word also travels on a short message that hopefully will arrive sooner), and therefore, they can be sent using the power-efficient *PW-Wires* without hurting performance. For a reply with data, a performance improvement will be obtained provided that the arrival time of the *Ordinary Reply* happens within the time gap shown in Fig. 7b.

Fig. 8 shows the new behavior for the example discussed previously (Fig. 5) when *Reply Partitioning* is applied. We can observe that now *all* messages belonging to the critical path between the processor request and the memory system response are sent employing *L-Wires*, represented by solid lines. At the same time, noncritical messages such as the revision to the directory tile and the ordinary reply are sent using power-efficient *PW-Wires*, represented by dashed lines in the figure.

Additionally, splitting reply messages into a critical *Partial Reply* and a noncritical *Ordinary Reply* has some implications over the coherence protocol. Recall that, in a nonblocking cache, MSHR registers are used to keep track of outstanding misses [26]. When the corresponding reply arrives, requests held in MSHR are processed and the corresponding processor instructions are committed if needed. In our mechanism, we have two different replies and we need to define the actions required after the arrival of each one. Furthermore, with a 2D-mesh direct interconnection network, the arrival order could not be guaranteed. This means that, although unlikely, the noncritical *Ordinary Reply* could be received before the critical *Partial Reply*.[2]

When a *Partial Reply* arrives we are sure that all coherence actions that ensure L1 cache coherence have been done. Therefore, after its arrival, all waiting requests that can be satisfied are processed (e.g., read requests that hit in the cache line subblock and all write requests). For a read request, the corresponding value is sent to the processor, and for a write request, the value is held in the MSHR but the rest of hardware resources are released. In both cases, appropriate processor instructions are committed. Only MSHR with nonprocessed read requests and all the write requests, if any, are kept until the arrival of the *Ordinary Reply* containing the full line. At the *Ordinary Reply* arrival time, the rest of read requests are performed and the block is modified with the corresponding write values held in the MSHR. In case of receiving the *Ordinary Reply* before the *Partial Reply*, all requests waiting in the MSHR register are processed and the corresponding instructions are committed; all hardware resources are released but the MSHR, which is released when both replies arrive.

2. This could happen when adaptive routing is used.

3.4 Designing the Interconnect for Efficient Message Management

As discussed in Section 2.1, *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, the number of wires should be fixed by considering the typical size of short messages. The remaining area will be consumed by *PW-Wires* employed for sending long, noncritical messages.

In this work, we use the same main parameters for the interconnect as in [10]. 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, and partial replies are 11 bytes long, since besides control information (3 bytes), they also carry address information (in the first two cases) or the subblock of data of one word size (for partial replies). On the other hand, coherence replies are just 3 bytes long. Replacements for lines in modified state are 75 bytes long since they carry both address (8 bytes) and a cache line (64 bytes) besides control information (3 bytes). Finally, ordinary reply messages are 67 byte long since they carry control information (3 bytes) and a cache line (64 bytes). In order to match the metal area of the baseline configuration, each original 75-byte unidirectional link is designed to be made up of 88 *L-Wires* (11 bytes) and 248 *PW-Wires* (31 bytes). For a discussion regarding the implementation complexity of heterogeneous interconnects, refer to [10].

The resulting design is similar to that proposed in [10], but with some important differences. First of all, the election of the right set of wires for a message does not require any additional logic since it can be made based exclusively on one bit in the length field (some of the proposals developed in [10] require checking of the directory state or tracking the level of congestion in the network). Second, the routing logic and the multiplexers and demultiplexers associated with wires are simpler since we only consider two types of wires instead of three. Finally, our proposal achieves better levels of utilization of each set of wires (as we will discuss in Section 4.2.1).

4 Experimental Results

This section shows the results that are obtained for our proposal and compares them against those achieved with two different configurations of the interconnect. The first is the configuration that employs just *B-Wires*, which is taken as the baseline. The second configuration is an implementation of the heterogeneous interconnect presented in [10] that uses *L*, *B*, and *PW-Wires*.

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 that estimates both dynamic and leakage power and is based on RSIM [27]. RSIM is a detailed execution-driven simulator that models out-of-order superscalar processors (although in-order issue is also supported), several levels of caches, an aggressive memory, and the interconnection network, including contention at all resources. In particular,

TABLE 3
Configuration of the Baseline CMP Architecture and Applications Evaluated

CMP Configuration	
Parameter	
Process technology	65 nm
Tile area	25 mm^2
Number of tiles	16
Cache line size	64 bytes
Core	4GHz, in-order 2-way model
L1 I/D-Cache	32KB, 4-way
L2 Cache (per core)	256KB, 4-way, 6+2 cycles
Memory access time	400 cycles
Network configuration	2D mesh at 75 GB/s
Router parameters	3 stages (full pipeline)
	2 VCs
	32 flit buffers per input port
Link width	75 bytes (*8X-B-Wires*)
Link latency	4 cycles (full pipeline)
Link length	5 mm

(a)

Application	Problem size
Barnes-Hut	16K bodies, 4 timesteps
EM3D	9600 nodes, 5% remote links, 4 timesteps
FFT	256K complex doubles
LU-cont	256×256, B=8
LU-noncont	256×256, B=8
MP3D	50000 nodes, 2 timesteps
Ocean-cont	258×258 grid
Ocean-noncont	258×258 grid
Radix	2M keys
Raytrace	car.env
Unstructured	mesh.2K, 5 timesteps
Water-nsq	512 molecules, 4 timesteps
Water-spa	512 molecules, 4 timesteps

(b)

our simulation tool employs as performance simulator a modified version of RSIM that models the architecture of the tiled CMP presented in Section 3. We have incorporated into our simulator already proposed and validated power models for both dynamic power (from Wattch [28], CACTI [29]) and leakage power (from HotLeakage [30]) of each processing core, as well as the interconnection network (from Orion [31]). Our simulation tool has been previously validated against HotLeakage for the processing cores and Orion for the power modeling of the interconnection network [6].

Table 3a shows the architecture configuration used across this paper. It describes an 8 and 16-core CMP built in 65 nm technology. The tile area has been fixed to 25 mm^2, including a portion of the second-level cache [2]. With this configuration, links that interconnect routers configuring the 2D mesh topology would measure around 5 mm. For our configuration, about 70 percent of the power dissipated by the interconnect is due to the link circuitry. This value is a little higher than the 60 percent dissipated by links in the Alpha 21364 routers and a little lower than the 82 percent cited in [7]. The router versus link power strongly depends on the amount of buffer space assigned to the router compared with channel bandwidth.

TABLE 4
Relative Area, Delay, and Power Characteristics of *L* and *PW-Wire* Implementations in the 8X Plane Related to Baseline Wires (*B-Wires*)

Wire Type	Relative Latency	Relative Area	Dynamic Power (W/m) α=Switching Factor	Static Power W/m
L-Wire (8X plane)	$0.5x$	$4x$	1.46α	0.5670
PW-Wire (8X plane)	$2x$	x	0.80α	0.2720

With respect to messages traveling on the interconnect, reply messages are 67 bytes long since they carry 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 bytes long (just 3 bytes long for coherence replies). A replacement message for a line in modified state, which is 75 bytes long, is the largest message in this configuration (with 3 control bytes, 8 address bytes, and 64 bytes of data).

Table 3b shows the applications used in our experiments. *MP3D* is from the SPLASH benchmark suite [32], *Barnes-Hut*, *FFT*, *LU-cont*, *LU-noncont*, *Ocean-cont*, *Ocean-noncont*, *Radix*, *Raytrace*, and *Water-nsq* are from the SPLASH-2 benchmark suite [33], Berkeley *EM3D* simulates the propagation of electromagnetic 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 simulations, following the recommendations given in [33]. All experimental results reported in this work are for the parallel phase of these applications. Data placement in our programs is either done explicitly by the programmer or by our simulator that uses a first-touch policy on a cache-line granularity. Thus, initial data placement is quite effective in terms of reducing traffic in the interconnection network.

Table 4 shows the relative area, delay, and power characteristics of *L* and *PW-Wires* compared to baseline wires (*B-Wires*) when a single metal plane is considered (the 8X plane). In order to match the metal area of the baseline configuration, each original 75-byte unidirectional link (600 B-Wires) is designed in our heterogeneous configuration to be made up of 88 *L-Wires* (11 bytes) and 248 *PW-Wires* (31 bytes). For comparison purpose, the 3-subnetwork heterogeneous interconnect described in [10] was also implemented. In that configuration, each link is composed of three types of wires in two metal planes. Each wire type has the area, delay, and power characteristics described in Table 1, so each original link is designed to be made up of 24 *L-Wires* (3 bytes), 512 *PW-Wires* (64 bytes), and 256 *B-Wires* (32 bytes).

4.2 Simulation Results and Analysis

In this section, we show how messages distribute between the different types of wires of the heterogeneous networks evaluated in this work. Then, we analyze the impact of our proposal on execution time and the energy dissipated by the intercore links. Finally, we report the energy and energy-delay^2 product metrics for the full CMP. As in [10], all results have been normalized with respect to the baseline

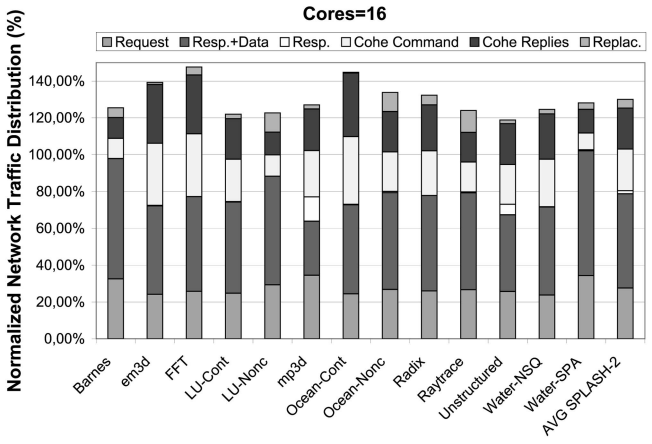

Fig. 9. Breakdown of the messages that travel on the interconnection network for a 16-core CMP when an *L-Wire*/*PW-Wire* heterogeneous network is used and long critical messages are split.

case, where only *B-Wire*, unidirectional 75-byte wide links are considered.

4.2.1 Message Distribution in Heterogeneous Networks

Fig. 9 plots the fraction of each message type over the total number of messages sent in the baseline configuration for the 16-core configuration. It is important to note that *Reply Partitioning* increases the total number of messages that travels on the interconnect. The reason is that replies with data are converted into two messages, the *Partial Reply* and the *Ordinary Reply*. In our particular case, it can be observed that the number of messages increases around 30 percent, on average.

Fig. 10 plots the workload distribution between the different types of wires of the heterogeneous networks evaluated in this paper. This figure is obtained by measuring the traffic observed for each type of wire, normalized to the width of the wire. As it can be seen in the left graph, when using three subnetworks, there is an underutilization of *L* and *PW*-wires that leads to an imbalanced workload distribution. However, the use of a 2-subnetwork interconnect, where *B-Wires* have been replaced with wider *L-Wires*, in conjunction with the *Reply Partitioning* technique leads to a more balanced workload distribution (Fig. 10b).

4.2.2 Execution Time Results

Fig. 11 depicts the normalized execution time with respect to that obtained for the baseline configuration for an 8 and a 16-core CMP. The first barline ($C = 8$ *or* 16, $Subnets = 3$) shows the normalized execution time for the 3-subnetwork interconnect (as proposed in [10]). It can be observed an average performance degradation of 5-13 percent, which is a trend, also reported in [10] when a 2D torus topology is employed. The reason of this degradation is the low use of the *L-Wires* as it was shown in Fig. 10. Similar results are obtained when a 2-subnetwork interconnect (*L-Wire*/*PW-Wire*) is considered without using the proposed *Reply Partitioning* mechanism, as shown in the second barline ($C = 8$ *or* 16, $Subnets = 2$). The reason of this performance degradation is the increased latency of the reply messages that carry data (sent through the slower *PW-Wires*), which

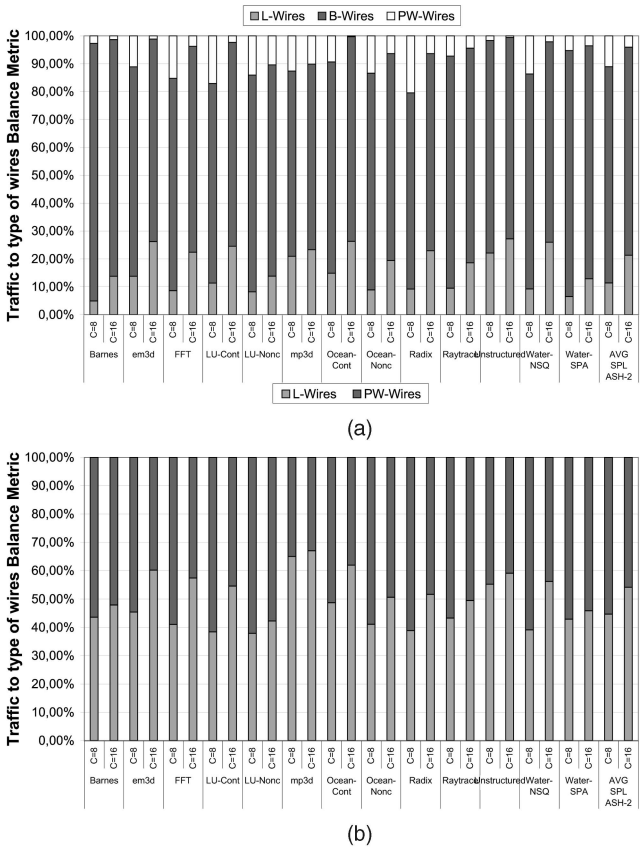

Fig. 10. Workload distribution for (a) 3-subnetwork approach (as in [10]) and (b) 2-subnetwork approach. The use of a 2-subnetwork interconnect in conjunction with the *Reply Partitioning* technique leads to a more balanced workload distribution.

cannot be hidden by using faster *L-Wires* for critical messages. This degradation has high variability, ranging from almost negligible for mp3d and Water-NSQ applications to almost 20 percent for Ocean-Cont application. This result is quite interesting because it shows that the increased latency imposed by the use of *PW-Wires* for replies with data can be hidden in some applications, while in others, as Barnes or Ocean-Cont, it translates into significant performance degradation. Finally, the third barline ($C = 8$ *or* 16, $Subnets = 2$ *RP*) shows the case when the proposed *Reply Partitioning* (RP) is applied, splitting replies into critical, short *partial replies*, and noncritical

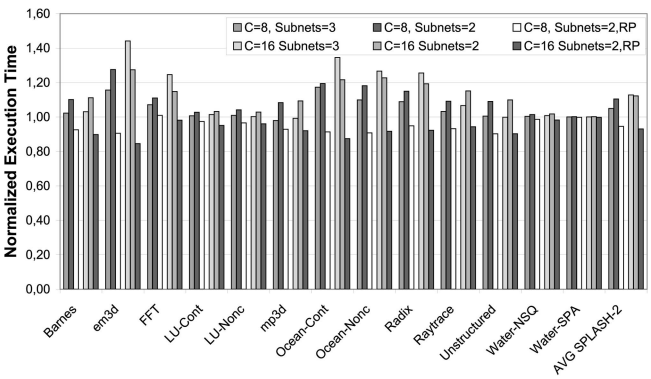

Fig. 11. Normalized execution time when heterogeneous links are used.

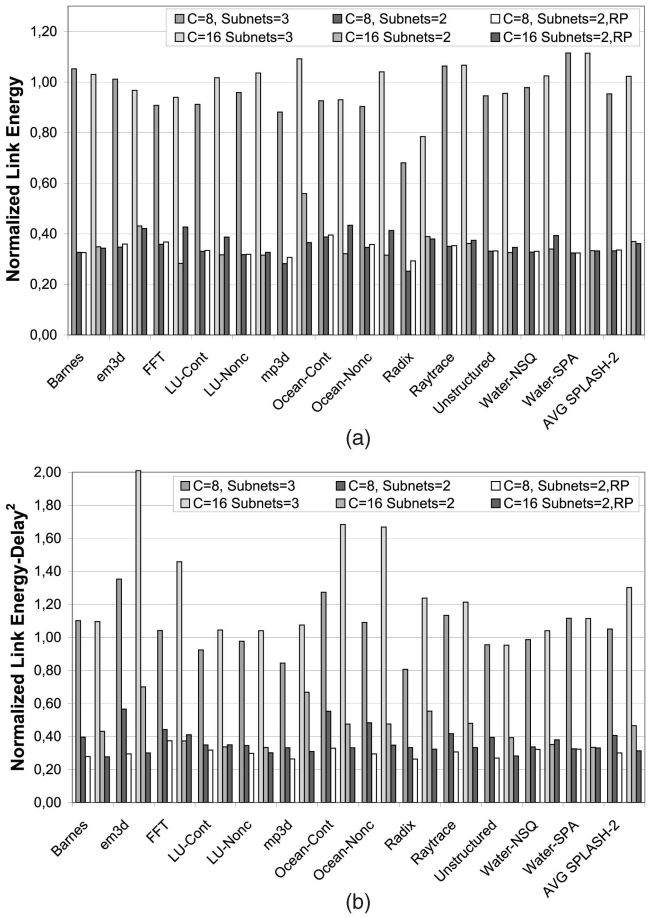

Fig. 12. (a) Normalized link energy and (b) $\mathrm{energy\text{-}delay}^2$ product for the evaluated configurations.

ordinary replies. On average, we observe performance improvements of 16 percent over the two previous options for a 16-core CMP. These important improvements are a direct consequence of the better distribution of the messages between *L-Wires* and *PW-Wires* that *Reply Partitioning* allows. Again, a high variability is found, with improvements ranging from 1-2 percent in some applications to 50-55 percent in others. Compared with the baseline configuration where no heterogeneous network is used, an average performance improvement of 7 percent is obtained.

4.2.3 Energy Efficiency Results

Fig. 12 plots both normalized link energy and $\mathrm{energy\text{-}delay}^2$ product (ED^2P) metrics. The proposal developed in this work results in an average reduction of 60-65 percent in the energy dissipated by the intercore links. This reduction is quite similar for all applications. The ED^2P metric also shows good results, with average improvements close to 70 percent although, in this case, the variability between applications is even higher because in the ED^2P metric the execution time gains importance.

Finally, Fig. 13 presents both the normalized energy and ED^2P product metrics for the full CMP. As it can be observed, important energy savings are obtained for the whole CMP when applying our proposal. The magnitude of these savings depends on the total number of cores of the CMP, ranging from 16 percent for the 8-core configuration

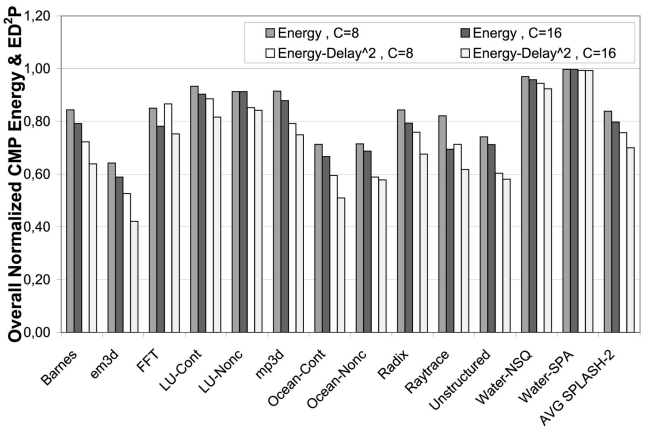

Fig. 13. Normalized energy and $\mathrm{energy\text{-}delay}^2$ product (ED^2P) for the full CMP.

to 20 percent for the 16-core configuration. On the other hand, when the ED^2P metric is considered, we find an increased improvement that ranges from 24 percent for the 8-core CMP to 30 percent for the 16-core one, due to the bigger emphasis on the execution time.

5 Sensitivity Analysis

In this section, we discuss the impact of the issue policy of the processing cores, link bandwidth, relative latency between the second-level cache and the interconnect, and finally, the critical subblock size on our proposed *Reply Partitioning* approach with a heterogeneous interconnect.

5.1 Out-of-Order/In-Order Processors

In this paper, we have considered in-order processing cores. We have configured *SimPower-CMP* to model a CMP with out-of-order processing cores with the same parameters shown in Table 3. Fig. 14 (first barline) shows the performance speedup of our proposal over an out-of-order baseline configuration for a 16-core CMP. All benchmarks except Ocean show degrees of performance improvement that range from 10 percent for Barnes to almost negligible for mp3d and Water-NSQ. The average reduction in the execution time is less than what we observe in a CMP using in-order cores (an improvement of 3 percent versus a 7 percent obtained when

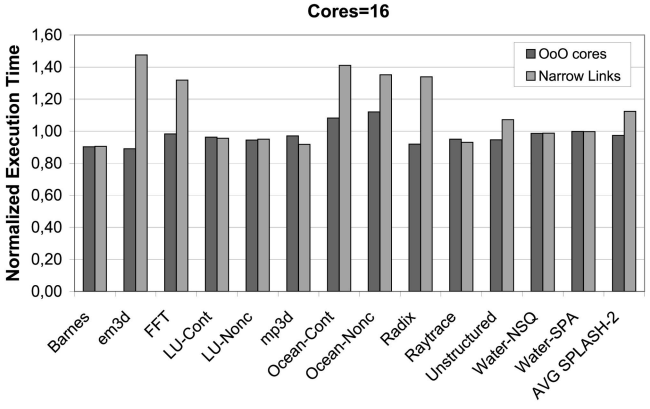

Fig. 14. Normalized execution time when heterogeneous links and OoO cores are used (first barline) or narrow links are considered (second barline) for a 16-core CMP.

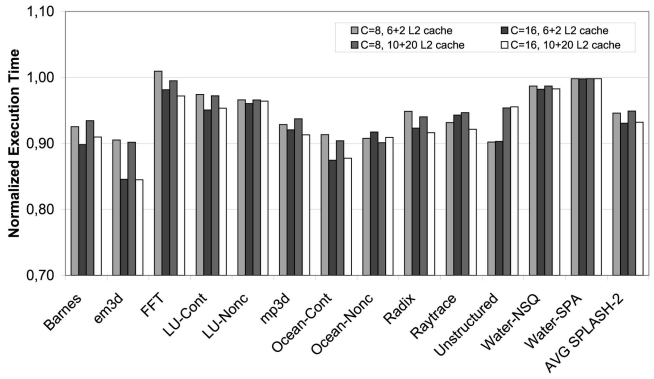

Fig. 15. Normalized execution time when heterogeneous links are used for different L2 cache access time.

in-order processors 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 2.1, *L-Wires* have a four-fold area cost compared to baseline wires, and therefore, the number of *L-Wires* is quite limited. In a bandwitdth-constrained system, this constraint is exacerbated, and therefore, our proposal is likely to not perform very well. To verify this, we consider the same base case that authors propose in [10], where each link only has 80 *B-Wires* in the 8X metal layer. As in [10], 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 *PW-Wires* (8 bytes). Fig. 14 (second barline) shows the performance speedup when considering narrow links with respect to the new baseline interconnect. Benchmarks such as Water-SPA, Barnes or LU, which exhibit lower network utilization or bigger available gap for decoupling data messages into partial and ordinaries replies, show performance improvements close to those obtained in the high-bandwidth simulations. The rest of the benchmarks suffer significant performance losses. Overall, the heterogeneous model performs 16 percent worse than the base case.

5.3 Relative Latency of the Interconnect with Respect to the L2 Cache

As we explained in Section 1, across this paper, we have considered a more up-to-date access time to the L2 cache slices ($6 + 2$ cycles) than in previous works [10], [11], where L2 caches present higher latencies ($10 + 20$ cycles). In this section, we evaluate the impact of the relative latency of the interconnect with respect to the L2 cache. Fig. 15 shows the normalized execution time when reply messages are split into critical, short *Partial Replies*, and noncritical *Ordinary Replies* for both L2 cache configurations. For benchmarks such as Barnes, mp3d, or Unstructured with either bigger available gap or little L2 cache line utilization, *Reply Partitioning* performs better when the weight of the interconnect over the overall L2 access time is higher ($6 + 2$ cycles configuration). On the other hand, for benchmarks that show higher L2 cache utilizations, such as FFT, sending the full cache line using *PW-Wires* has bigger impact in the execution time. This is due to the higher weight of the interconnect in this configuration. On average, results obtained under both configurations are quite similar because some applications perform better under the first configuration and others under the second one.

5.4 Partial Reply Subblock Size Effect

Finally, Fig. 16 shows the effect of augmenting the size of the cache-line subblock carried within *Partial Replies*. Fig. 16a shows the normalized execution time with respect to that obtained for the baseline configuration for an 8 and a 16-core CMP when *Reply Partitioning* technique is used along with heterogeneous links composed of *L-Wires* and *PW-Wires*. On average, the obtained results are quite similar for subblocks of size 4 and 8 bytes; meanwhile, the execution time is minimized when *Partial Replies* carry on subblocks of 16 bytes (on average, an additional 1 percent improvement on the normalized execution time over the 4 and 8-byte subblock size configurations). Fig. 16b plots the normalized energy consumed by the heterogeneous links. On average, the 4-byte subblock size configuration consumes from 5 to 7 percent less energy in the intercore links than the 16-byte one (around 2 percent less when the ED^2P metric is considered). Therefore, from the point of the energy consumed in the intercore links, the best result is obtained for a 4-byte subblock size for the *Partial Replies*.

6 Conclusions

In this work, we have proposed an efficient message management mechanism (from the point of view of both performance and energy) for tiled CMPs that consists of two approaches. The first one is *Reply Partitioning*, a technique that allows all messages used to ensure coherence between the L1 caches of a CMP to be classified into two groups: critical and short, and noncritical and long. In particular, *Reply Partitioning* concentrates on replies that carry data and splits them into a critical and short *Partial Reply* message that carries the word requested by the processor plus a noncritical *Ordinary Reply* with the full cache line. The second approach of our proposal is the use of a heterogeneous interconnection network composed of only two different types of wires: low-latency wires for critical messages and low-energy wires for noncritical messages, which also allows for a more balanced workload across the interconnect.

Results obtained through detailed simulations of 8 and 16-core CMPs show that the proposed on-chip message management mechanism can reduce the power dissipated by the links of the interconnection network about 65 percent with an additional reduction in execution time of 7 percent over previous works. Finally, these reductions translate into overall CMP energy savings ranging from 16 percent for the 8-core configuration to 20 percent for the 16-core one (from 24 to 30 percent if 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 percent versus 3 percent 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 on the performance of *Reply Partitioning*. Finally, when a

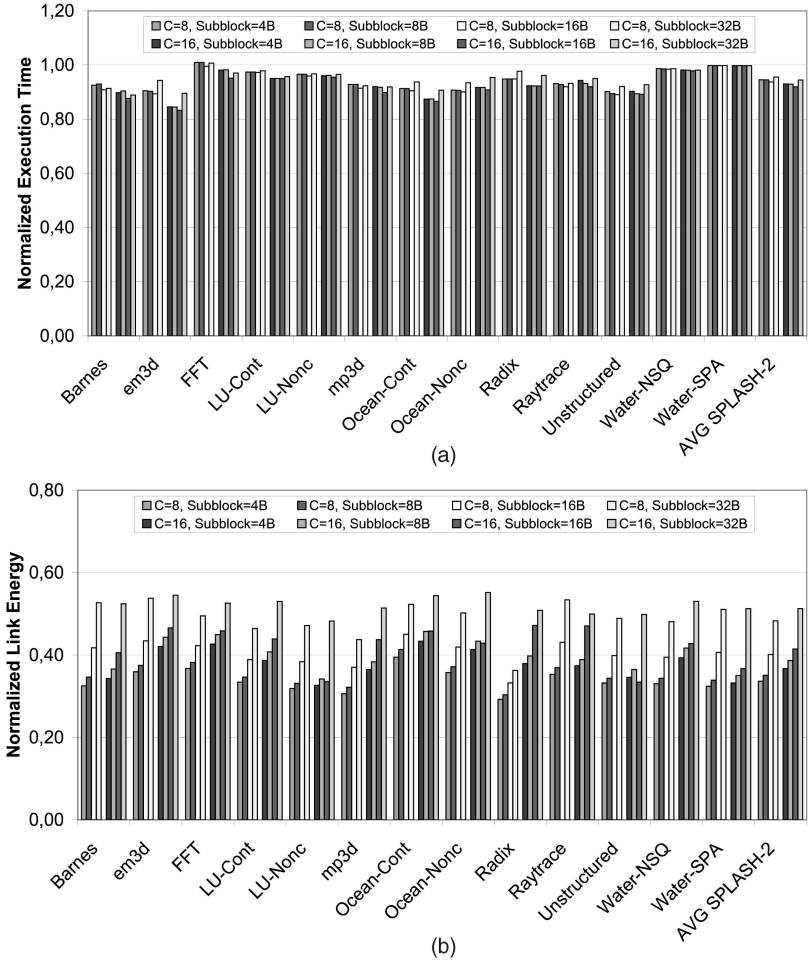

Fig. 16. (a) Normalized execution time and (b) normalized link energy for different size of the partial replies for 8 and 16-core CMPs.

bandwitdth-constrained system is considered, the fourfold area cost of the *L-Wires* becomes a problem, and on average, the heterogeneous model performs 16 percent 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] M.B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, J.-W. Lee, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, and A. Agarwal, "The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs," *IEEE Micro,* vol. 22, no. 2, pp. 25-35, Mar./Apr. 2002.

[2] M. Zhang and K. Asanovic, "Victim Replication: Maximizing Capacity While Hiding Wire Delay in Tiled Chip Multiprocessors," *Proc. 32nd Int'l Symp. Computer Architecture (ISCA-32),* pp. 336-345, June 2005.

[3] H. Wang, L.-S. Peh, and S. Malik, "Power-Driven Design of Router Microarchitectures in On-Chip Networks," *Proc. 36th Int'l Symp. Microarchitecture (MICRO-36),* pp. 105-111, Dec. 2003.

[4] N. Magen, A. Kolodny, U. Weiser, and N. Shamir, "Interconnect-Power Dissipation in a Microprocessor," *Proc. Sixth Int'l Workshop System Level Interconnect Prediction (SLIP-6),* pp. 7-13, Feb. 2004.

[5] A. Flores, J.L. Aragón, and M.E. Acacio, "Sim-PowerCMP: A Detailed Simulator for Energy Consumption Analysis in Future Embedded CMP Architectures," *Proc. Fourth Int'l Symp. Embedded Computing (SEC-4),* pp. 752-757, May 2007.

[6] A. Flores, J.L. Aragón, and M.E. Acacio, "An Energy Consumption Characterization of On-Chip Interconnection Networks for Tiled CMP Architectures," *The J. Supercomputing,* vol. 45, no. 3, pp. 341-364, 2008.

[7] L. Shang, L. Peh, and N. Jha, "Dynamic Voltage Scaling with Links for Power Optimization of Interconnection Networks," *Proc. Ninth Int'l Symp. High-Performance Computer Architecture (HPCA-9),* pp. 91-102, Feb. 2003.

[8] H.-S. Wang, L.-S. Peh, and S. Malik, "A Power Model for Routers: Modeling Alpha 21364 and InfiniBand Routers," *IEEE Micro,* vol. 23, no. 1, pp. 26-35, Jan./Feb. 2003.

[9] K. Banerjee and A. Mehrotra, "A Power-Optimal Repeater Insertion Methodology for Global Interconnects in Nanometer Designs," *IEEE Trans. Electron Devices,* vol. 49, no. 11, pp. 2001-2007, Nov. 2002.

[10] L. Cheng, N. Muralimanohar, K. Ramani, R. Balasubramonian, and J. Carter, "Interconnect-Aware Coherence Protocols for Chip Multiprocessors," *Proc. 33rd Int'l Symp. Computer Architecture (ISCA-33),* pp. 339-351, June 2006.

[11] A. Flores, J.L. Aragón, and M.E. Acacio, "Efficient Message Management in Tiled cmp Architectures Using a Heterogeneous Interconnection Network," *Proc. 14th Int'l Conf. High Performance Computing (HiPC '07),* pp. 133-146, 2007.

[12] R. Ho, K. Mai, and M. Horowitz, "The Future of Wires," *Proc. IEEE,* vol. 89, no. 4, pp. 490-504, Apr. 2001.

[13] R. Kumar, V. Zyuban, and D.M. Tullsen, "Interconnections in Multi-Core Architectures: Understanding Mechanisms, Overheads and Scaling," *Proc. 32nd Int'l Symp. Computer Architecture (ISCA-32),* pp. 408-419, June 2005,

[14] B.M. Beckmann and D.A. Wood, "TLC: Transmission Line Caches," *Proc. 36th Int'l Symp. Microarchitecture (MICRO-36),* pp. 43-54, Dec. 2003.

[15] B.M. Beckmann and D.A. Wood, "Managing Wire Delay in Large Chip-Multiprocessor Caches," *Proc. 37th Int'l Symp. Microarchitecture (MICRO-37),* pp. 319-330, Dec. 2004.

[16] N. Nelson, G. Briggs, M. Haurylau, G. Chen, H. Chen, D. Albonesi, E. Friedman, and P. Fauchet, "Alleviating Thermal Constraints While Maintaining Performance via Silicon-Based On-Chip Optical Interconnects," *Proc. Workshop Unique Chips and Systems (UCAS-1),* pp. 339-351, Mar. 2005.

[17] R. Balasubramonian, N. Muralimanohar, K. Ramani, and V. Venkatachalapathy, "Microarchitectural Wire Management for Performance Power in Partitioned Architectures," *Proc. 11th Int'l Symp. High-Performance Computer Architecture (HPCA-11),* pp. 28-39, Feb. 2005.

[18] N. Muralimanohar and R. Balasubramonian, "The Effect of Interconnect Design on the Performance of Large L2 Caches," *Proc. Third IBM Watson Conf. Interaction between Architecture, Circuits, and Compilers (P=ac2),* Oct. 2006.

[19] C. Kim, D. Burger, and S.W. Keckler, "An Adaptive Non-Uniform Cache Structure for Wire-Delay Dominated On-Chip Caches," *Proc. 10th Int'l Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS-10),* pp. 211-222, Nov. 2002.

[20] I. Walter, I. Cidon, and A. Kolodny, "BENoC: A Bus-Enhanced Network On-Chip for a Power Efficient CMP," *IEEE Computer Architecture Letters,* vol. 7, no. 2, pp. 61-64, July-Dec. 2008.

[21] E. Bolotin, Z. Guz, I. Cidon, R. Ginosar, and A. Kolodny, "The Power of Priority: Noc Based Distributed Cache Coherency," *Proc. First Int'l Symp. Networks-on-Chip (NOCS '07),* pp. 117-126, 2007.

[22] J. Balfour and W.J. Dally, "Design Tradeoffs for Tiled CMP On-Chip Networks," *Proc. 20th Int'l Conf. Supercomputing (ICS-20),* pp. 187-198, June 2006.

[23] L. Zhao, R. Iyer, S. Makineni, J. Moses, R. Illikkal, and D. Newell, "Performance, Area and Bandwidth Implications on Large-Scale CMP Cache Design," *Proc. First Workshop Chip Multiprocessor Memory Systems and Interconnects (CMP-MSI'07),* in conjunction with HPCA-13, Feb. 2007.

[24] C. Liu, A. Sivasubramaniam, and M. Kandemir, "Organizing the Last Line of Defense before Hitting the Memory Wall for CMPs," *Proc. 10th Int'l Symp. High Performance Computer Architecture (HPCA-10),* pp. 176-185, Feb. 2004.

[25] J.L. Hennessy and D.A. Patterson, *Computer Architecture: A Quantitative Approach,* fourth ed. Morgan Kaufmann, 2006.

[26] D. Kroft, "Lockup-Free Instruction Fetch/Prefetch Cache Organization," *Proc. Eighth Int'l Symp. Computer Architecture (ISCA-8),* pp. 81-87, May 1981.

[27] C.J. Hughes, V.S. Pai, P. Ranganathan, and S.V. Adve, "RSIM: Simulating Shared-Memory Multiprocessors with ILP Processors," *Computer,* vol. 35, no. 2, pp. 40-49, Feb. 2002.

[28] D. Brooks, V. Tiwari, and M. Martonosi, "Wattch: A Framework for Architectural-Level Power Analysis and Optimizations," *Proc. 27th Int'l Symp. Computer Architecture (ISCA-27),* pp. 83-94, June 2000.

[29] P. Shivakumar and N.P. Jouppi, "Cacti 3.0: An Integrated Cache Timing, Power and Area Model," technical report, Western Research Lab (WRL), 2001.

[30] Y. Zhang, D. Parikh, K. Sankaranarayanan, K. Skadron, and M. Stan, "HotLeakage: A Temperature-Aware Model of Subthreshold and Gate Leakage for Architects," technical report, Univ. of Virginia, 2003.

[31] H.-S. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: A Power-Performance Simulator for Interconnection Networks," *Proc. 35th Int'l Symp. Microarchitecture (MICRO-35),* pp. 294-305, Nov. 2002.

[32] J. Singh, W.-D. Weber, and A. Gupta, "SPLASH: Stanford Parallel Applications for Shared-Memory," *Computer Architecture News,* vol. 20, no. 1, pp. 5-44, 1992.

[33] S.C. Woo, M. Ohara, E. Torrie, J.P. Singh, and A. Gupta, "The SPLASH-2 Programs: Characterization Methodological Considerations," *Proc. 22nd Int'l Symp. Computer Architecture (ISCA-22),* pp. 24-36, June 1995.

Antonio Flores received the MS degree in computer science from the Universidad de Murcia, Spain, in 1994. He is currently working toward the PhD degree and is an 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.

Juan L. Aragón received the MS degree in computer science and the PhD degree in computer engineering from the Universidad de Murcia, Spain, in 1996 and 2003, respectively, followed by a one-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 is currently 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.

Manuel E. Acacio received the MS and PhD 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.