Reducing Sensitivity to NoC Latency in NUCA Caches

Pierfrancesco Foglia[*], Giacomo Gabrielli[*], Francesco Panicucci[†], Marco Solinas[*]
{foglia, giacomo.gabrielli, marco.solinas}@iet.unipi.it, f.panicucci@imtlucca.it
Members of the HiPEAC European Network of Excellence

Abstract

Non Uniform Cache Architectures (NUCA) are a novel design paradigm for large last-level on-chip caches which have been introduced to deliver low access latencies in wire-delay dominated environments. Typically, NUCA caches make use of a network-on-chip (NoC) to connect the different sub-banks and the cache controller. This work analyzes how different network parameters, namely hop latency and buffering capacity of routers, affect the overall performance of NUCA-based systems for the single processor case, assuming a reference NUCA organization derived from previous works. This analysis leads to some important design guidelines: first, the sensitivity of the system to the hop latency is very high and router architectures with long pipelines, suitable for throughput-oriented applications, are not adequate; second, limited buffering capacity is sufficient to achieve a good performance level. As a consequence, in this work we propose an alternative NUCA organization based on bank clustering that limits the average number of hops experienced by cache accesses. This organization is better performing in most of the cases and scales better as the cut-through latency increases, thus simplifying the implementation of routers.

1. Introduction

This paper presents an analysis of the impact of two main NoC parameters on the performance of NUCA caches, i.e. the cut-through latency of routers (the latency to deliver a traffic unit from an input channel to an output channel in a no-load condition) and their buffering capacity. While several implementations of NoCs have been proposed and adopted so far in other scenarios, such as system-on-chips [1] and tiled architectures [2, 3], NUCA caches are an emerging technology and, as far as we know, none of the previous studies clearly focused on the impact of the characteristics of the network routers on the system performance. The analysis described in this paper shows that different implementations of network routers can significantly affect the overall performance of single processor systems adopting a reference NUCA L2 cache, whose structure has been derived by previously proposed schemes [4, 5]. Taking into account the considerations derived from the performed analysis, we derive an alternative NUCA organization based on bank clustering that is better performing and it is able to reduce the strong sensitivity to the cut-through latency of routers.

1.1. NUCA caches

Non Uniform Cache Architectures have been proposed as a novel design paradigm for large last-level on-chip caches [4] in order to reduce the effects of wire delays, which significantly limit the performance scaling of today's high clock frequency microprocessors [6]. This is achieved by the adoption of a storage structure partitioned into sub-banks, with each sub-bank being an independently accessible entity, and by the adoption of a fast interconnection network to connect the banks and the cache controller. The access latency exhibited by a NUCA cache is a function of the physical location of the requested line. The mapping between cache lines and banks can be either static (S-NUCA) or dynamic (D-NUCA). In a D-NUCA cache a line can be located in one of a set of allowed bank locations, which collectively form a *bank set*, and each bank of the bank set behaves like a single way of a set associative cache [4]. Lines can dynamically migrate from one bank to another, provided that it belongs to the pertaining bank set, and the migration is triggered by a certain number of consecutive line accesses. Different implementation policies have been proposed for D-NUCA, i.e. mapping policies, line search policies and migration policies [4]; in order to keep the number of variable parameters reasonably low, in this study a specific set of policies has been selected for D-NUCA, leading to a configuration that is a good tradeoff between performance and complexity. The selected policies are: simple mapping, with each row of banks making up a bank set; broadcast search; promotion in the adjacent bank upon each hit (1 bank/1 hit).

1.2. Networks-on-chip for NUCA caches

A viable solution to connect the banks and the controller of a NUCA cache is represented by a NoC. The NoC paradigm tends to favour the reuse of design and

[*] Università di Pisa, Dipartimento di Ingegneria dell'Informazione, Via Diotisalvi 2, 56122 Pisa (Italy)
[†] IMT Lucca, Institute for Advanced Studies, Piazza S. Ponziano 6, 55100 Lucca (Italy)

verification efforts, which is particularly important for modern VLSI processes. In addition, the resulting interconnection scheme is more scalable than traditional approaches based on broadcast media, such as busses and rings. The intrinsic features of NUCA caches introduce constraints on the design of the NoC, in particular on the design of the network routers. These constraints impact on the characteristics of the network itself, such as topology, routing, and flow control, but, primarily, they are influenced by the way with which last-level on-chip caches are accessed by the CPU. A fundamental property of the NUCA on-chip network is that it is self-throttling [7], as it is common for processor-to-memory interconnects. In fact, non-blocking caches are able to support only a limited number of outstanding misses, therefore the number of simultaneous requests on the last-level cache is limited by the number of outstanding misses supported by the higher level. This number is determined by the number and size of the Miss Status Holding Registers (MSHRs) [8], which are used to keep track of the pending misses. From these considerations, we might expect the network traffic offered to the network to be quite moderate. Since the access latency is the fundamental performance metric of a NUCA cache, together with the hit rate, we also might expect latency, instead of bandwidth, to be the primary design goal for the switching elements of the network, in order to build fast NUCA caches. However, as far as we know, none of the previous studies put the emphasis on the impact of the router parameters on the performance of NUCA-based systems and it is not clear how the characteristics described above translate into constraints on the network design. Jin *et al.* [12] have focused on NoC-related aspects of NUCA, but in their work a fixed single-cycle router architecture is considered and the effects of this choice on the overall system performance are not reported; in this sense, our work can be considered as orthogonal to their study.

2. Methodology

The analysis described in this paper assumes a reference NUCA structure whose topology is derived from a 2D mesh, which will be called *partial 2D mesh* in the following, since only a subset of the links of a full 2D mesh are employed in order to reduce the area overhead (Figure 1). The sole injection point of the network is the L2 cache controller, which is assumed to be directly attached to the external DRAM controller.

The reference NoC architecture is based on a wormhole scheme, with routing and flow control policies working on a per-flit basis. The size of a flit is assumed to be equal to the link width, and the links are bidirectional. The routing scheme is deterministic, dimension ordered.

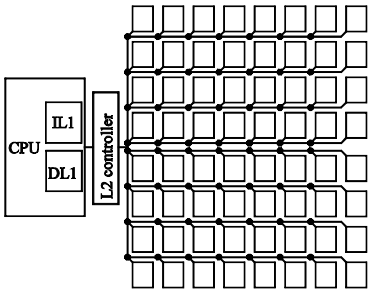

Figure 1. Partial 2D mesh topology. The NUCA structure represented here is made up of 64 banks (8x8). The black circles depict the network routers.

For the NUCA architectures considered in this work, a request packet is first propagated along the vertical dimension (vertical links in Figure 1), then it is propagated along the horizontal dimension (horizontal links in Figure 1); reply packets follow the same path. For D-NUCA caches, since a bank set is mapped to a single row of banks, first a flit has to reach the pertaining bank set, then it is propagated to the nodes attached to the banks of its bank set, starting from the nearest one to the cache controller. This causes the global access latency to raise as the distance of the requested cache line from the first node of the pertaining bank set increases.

The network routers are assumed to be input buffered and the buffers are managed on a per-flit basis in a FIFO manner; a crossbar switch is adopted to minimize contention on output channels; the flow control is credit-based. A detailed model of the router architecture described so far has been incorporated into the selected simulation platform.

Table 1. Configuration parameters for the CPU and the memory hierarchy

Parameter	Value
Technology node	65nm
CPU	Alpha 21264
Fetch/Issue/Commit width	4 / 4 int. + 2 f.p. / 11
Functional units	4 int. ALUs, 4 int. MUL/DIVs, 1 f.p. ALU, 1 f.p. MUL
Instr. L1 cache	64 KB, 2-way s.a., 64B line, 1 cycle hit lat.
Data L1 cache	64 KB, 2-way s.a., 64B line, 3 cycles hit lat., 2 ports
L1 caches MSHR size	8 entries, each points up to 4 targets
Main mem. latency	300 cycles
UCA L2 cache	8 MB, 4-way s.a., 64B line acc. time = 37, cyc. time = 4 (cycles)
S-NUCA L2 cache	8MB, 64 B line 32 banks (8x4), each one 4-way s.a. with acc. time = 13, cyc. time = 3 (cycles)
D-NUCA L2 cache	8MB, 64 B line 64 banks (8x8), each one direct mapped with acc. time = 11, cyc. time = 2 (cycles)

For this study, we selected three different L2 cache architectures, i.e. UCA, S-NUCA and D-NUCA; for each one, we selected the best performing configuration,

assuming a constant cache size fixed at 8 Mbytes and a line size of 64 bytes. The design space exploration for this step comprised several parameters: global associativity, bank associativity, number of banks and their organization in rows and columns. The configurations of the simulated systems are reported in Table 1.

The values of bank access latency and wire delay were obtained from CACTI 5.1 [9], which derives the technological parameters for devices and wires from the projections of the ITRS report [10].

The selected simulation platform is an extended version of *sim-alpha* [11], which is able to model NUCA cache architectures and the related NoC traffic with cycle-accurate fidelity. We selected L2 cache intensive applications from the SPEC CPU2000 and NAS Parallel Benchmarks suites (*applu*, *art*, *bt*, *bzip2*, *cg*, *equake*, *galgel*, *gcc*, *mcf*, *mesa*, *mgrid*, *parser*, *perlbmk*, *sp*, *twolf*) and we simulated a representative phase of each application; to identify the run phases we applied the same methodology as described in [4].

3. Results

Figure 2 shows the average IPC (Instructions Per Cycle) for the entire workload as the cut-through latency varies from 0 (when the hop latency is given only by the wire delay) to 5 clock cycles for 10 flits per channel buffer capacity. We can highlight that the overall system performance for NUCA is highly sensitive to the hop latency. While D-NUCA always outperforms S-NUCA, the performance of NUCA-based architectures rapidly decreases from a simulation node to the next. For 2 cycles cut-through latency, S-NUCA is less performing than UCA, while the benefits of employing a D-NUCA are poor (only 2.7% improvement over UCA). This high sensitivity witnesses that the NoC latency has strong effects on the overall system performance, while the latency of bank accesses becomes less influential as we move towards higher latencies for hops.

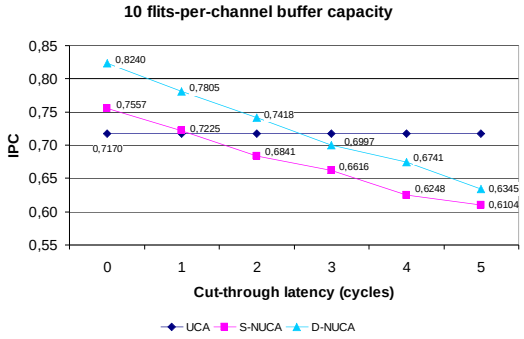

Figure 2. IPC vs. cut-through latency for 10 flits-per-channel buffer capacity

Focusing on a single value for hop latency, e.g. 2 cycles, it is possible to quantitatively evaluate the performance degradation due to limited buffer capacity with respect to the ideal router case (infinite buffer capacity), for both S-NUCA and D-NUCA. Figure 3 highlights this degradation, reporting the normalized IPC with respect to the ideal router case with infinite buffer capacity. The resulting performance degradation is negligible even for the 5 flits per channel buffer capacity; for both S-NUCA and D-NUCA the degradation is less than 1%.

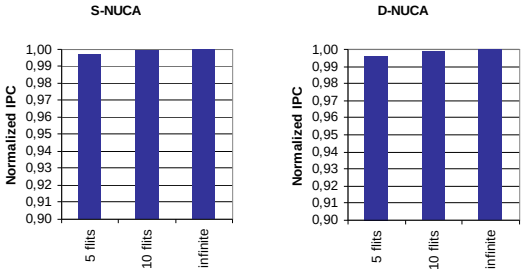

Figure 3. IPC vs. buffer capacity for 1 cycle cut-through latency.

The limited performance sensitivity to the amount of buffering resources may be explained by analyzing the average network traffic, in terms of buffer occupancy. Figure 4 shows the distribution of the buffer occupancy for two applications: we selected the *parser* benchmark, which exhibits a moderate load on the network, as witnessed by the utilization of the link that experiences the highest occupancy (the link is occupied for the 1.9% of the time), and the *gcc* benchmark, which experiences a relatively higher load (being the link with highest occupancy transmitting for the 15.9% of time). The configuration consists of a D-NUCA architecture, with single-cycle cut-through latency and infinite buffering capacity. The queue length distribution is shown for the router located at the injection point of the network, which shows the highest average queue length for all the applications, since this router has to propagate all the traffic generated by the cache controller. We selected the queue that experiences the highest average occupancy w.r.t. the other queues of the router. For the *gcc* benchmark, the queue length at the injection point is null (meaning that no buffering resources are occupied) in 90.43% of the time; the maximum measured queue length is 17 flits, but a queue longer than 5 flits is found with a very low frequency (less than 0.6% of the time), while a queue longer than 10 flits is found with a frequency lower than 0.001%. The *parser* benchmark experiences an even lower load condition, being the maximum measured queue length 7 flits, but with an occupancy of more than 5 flits being found only in the 0.002% of the time.

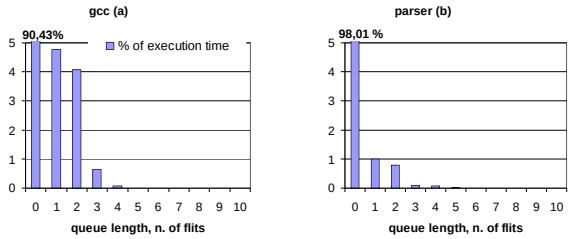

Figure 4. Distribution of buffer occupancy for gcc (a) and parser (b). The percentage of total execution time spent for each occupancy state is shown. The selected queues belong to the injection point of the NUCA on-chip network. Data refer to a D-NUCA architecture, with single-cycle cut-through latency and infinite buffering capacity.

4. Reducing sensitivity to NoC latency

One of the most effective ways to mitigate the high sensitivity to NoC latency is to reduce the average number of hops that cache accesses experience. This can be achieved by reducing the number of cache banks (assuming a constant cache capacity, this means that the size of banks is increased) or clustering the banks so that each cluster is attached to a network node, while keeping the bank size fixed. Since a wire-delay dominated environment put strong constraints on the topology, the only relevant scheme that we take into account for the clustered approach is a configuration with 4 banks per cluster, as depicted in Figure 5. The partitioning of the address space inside a single cluster is obtained by checking the least significant bits from the index field of the address. For D-NUCA caches, in order to achieve a significant improvement, we also introduced an alternative logical organization, which involves the way with which lines are mapped onto cache banks: with the clustered approach each bank set is mapped onto a row of clusters, and each column of clusters now behaves like a single way of a set associative cache.

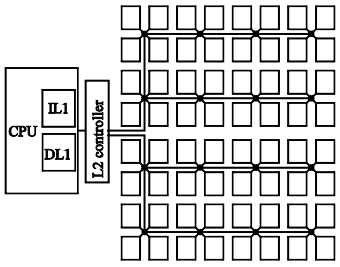

Figure 5. Partial 2D mesh topology with 4 banks per node (clustered approach).

Figure 6 reports the performance achieved by the new scheme, when applied to both S-NUCA and D-NUCA architectures. Except for the null cut-through latency case, the clustered scheme always outperforms the reference one. These results indicate that the minimal cut-through latency constraint can be relaxed, as this configuration is much more scalable w.r.t. the reference architecture.

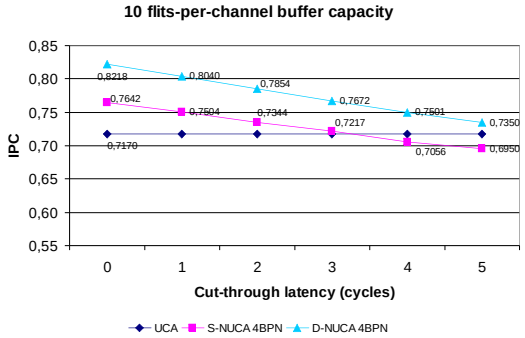

Figure 6. IPC vs. cut-through latency for the clustered scheme (4BPN = 4 banks per node).

Even for this architecture, the considerations about buffer occupancy are the same as for the non-clustered approach: the system exhibits very poor sensitivity to the buffer capacity of routers.

The clustered scheme, while being better performing, reduces the number of network routers, thus leading to a simpler implementation. The additional overhead is given by the additional ports to connect each router to its local banks (3 additional ports w.r.t. the traditional scheme). However, a solution based on the multiplexing of a single port to connect to the local banks could be used, employing a simple arbiter. We performed a set of simulations which indicated that the performance degradation due to the loss of parallelism introduced by this solution is negligible: for instance, for a D-NUCA with a single-cycle cut-through latency and infinite buffering capacity, the performance degradation is only 0.26%.

5. Acknowledgements

We wish to thank the anonymous reviewers for their helpful and valuable comments. We also wish to thank Stephen Keckler who furnished us with the initial version of the modified sim-alpha simulator, José Duato for his suggestions on our work, and Cristian Croce for helping us in the development of the simulation platform.

This work is partially supported by the SARC project funded by the European Union under contract no. 27648.

6. References

[1] L. Benini and G. De Micheli. Networks on chips: a new SoC paradigm. *IEEE Computer*, 35(1):70–78, 2002.

[2] S. Vangal et al. An 80-tile 1.28TFLOPS network-on-chip in 65nm CMOS. In *Digest of Technical Papers, International Solid-State Circuits Conference* (ISSCC), pages 98–589, 2007.

[3] K. Sankaralingam et al. Distributed microarchitectural protocols in the TRIPS prototype processor. In *Proceedings of the 39th International Symposium on Microarchitecture* (MICRO), pages 480–491, 2006.

[4] C. Kim, D. Burger, and S. W. Keckler. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. In *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems* (ASPLOS), pages 211–222, 2002.

[5] A. Bardine, P. Foglia, G. Gabrielli, C. A. Prete, and P. Stenstrom. Improving power efficiency of D-NUCA caches. *ACM SIGARCH Computer Architecture News*, 35(4):53–58, 2007.

[6] V. Agarwal, M. S. Hrishikesh, S.W. Keckler, and D. Burger. Clock rate versus IPC: the end of the road for conventional microarchitectures. In *Proceedings of the 27th International Symposium on Computer Architecture* (ISCA), pages 248–259, 2000.

[7] W. J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2003.

[8] D. Kroft. Lockup-free instruction fetch/prefetch cache organization. In *Proceedings of the 8th Annual Symposium on Computer Architecture* (ISCA), pages 81–87, 1981.

[9] S. Thoziyoor, N. Muralimanohar, and N. P. Jouppi. CACTI 5.0. Technical report, HP, 2007.

[10]International Technology Roadmap for Semiconductors, 2005 Edition Report.

[11]R. Desikan, D. Burger, S. Keckler, and T. Austin. Sim-alpha: a validated, execution-driven Alpha 21264 simulator. Technical report, Department of Computer Sciences, University of Texas at Austin, 2001.

[12]Y. Jin, E.J. Kim, and K.H. Yum. A domain-specific on-chip network design for large scale cache systems. In *Proceedings of the 13th International Symposium on High Performance Computer Architecture* (HPCA), pages 318–327, 2007.