A Lightweight Network of IDs to Quickly Deliver Simple Control Messages

Mario Lodde and José Flich

Universitat Politècnica de València, Spain

Abstract. Most of the network traffic in a Chip Multiprocessor (CMP) is due to messages exchanged by the caches according to the cache coherence protocol. Different types of messages have different requirements as far as latency and bandwidth are concerned, so the Network-on-Chip (NoC) should be tailored to fit the needs of each class of messages to maximize overall chip performance.

In this work we extend the Gather Network, a recently proposed dedicated lightweight network which is used for the transmission of many-to-one acknowledgement messages, and adapt it to transmit unicast acknowledgements, obtaining a dedicated network which we call Network of IDs. Evaluation results with Splash-2 and PARSEC applications show that the Network of IDs is able to deliver unicast acknowledgements in a few clock cycles, relieving the NoC from 4.5% of the traffic on average and reducing total energy by 4% on average and up to 8% for some applications.

1 Introduction

Current chip multiprocessors include tens of cores in a single chip, typically organized in a regular structure. A tile-based design is particularly appealing, since design efforts can be focused on a single modular component which is then replicated to build the final system. Figure 1 show the sample system we assume in this work: a 16-core CMP is built replicating a tile which includes a core, a private L1 cache, a bank of L2 cache and a network switch. The Network-on-Chip (NoC) used to connect the tiles is a 4×4 2-D mesh.

Different cores communicate through shared memory, thus the necessity to implement a cache coherence protocol to keep coherent the data stored in the on-chip cache hierarchy. Since most of the traffic in the NoC is due to messages injected by the caches according to the coherence protocols, the NoC and the coherence protocol should be co-designed to achieve the best chip performance, adapting the NoC to meet the requirements of the traffic pattern of coherence messages and tailoring the coherence protocol to efficiently use the resources provided by the NoC.

Various research works have been published in the near past tackling these challenges, which are becoming more and more critical as the number of cores in CMP systems increases. Typically, coherence messages are classified depending on the size: on the one hand there are long data messages, which are sent

D. an Mey et al. (Eds.): Euro-Par 2013 Workshops, LNCS 8374, pp. 477–487, 2014.

answering a block request or when a dirty cache line is replaced in L1; on the other hand there are short control messages, such as block requests, invalidation messages and acknowledgements. If the NoC flit size is set to be equal to the short message size, there are no bandwidth issues as far as short messages are concerned and design efforts can focus on reducing the network latency; data messages however will not fit in a flit and their latency will be higher. If the flit size is as wide as the long message size, data messages will fit in a flit but network resources will be underutilized when a short message is transmitted.

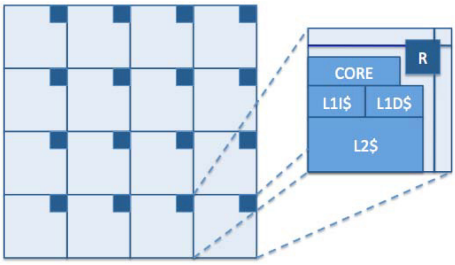

Fig. 1. Tiled CMP system

Another classification can be done depending on whether a coherence message is unicast (e.g. data messages) or it involves collective communication, both one-to-many (e.g. invalidation messages sent to the sharers or broadcast requests used in Dir_0B protocols like Hammer [1]) and many-to-one (e.g. invalidation acknowledgements).

Messages can be classified also depending on the amount of information they deliver: a block request message, for instance, must specify at least the requested block's address, the ID of the requestor and of destination node and the request type (load or store); a data message must specify at least the block address, the destination node's ID, the data type (shared or exclusive) and must contain the data block. All these messages thus need to be mapped to a packet structure which includes proper fields to store that information. Other coherence messages however do not deliver any information at all: if we consider invalidation acknowledgements which are sent by the sharers to the requestor of a block after a write request, the only field which is really needed is the ID of the destination node; if we consider in-order cores, even the block address is redundant since the requestor is blocked waiting for the reception of the acknowledgments. However, these messages are typically transmitted through the NoC using the same short message structure of requests, thus wasting NoC resources and power to transmit a packet in which most of the fields are not used or redundant. Basing on these considerations, we recently proposed a lightweight control network, called the Gather Network [4], used to collect many-to-one acknowledgements in invalidation-based coherence protocols. In this paper we extend the Gather Network and adapt it to transmit unicast acknowledgements (e.g. writeback

acknowledgements or directory unblock messages). Evaluation results of our proposal with various applications of the Splash-2 and PARSEC benchmark suites show that unicast acknowledgements are delivered in a few clock cycles (2.5 on average) and the traffic in the regular NoC is reduced by 4.5% on average, leading to an average 4% reduction of total NoC energy.

The rest of this paper is organized as follows: in Section 2 we briefly describe the Gather Network, in Section 3 we extend the Gather Network to allow the transmission of unicast messages, in Section 4 we provide some evaluation results, then Section 5 describes the related work and in Section 6 we draw the conclusions.

2 Gather Network

The Gather Network is a dedicated control network that can transmit, gather and deliver simple control messages. It was specifically designed to manage many-to-one acknowledgements generated by cache coherence protocols in tiled CMP systems. These messages are very simple and the information they include is implicit in the message itself: since the destination node is blocked while it's waiting to receive all the expected acknowledgement messages, the only necessary information these messages need to include is the destination node address. In its first implementation [2], [3] the Gather Network was used to transmit, gather and deliver many-to-one acknowledgements sent by L1 caches in broadcast-based coherence protocols like Hammer when a request is broadcasted by the home L2 bank; since each request is broadcasted to all L1 caches, the block requestor always receives N-1 acknowledgements, where N is the number of cores in the system. This first implementation used a very simple logic providing a very low latency and an extremely low area overhead. The whole Gather Network is composed by a set of subnetworks, each subnetwork being associated to a different tile.

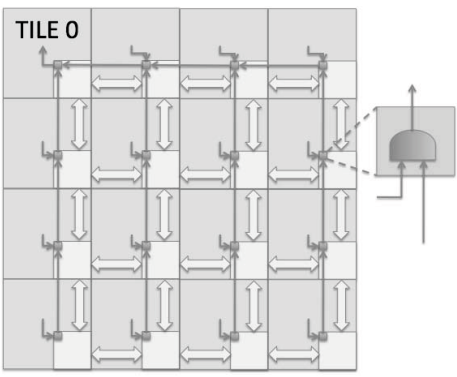

Fig. 2. Logical view of the Gather Network for tile 0

Figure 2 shows the structure of the subnetwork associated to the upper-left tile (Tile 0[1]): it is basically a tree of AND gates, with Tile 0 being the destination at the root of the AND tree and all the other tiles at the leaves. When a tile wants to send an acknowledgement to Tile 0, it just has to set its output of the subnetwork associated to Tile 0; once all the tiles have set their outputs, the output of the AND tree will be set and Tile 0 is notified of the reception of all the acknowledgements. This completely combinational design has been implemented with Synopsis DC and the area overhead of the whole Gather Network resulted to be 2.3% of the area of a single switch, while its critical path resulted to be lower than the slowest module of a switch or lower than twice the latency of the slowest module of a switch depending on the design parameters, meaning that an acknowledgement message can be transmitted in one or two clock cycles respectively between two opposite corners of the CMP.

This simple implementation has some drawbacks:

- since it requires one subnetwork for each tile, its wiring requirements grow with the number of tiles, thus limiting its scalability to a system with tens of tiles (for larger systems the size of the Gather Network would be comparable to the size of the regular NoC).
- this design only allows one pending request for each core; out-of-order cores which can have up tp N outstanding requests would require N subnetwork, thus limiting the scalability.

To address these issues, we implemented the logical behavior of the combinational implementation with sequential logic [4]. This second version of the Gather Network uses sequential modules at each switch, which communicate transmitting the ID of the acknowledgement's destination node, thus improving the scalability since the wiring requirements are reduced to the logarithm in base two of the number of tiles in the CMP.

Both the sequential and the combinational implementations however can be used without coherence protocol modifications only with broadcast-based protocols, since all nodes participate in every acknowledgment phase; this is not true in directory-based coherence protocols: in that case, only the nodes who hold a shared copy of a block receive the invalidation message and answer with the acknowledgement message, so an additional logic is needed to configure the Gather Network and enable only the signals generated by those nodes [3]; the configuration is done through the multicast invalidation message sent by the home L2 bank, and after the configuration the tree structure of the enabled outputs reflect the same path of the multicast message, but in opposite direction. This means that the tile which receives the acknowledgments must be the same tile which issued the multicast message, so the default directory protocol must be slightly modified to be able to use the Gather Network.

In this work we assume the sequential implementation of the Gather Network and adapt the modules to transmit unicast acknowledgements; this allows to use the Gather Network for acknowledgements sent by the home L2 bank as

[1] Tiles are numbered 0 to 16 starting from the upper-left corner of the chip and descending row by row.

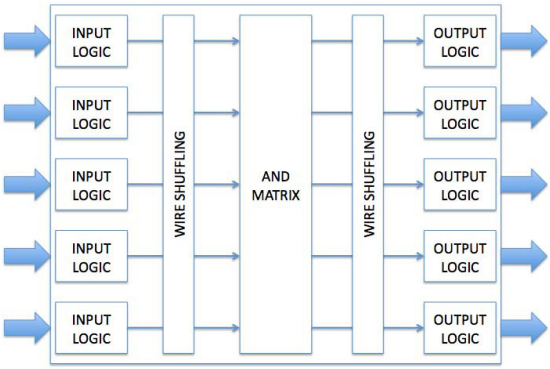

Fig. 3. Network of ID's module at each switch

answer to a writeback message (replacement in the L1 cache) and for invalidation acknowledgements in directory-based coherence protocols without any protocol modification. Since the use of this improved version is not limited to gather operations, we'll call it our Network of IDs.

3 Network of IDs

The Gather Network sequential module can be used to transmit unicast acknowledgements after some simple modification. Figure 3 shows the main structure of this module: the messages coming from neighboring modules or from the local node are processed by an input logic which decodes the destination ID. The decoded signals are grouped by destination node and sent to the AND matrix, where the collective acknowledgement is generated and sent to the proper output port. The structure of the messages exchanged by the Network of ID's modules is shown in Figure 4: for a 4×4 system, four bits are used to codify the destination node's ID and an additional bit indicates whether the message is a unicast acknowledgement or not.

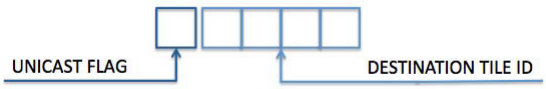

Fig. 4. Network of ID's message structure

Figure 5 shows the main blocks more in detail: at the Input logic the ID is separated from the unicast flag and decoded through a demultiplexer. The AND matrix has an AND gate for each node in the system; signal coming from different ports and destined to the same node are grouped at the same AND gate, then the output of each AND gate is ORed with the unicast flag and the resulting signal is sent to the proper output port, where it is encoded and transmitted to the next hop.

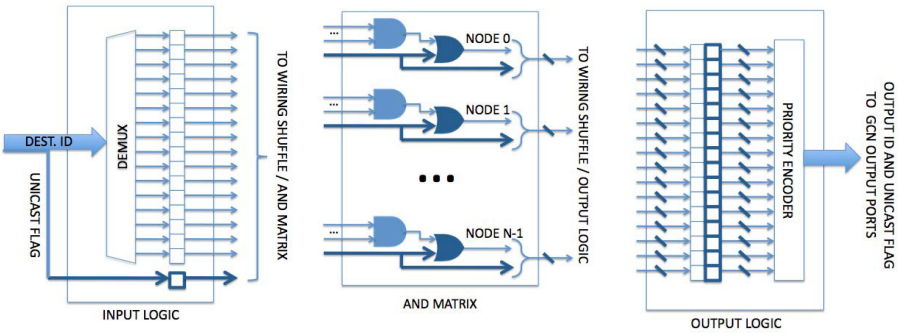

Fig. 5. Input logic, AND matrix and Output logic schemes

The additional logic needed by the network of IDs is marked with thicker lines: at the input logic, the unicast flag is decoupled from the destination node ID and sent to the AND matrix, where it is used to bypass the output signal of the AND gate correspondent to the destination node; at the output ports, an additional bit per destination is added to the registers to mark whether the outbound message must have the unicast flag set or not.

The module has been implemented using the 45nm technology open source Nangate [5] library with Synopsys DC. Cadence Encounter has been used to perform the Place&Route. The additional logic increases the modules area by 47% (the area of the final design is $0.610 \times 10^{-3} mm^2$), which means that each Network of IDs module introduces a 3.38% area overhead compared to the basic NoC switch we assumed in this work.

4 Evaluation

In this Section we provide some evaluation results of the Network of IDs. We implemented the NoC, the Network of IDs and the cache hierarchy of a 4×4 CMP system using our in-house cycle-accurate NoC and cache hierarchy simulator. Since our simulator is trace-driven, we used Graphite [6] and Sniper [7] multi-core simulators to capture the memory accesses of the cores when executing different applications of the Splash-2 and PARSEC benchmark suites. We assumed a MESI directory-based coherence protocol; network and cache parameters are shown in Table 1.

We compared a baseline case in which all messages are sent through the regular NoC and our proposal in which the Network of IDs is used to transmit invalidation and writeback acknowledgements while the NoC is used for all other messages.

Figure 6 shows the load and store latency of our proposal, normalized to the baseline. Load and store latencies are just slightly reduced when the Network of IDs is used, due to two reasons:

Table 1. Network and cache parameters

Routing	XY	Coherence protocol	Directory (MESI)
Flow control	credits	L1 cache size	16 + 16 kB (I + D)
Flit size	8 byte	L1 tag access latency	1 cycle
Switch model	4-stage pipelined	L1 data access latency	2 cycles
Switching	virtual cut-through	L2 bank size	256 kB
Buffer size:	9 flit deep	L2 tag access latency	1 cycle
Virtual channels:	4	L2 data access latency	4 cycles
GCN module delay	1 cycle	Cache block size	64 B

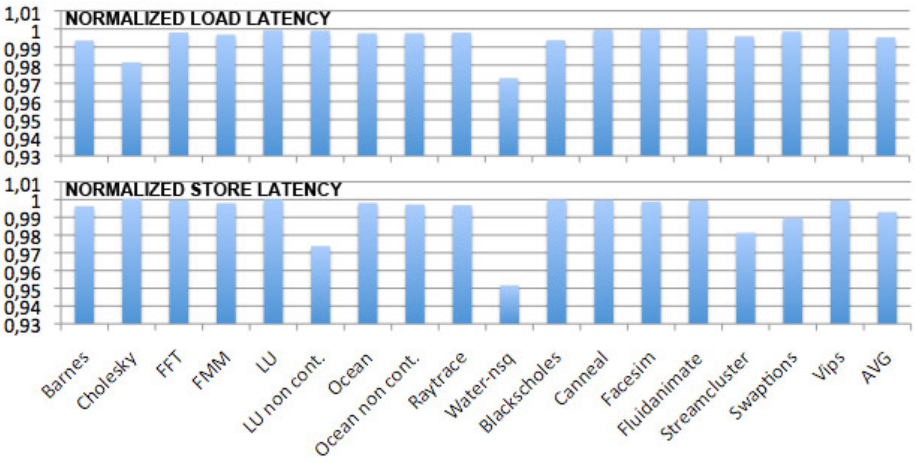

Fig. 6. Normalized load and store latency

- The number of invalidations in the benchmarks we simulated is less than 0.9% of the total number of messages on average, so the impact of optimizing their acknowledgement's transmission on the overall performance is limited.
- The writeback acknowledgments are not on the critical path of load and store requests, so optimizing their transmission only has a marginal impact on load and store latencies.

This implies that the Network of IDs has a negligible impact on execution time for the applications we simulated.

Figure 7 shows the normalized number of messages injected in the NoC. Using the Network of IDs, the NoC is relieved from a percentage of messages which ranges from 17% to 25% depending on the application (21% on average).

However, since the messages transmitted through the Network of IDs are short messages, the NoC traffic reduction in terms of injected flits is lower, as shown in Figure 8: NoC traffic is reduced by a percentage which ranges from 3.5% to 6%, with an average NoC traffic reduction of 4.5%.

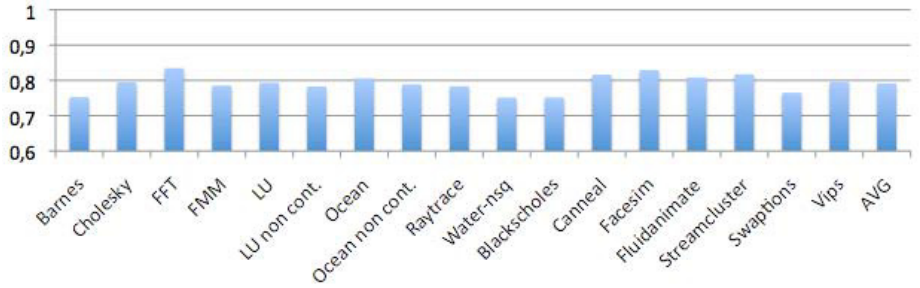

Fig. 7. Normalized number of injected messages

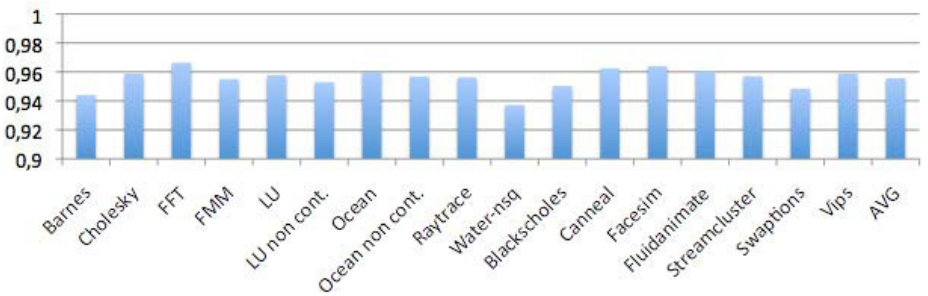

Fig. 8. Normalized number of injected flits

Figure 9 shows the average latency of the messages transmitted through the Network of IDs. On average, the Network of IDs is able to deliver a message in 2.5 clock cycles, and the variability of this value for each application compared to the average is quite low.

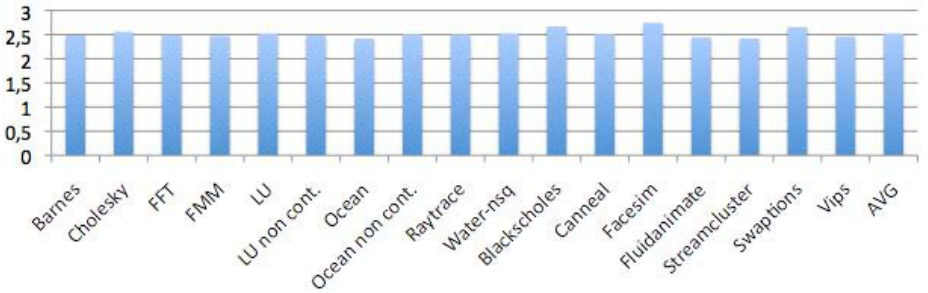

Fig. 9. Average ID network latency

In Figure 10 we show the number of contention events at the output ports of the Network of IDs modules, divided by the total number of messages transmitted through the Network of IDs. We can notice an extremely low contention: on average, one message each 2000 experiences contention at the output port, the worst case being that of LU non contiguous, in which contention is also very low

anyway (one message out of 555 experiences contention). Since sending unicast acknowledgements introduce a very low load on the Network of IDs, we believe it can also be used with a broadcast-based coherence protocol, where the total traffic of many-to-one acknowledgements gathering at the Network of IDs modules (behaving as a Gather Network) and unicast acknowledgements would not stress excessively the Network of IDs degrading its performance. We leave this evaluation for future work.

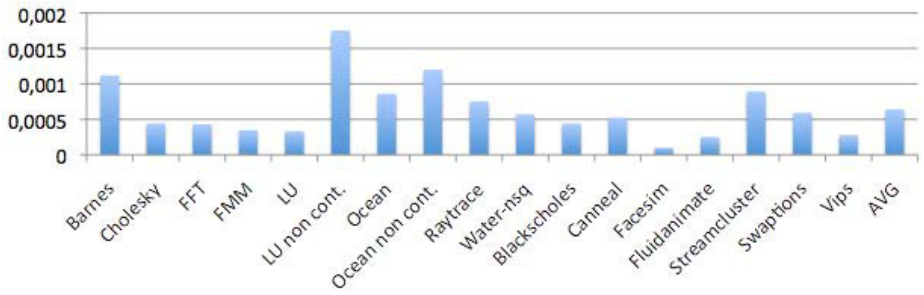

Fig. 10. Number of conflicts per message

Finally, we used Orion 2 [8] to evaluate the NoC energy savings due to the Network of IDs; as shown in Figure 11, the NoC dynamic energy is reduced by 6% on average, which, added to the static energy consumed during the whole application execution, gives a total energy reduction by 4% on average and up to 8%. Energy results include the energy consumption of both the regular NoC and the Network of IDs. The latter has been calculated basing on the power requirements of a single module obtained using Cadence Encounter, and resulted to be less than 0.1% of the total NoC energy for all the applications.

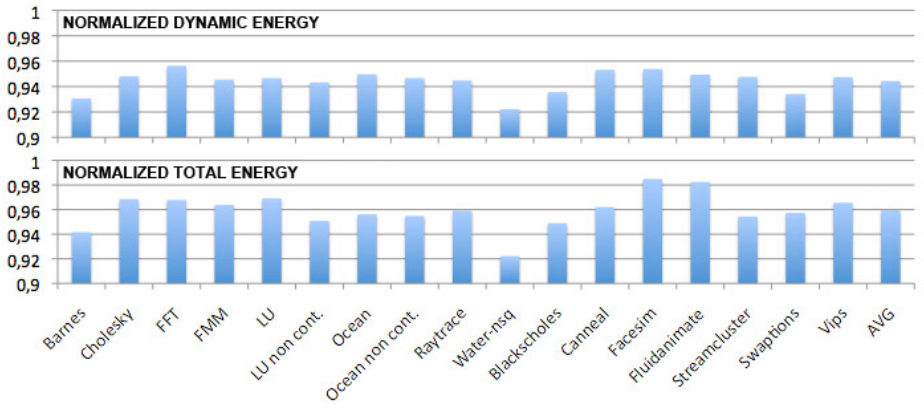

Fig. 11. Normalized energy

5 Related Work

Several works have appeared recently tackling the challenges of optimizing the NoC to meet the demands of the coherence protocol.

Recent proposals to tackle collective communication include hardware NoC mechanisms to optimize multicasts and broadcasts [9] injecting a single message in the NoC and then replicating the message at the switches to reach all the destination nodes, and mechanisms to collect many-to-one messages at the switches to avoid the serialization of those messages at the destination node [10].

Some proposals tightly couple the caches with the interconnect, embedding the cache structure into the network fabric: an example is In-Network Cache Coherence, proposed by Eisley et al. [11], where the directory information is from the cache banks at the nodes into virtual trees within network switches and leave to the network the task to route a request to the nearest nodes holding a copy of the requested data.

Differentiating short control messages and long data messages, Bolotin et al. [12] proposed a priority-based NoC which gives higher priority to short messages in order to reduce cache access latency. Their NoC design also optimizes broadcast short messages and provides support for synchronization.

All these proposals however still use the NoC resources for acknowledgement messages, which actually don't need the information contained in a short message packed besides the destination node's ID. The Network of IDs, explicitly designed for this class of messages, speeds-up the message transmission providing a lightweight structure with minimal resources.

6 Conclusions and Future Work

In this work we proposed the Network of IDs, a lightweight dedicated network added to the regular NoC and used for the transmission of unicast acknowledgement messages. The network has been obtained by adapting the modules of the previously proposed Gather Network. The resulting Network of IDs can deliver unicast acknowledgements in a few clock cycles (2.5 on average for the applications we simulated) and introduces a very low overhead in terms of area and wiring requirements. Used in a system which employs a directory-based coherence protocol, it reduces the NoC traffic by 4% on average, thus reducing also NoC energy by 4%.

In the future we plan to use the Network of IDs with a broadcast-based coherence protocol like Hammer, where it would be used for many-to-one acknowledgement messages and unicast writeback messages. Another future work direction is to employ the Network of IDs to provide hardware support to threads synchronization events.

Acknowledgements. This work has been supported by the VIRTICAL project (grant agreement n 288574) which is funded by the European Commission within the Research Programme FP7.

References

1. Owen, J.M., et al.: United states patent: 7069361 - System and method of maintaining coherency in a distributed communication system (June 2006)
2. Lodde, M., et al.: Heterogeneous NoC design for efficient broadcast-based coherence protocol support. In: NOCS 2012 (2012)
3. Lodde, M., et al.: Heterogeneous network design for effective support of invalidation-based coherency protocols. In: INA-OCMC 2012 (2012)
4. Lodde, M., et al.: Built-in fast gather control network for efficient support of coherence protocols. IET-CDT INA-OCMC 2012 Special Issue (2012)
5. The Nangate Open Cell Library, 45nm FreePDK available online at `https://www.si2.org/openeda.si2.org/projects/nangatelib/`
6. Miller, J.E., et al.: Graphite: A distributed parallel simulator for multicores. In: HPCA 2010 (2010)
7. Carlson, T.E., et al.: Sniper: Exploring the level of abstraction for callable and accurate parallel multi-core simulations. In: SC 2011 (2011)
8. Kahng, A., et al.: Orion 2.0: A power-area simulator for interconnection networks. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 20(1) (2012)
9. Rodrigo, S., et al.: Efficient Unicast and Multicast Support for CMPs. In: MICRO 2008 (2008)
10. Krishna, T., et al.: Towards the ideal on-chip fabric for 1-to-many and many-to-1 communication. In: MICRO 2011 (2011)
11. Eisley, N., et al.: In-Network Cache Coherence MICRO 2006 (2006)
12. Bolotin, E., et al.: The Power of Priority: NoC Based Distributed Cache Coherency. In: NOCS 2007 (2007)