Efficient Message Management in Tiled CMP Architectures Using a Heterogeneous Interconnection Network

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

Departamento de Ingeniería y Tecnología de Computadores
University of Murcia, 30100 Murcia (Spain)
{aflores, jlaragon, meacacio}@ditec.um.es

Abstract. 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 present a proposal for performance-and energy-efficient message management in tiled CMPs by using a heterogeneous interconnect. Our proposal consists of *Reply Partitioning*, a technique that classifies all coherence messages into critical and short, and non-critical and long messages; and the use of a heterogeneous interconnection network comprised of low-latency wires for critical messages and low-energy wires for non-critical ones. Through detailed simulations of 8- and 16-core CMPs, we show that our proposal obtains average improvements of 8% in execution time and 65% in the Energy-Delay^2 Product metric of the interconnect over previous works.

Keywords: Chip-Multiprocessor, Energy-Efficient Architectures, Heterogeneus On-Chip Interconnection Network, Parallel Scientific Applications.

1 Introduction

High performance processor designs are evolving toward architectures that implement multiple processing cores on a single die. Chip-multiprocessors (CMPs) can provide higher throughput, more scalability and greater energy-efficiency compared to wider-issue, single-core processors. Furthermore, energy-efficient architectures are currently one of the major goals pursued by designers in both high performance and embedded processor domains.

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 a switched direct network [1,2]. In these architectures, the design of the on-chip interconnection network has been shown to have significant impact on overall system performance and energy consumption, since it is implemented using global wires that show long delays and high capacitance properties. Recently, Wang *et al.* [3] reported that the on-chip network of the Raw processor consumes 36% of total chip power. Magen *et al.* [4] 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 [5,3].

By tuning wire's characteristics such as wire width, spacing or repeater size, it is possible to design wires with varying latency, bandwidth and energy properties [6]. Using links that are comprised of wires with different physical properties, a heterogeneous on-chip interconnection network is obtained. In [7], the authors propose the use of links that are comprised of two wire implementations apart from baseline wires (*B-Wires*): power optimized wires (*PW-Wires*) with fewer and smaller repeaters, and latency optimized wires (*L-Wires*) that have high widths and spacing. Then, coherence messages are mapped to the appropriate set of wires taking into account their latency and bandwidth requirements, obtaining a reduction in both execution time and energy consumption for a CMP with a two-level tree interconnect topology. Unfortunately, the authors report insignificant performance improvements for the direct network topologies employed in tiled CMPs (such as a 2D-mesh).

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 L1 caches of a CMP to be classified into two groups: a) critical and short, and b) non-critical and long messages. The second approach uses a heterogeneous interconnection network comprised of only two types of wires: low-latency wires for critical messages and low-energy wires for non-critical ones.

The main contribution of our proposal is the partitioning of reply messages that carry data into a short critical message containing the sub-block of the cache requested by the core as well as a long non-critical 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 non-critical. The former can be sent through the *L-Wires* whereas the latter can be sent through the *PW-Wires*. Differently to proposals in [7], our partitioning approach first, eliminates a complex logic for choosing the correct set of wires (we need a single 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 direct topologies are used. Additionally, our proposal allows for a more balanced workload across the heterogeneous interconnect.

The rest of this paper is organized as follows. Our proposal for efficient message management in tiled CMPs is presented in section 2. Section 3 describes the evaluation methodology and presents the results of the proposed mechanism. Section 4 reviews some related work, and finally, section 5 summarizes the main conclusions of the work.

2 A Proposal for Efficient Message Management in Tiled CMPs

In this section we present our proposal for efficient message management in tiled CMPs. 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.

2.1 Tiled CMP Architectures

A tiled CMP architecture consists of a number of replicated *tiles* connected over a switched direct network (Figure 1). Each tile contains a processing core with primary caches (both I- and D-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. 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). In addition, the L2 cache tags store 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. In this paper we assume a process technology of 65 nm, tile area of approximately 25 mm^2, and a die size in the order of 400 mm^2 [2]. Note that this area is similar to the largest die in production today (Itanium 2 processor – around 432 mm^2). Note also that, due to manufacturing costs and form factor limitations, it would be desirable to keep die size as low as possible. Further details about the evaluation methodology and the simulated CMP configuration can be found in section 3.1.

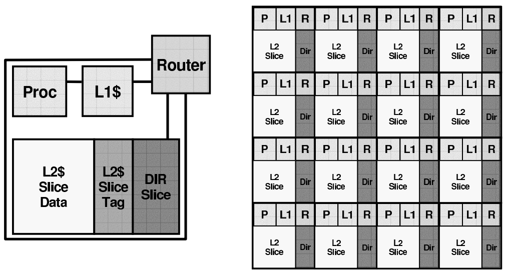

Fig. 1. Tiled CMP architecture overview

2.2 Classification of Messages in Tiled CMP Architectures

There are a variety of message types travelling on the interconnect of a CMP, each one with properties that are clearly distinct. In general, we can classify messages into the following groups: *Request messages*, that are generated by cache controllers in response to L1 cache misses and 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 they 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.

Messages involved in the L1 cache coherence protocol 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, otherwise the message is non-critical. As expected, delaying a critical message will result in longer L1 cache miss latencies. On the other hand, slight slowdowns in the delivery of non-critical messages will not cause any performance degradation.

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*. At the same time energy is saved, without affecting performance, when non-critical messages travel on slower, power-efficient *PW-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). 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 long messages.

Table 1. Area, delay and power characteristics of wire implementations (extracted from [7])

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
PW-Wire (8X plane)	$2x$	x	0.80α	0.2720

Table 1 shows the relative area, delay and power characteristics of *L-* and *PW-Wires* compared to baseline wires (*B-Wires*), as reported in [7]. A 65 *nm* process technology is considered, where 4X and 8X metal planes are used for global inter-core wires that are routed over memory arrays, as in [8]. It can be seen that *L-Wires* yield a two-fold latency improvement at a four-fold 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).

Regarding the power dissipated by each message type, Figure 2 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%). As previously commented, most of this power is dissipated in the point-to-point links and, therefore, message size plays a major role.

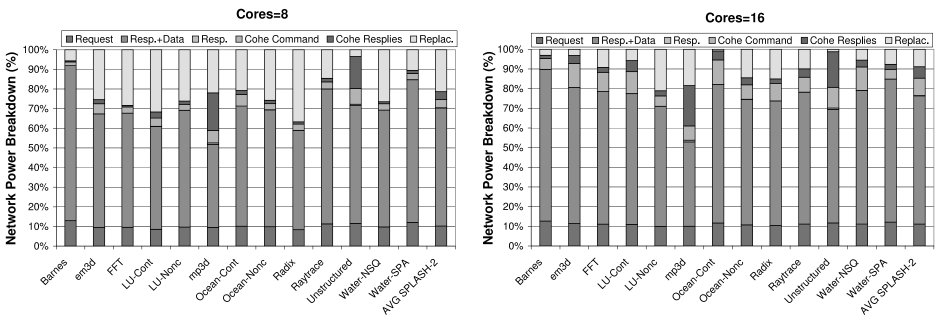

Fig. 2. Percentage of the power dissipated in the interconnection network by each type of message for an 8-core (left) and a 16-core CMP (right)

The use of a heterogeneous interconnect comprised 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 four-fold 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 using *PW-Wires*.

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

Table 2 summarizes the characteristics of each message type and points out the links that would be preferred in every case. In general, short messages are critical and, therefore, should be sent using *L-Wires*. On the other hand, long messages can be critical (responses with data) or non-critical (replacements with data), and the choice of wires is not clear. If we pursuit performance (criticality), then *L-Wires* might be the best choice. Contrarily, if energy savings are more important (length), then *PW-Wires* should be utilized. In this way, the policy of sending critical messages on *L-Wires* and non-critical 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. Differently to the policies proposed in [7], which are mainly based on message criticality, in this work we present an efficient message management mechanism based on taking advantage of both criticality and length properties simultaneously by means of *Reply Partitioning* as we will describe next.

2.3 *Reply Partitioning* for Decoupling Data Messages into Critical and Non-critical 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 on a 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 sub-block of the cache line that includes the word requested by the processor. And the second message, called *Ordinary Reply* (*OR*), is the original message and includes the full cache line.

This division of replies with data into *PRs* and *ORs* makes *all* critical messages short (note that *PRs* are critical since they contain the word requested by the processor) and, therefore, they can be sent using the low-latency *L-Wires.* At the same time, *all* long messages are non-critical (note that *ORs* become non-critical as the requested word also travels on a short message that hopefully will arrive sooner) and they can be sent using the power-efficient *PW-Wires* without hurting performance.

Additionally, splitting reply messages into a critical *PR* and a non-critical *OR* has slight implications on the coherence protocol. Recall that, in a non-blocking cache, MSHR registers are used to keep track of outstanding misses. In our mechanism, we have two different replies, and we need to define the actions required after the arrival of each one. Furthermore, with direct networks, the arrival order is not guaranteed and, although unlikely, the non-critical *OR* could be received before the critical *PR*.

When a *Partial Reply* arrives we are sure that all coherence actions 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 sub-block 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 no processed read requests and all the write requests, if any, are kept until the arrival of the *Ordinary Reply.* At the *OR* 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 *OR* before the *PR*, 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.

2.4 Interconnect Design for Efficient Message Management

As discussed previously, *L-Wires* have a four-fold 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 used by *PW-Wires* for sending long, non-critical messages.

In this work, we use the same main parameters for the interconnect as in [7]. 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-byte long since beside control information (3 bytes) they also carry address information (in the first two cases) or the sub-block of data of one word size (for *PRs*). On the other hand, coherence replies are just 3-byte long. Finally, *OR* 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 (600 *B-Wires*) 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 [7].

The resulting design is similar to that proposed in [7], 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 [7] require checking the directory state or tracking the congestion level in the network). Secondly, the routing logic and the multiplexers and de-multiplexers 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 3.2).

3 Experimental Results

This section shows the results that are obtained for our proposal and compares them against those achieved by two different configurations of the interconnect. The first is the configuration that employs just *B-Wires*, which is taken as baseline. The second configuration is an implementation of the 3-subnetwork heterogeneous interconnect proposed in [7] that uses *L-*, *B-* and *PW-Wires*.

3.1 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, called *Sim-PowerCMP* [9], that estimates both dynamic and leakage power and is based on RSIM [10]. 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, our simulation tool employs as performance simulator a modified version of RSIM that models the architecture of the tiled CMP presented in section 2. We have incorporated into our simulator power models already proposed and validated for both dynamic power (from Wattch [11]) and leakage power (from HotLeakage [12]) of each processing core, as well as the interconnection network (from Orion [13]). Further details about the implementation and validation of *SimPowerCMP* can be found in [9].

Table 3. Configuration of the baseline CMP architecture (left) and applications evaluated (right)

CMP Configuration	
Process technology	65 *nm*
Tile area	25 mm^2
Number of tiles	8, 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, 10+20 cycles
Coherence Protocol	MESI (with \$-\$ transfers)
Directory access time	10 cycles (tags access)
Memory access time	400 cycles
Network configuration	2D mesh (BW of 75 GB/s)
Router latency	1 cycle
Link width	75 bytes (*8X-B-Wires*)
Link latency/length	4 cycles / 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 × 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

Table 3 (left) shows the architecture configuration used along 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*. Reply messages are 67-byte 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-byte long (just 3-byte long for coherence replies).

Table 3 (right) shows the applications used in our experiments. *EM3D* and *Unstructured* are from the Berkeley suite, the rest of them are from the SPLASH/SPLASH-2 benchmark suites. Problem sizes have been chosen commensurate with the size of the L1 caches and the number of cores used in our simulations. 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 which 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.

In order to match the metal area of the baseline configuration, each original 75-byte unidirectional link (600 *B-Wires*) is designed to be made up of 88

L-Wires (11 bytes) and 248 *PW-Wires* (31 bytes) using the 8X metal plane. For comparison purposes, the 3-subnetwork heterogeneous interconnect described in [7] was also implemented. In that configuration, each link is comprised 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).

3.2 Simulation Results and Analysis

In this section, we report on our simulation results. First of all, 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 on the energy dissipated by the inter-core links. Finally, we report the energy and energy-delay^2 product (ED^2P) metrics for the full CMP. As in [7], all results have been normalized with respect to the baseline case where only *B-Wire*, unidirectional 75-byte wide links are considered.

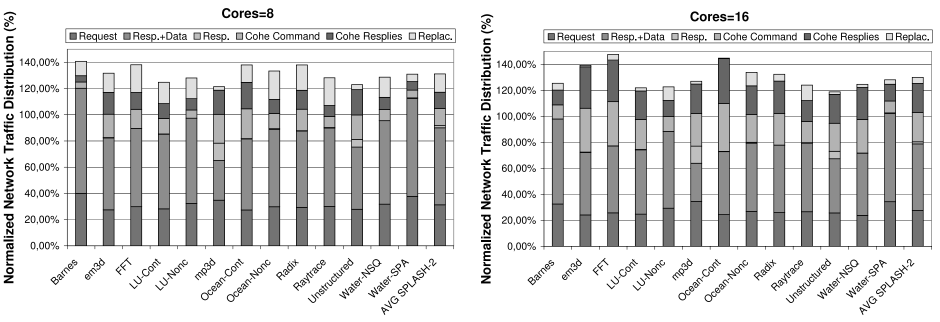

Fig. 3. Breakdown of the messages that travel on the interconnection network for an 8-core (left) and 16-core CMP (right) when an *L-Wire/PW-Wire* heterogeneous network is used and long critical messages are splitted

Figure 3 plots the percentage of each message type over the total number of messages sent in the baseline configuration for the 8- and 16-core configurations. It is important to note that *Reply Partitioning* increases the total number of messages that travel 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, we have observed that the number of messages increases around 30% on average. This extra traffic has been considered in all our evaluations.

Figure 4 plots the workload distribution between the different types of wires for both the 3- and 2-subnetwork heterogeneous interconnects. This figure is obtained 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, there is an underutilization of *L*- and *PW*- wires that leads to an imbalanced workload distribution when

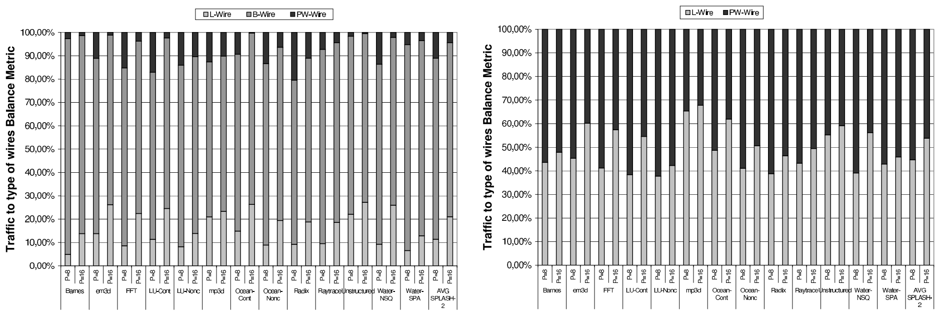

Fig. 4. Workload distribution for the 3-subnetwork approach (as in [7]) (left) and for the 2-subnetwork approach (right)

the 3-subnetwork configuration is used. However, the use of a 2-subnetwork interconnect, where *B-Wires* have been replaced by wider *L-Wires*, in conjunction with the *Reply Partitioning* technique, leads to a much more balanced workload distribution (Figure 4, right).

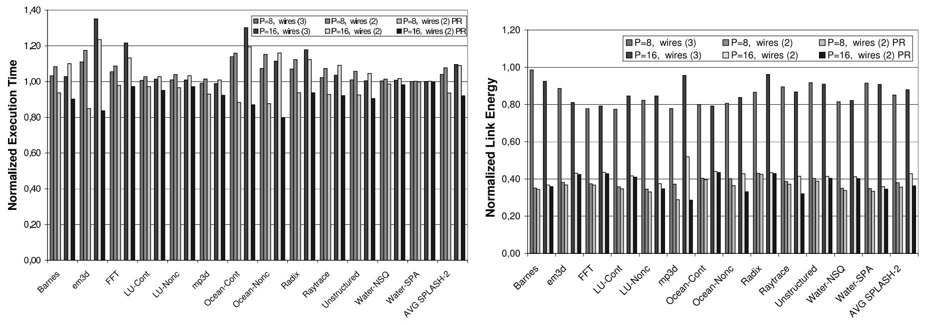

Fig. 5. Normalized execution time (left) and link energy (right) when heterogeneous links are used

Figure 5 (left) 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 (*P=8/16, wires (3)*) shows the normalized execution time for the 3-subnetwork interconnect (as in [7]). An average performance degradation of 4-8% is observed, which is a trend also reported in [7] 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 Figure 4. Similar results are obtained when a 2-subnetwork interconnect (*L-Wire/PW-Wire*) is considered without using the proposed *Reply Partitioning* mechanism, as shown by the second barline (*P=8/16, wires (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 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 40% 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, whilst in others, as Barnes or Ocean-Cont, it translates into significant performance degradation. This high variability is related with the access pattern and utilization of the cache lines. A sequential access pattern to the whole cache lines leads to a performance degradation whereas with a more irregular access pattern or an underutilization of the cache lines the increased latency is hidden. Finally, the third barline (*P=8/16, wires (2) PR*) shows the case when reply messages are split into critical, short *Partial Replies* (PR) and non-critical *Ordinary Replies*. On average, we observe performance improvements of 16% over the two previous options for a 16-core CMP as a direct consequence of the better distribution of the messages between *L-Wires* and *PW-Wires* that *Reply Partitioning* allows for. Again, high variability is found, with improvements ranging from 1-2% in some applications to 50-55% in other. Compared with the baseline configuration, where no heterogeneous network is used, an average performance improvement of 8% is obtained.

Figure 5 (right) plots the normalized link energy. Our proposed *Reply Partitioning* approach results in an average reduction of 60%-65% in the energy dissipated by the inter-core links. This reduction shows little variability. The ED^2P metric shows average improvements close to 75% although, in this case, the variability between applications is higher because in the ED^2P metric the execution time gains importance. Note also that, although *Reply Partitioning* increases the total number of messages that travel on the interconnect around 30% on average, the use of *PW-Wires* for sending long reply messages leads to important energy savings that overcome this drawback.

Finally, Figure 6 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 our proposal. The magnitude of these savings depends on the total number of cores of the CMP, ranging from 12% for the 8-core configuration to 17% for the 16-core configuration. On the other hand, when the ED^2P metric

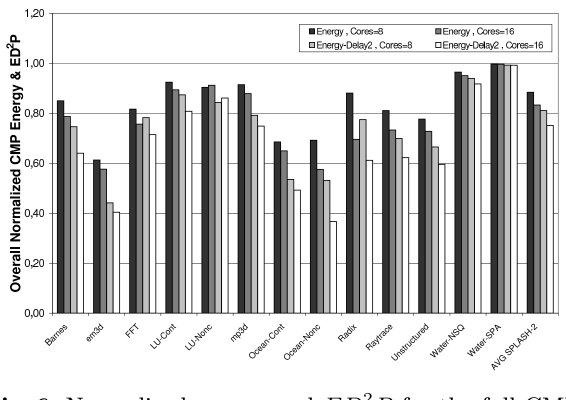

Fig. 6. Normalized energy and ED^2P for the full CMP

is considered, we find an increased improvement which ranges from 19% for the 8-core CMP to 25% for the 16-core one, due to bigger emphasis on the execution time.

4 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.* [8] 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 comprised of wires with varying area, latency, bandwidth and energy characteristics, and they apply it to register communication within a clustered architecture. Subsequently, they extend their proposal in [18] with new techniques aimed at accelerating cache accesses in large L2/L3 splitted caches by taking advantage of a lower-bandwidth, lower-latency network.

Very recently, Cheng *et al.* [7] 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.

Finally, Balfour and Dally [19] 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 is the best option. Differently from our work, the authors focus on the interconnection network design and obviate the cache coherence protocol (they assume an abstract communication protocol).

5 Conclusions

In this work we propose a performance- and energy-efficient message management mechanism for tiled CMPs that consists of two approaches. The first one is *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* 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 and a non-critical *Ordinary Reply* with the full cache block. The second approach of our proposal is the use of 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.

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% with an additional reduction in execution time of 8% over previous works. Finally, these reductions translate into overall CMP energy savings ranging from 12% for the 8-core configuration to 17% for the 16-core one (from 19% to 25% if the ED^2P metric is considered). 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. The authors would like to thank the anonymous reviewers for their helpful insights. This work has been jointly supported by the Spanish MEC and European Comission FEDER funds under grants "Consolider Ingenio-2010 CSD2006-00046" and "TIN2006-15516-C4-03".

References

1. Taylor, M.B., Kim, J., et al.: The Raw Microprocessor: A Computational Fabric for Software Circuits and General-Purpose Programs. IEEE Micro 22(2), 25–35 (2002)
2. Zhang, M., Asanovic, K.: Victim Replication: Maximizing Capacity while Hiding Wire Delay in Tiled Chip Multiprocessors. In: Proc. of the 32nd Int'l Symp. on Computer Architecture, pp. 336–345 (2005)
3. Wang, H., Peh, L.S., Malik, S.: Power-driven Design of Router Microarchitectures in On-chip Networks. In: Proc. of the 36th Int'l Symp. on Microarchitecture, pp. 105–111 (2003)
4. Magen, N., Kolodny, A.W., et al.: Interconnect-power dissipation in a microprocessor. In: Proc. of the 2004 Int'l Workshop on System Level Interconnect Prediction, pp. 7–13 (2004)
5. Shang, L., Peh, L., Jha, N.: Dynamic voltage scaling with links for power optimization of interconnection networks. In: Proc. of the 9th Int'l Symp. on High-Performance Computer Architecture, pp. 91–102 (2003)
6. Banerjee, K., Mehrotra, A.: A power-optimal repeater insertion methodology for global interconnects in nanometer designs. IEEE Trans. on Electron Devices 49(11), 2001–2007 (2002)

7. Cheng, L., Muralimanohar, N., et al.: Interconnect-Aware Coherence Protocols for Chip Multiprocessors. In: Proc. of the 33rd Int'l Symp. on Computer Architecture, pp. 339–351 (2006)
8. Kumar, R., Zyuban, V., Tullsen, D.M.: Interconnections in Multi-Core Architectures: Understanding Mechanisms, Overheads and Scaling. In: Proc. of the 32nd Int'l Symp. on Computer Architecture, pp. 408–419 (2005)
9. Flores, A., Aragón, J.L., Acacio, M.E.: Sim-PowerCMP: A Detailed Simulator for Energy Consumption Analysis in Future Embedded CMP Architectures. In: Proc. of the 4th Int'l Symp. on Embedded Computing, pp. 752–757 (2007)
10. Hughes, C.J., Pai, V.S., et al.: RSIM: Simulating Shared-Memory Multiprocessors with ILP Processors. IEEE Computer 35(2), 40–49 (2002)
11. Brooks, D., Tiwari, V., Martonosi, M.: Wattch: a framework for architectural-level power analysis and optimizations. In: Proc. of the 27th Int'l Symp. on Computer Architecture, pp. 83–94 (2000)
12. Zhang, Y., Parikh, D., et al.: HotLeakage: A Temperature-Aware Model of Subthreshold and Gate Leakage for Architects. Technical report, University of Virginia (2003)
13. Wang, H.S., Zhu, X., et al.: Orion: a power-performance simulator for interconnection networks. In: Proc. of the 35th Int'l Symp. on Microarchitecture, pp. 294–305 (2002)
14. Beckmann, B.M., Wood, D.A.: TLC: Transmission Line Caches. In: Proc. of the 36th Int'l Symp. on Microarchitecture, pp. 43–54 (2003)
15. Beckmann, B.M., Wood, D.A., et al.: Managing Wire Delay in Large Chip-Multiprocessor Caches. In: Proc. of the 37th Int'l Symp. on Microarchitecture, pp. 319–330 (2004)
16. Nelson, N., Briggs, G., et al.: Alleviating Thermal Constraints while Maintaining Performance via Silicon-Based On-Chip Optical Interconnects. In: Workshop on Unique Chips and Systems (2005)
17. Balasubramonian, R., Muralimanohar, N., et al.: Microarchitectural Wire Management for Performance and Power in Partitioned Architectures. In: Proc. of the 11th Int'l Symp. on High-Performance Computer Architecture, pp. 28–39 (2005)
18. Muralimanohar, N., Balasubramonian, R.: The Effect of Interconnect Design on the Performance of Large L2 Caches. In: 3rd IBM Watson Conf. on Interaction between Architecture, Circuits, and Compilers (P=ac2) (2006)
19. Balfour, J., Dally, W.J.: Design tradeoffs for tiled CMP on-chip networks. In: Proc. of the 20th Int'l Conf. on Supercomputing, pp. 187–198 (2006)