Towards Hierarchical Cluster based Cache Coherence for Large-Scale Network-on-Chip

Yuang Zhang[1,2], Zhonghai Lu[2], Axel Jantsch[2], Li Li[1], Minglun Gao[1]

[1]Institute of VLSI Design, Nanjing University
Key Laboratory of Advanced Photonic and Electronic Materials,
Nanjing University
Nanjing, China
Email: {zhangyuang, lili, gaominglun} @ nju.edu.cn

[2]Department of Electronic, Computer and Software Systems
School of Information and Communication Technology
Royal Institute of Technology
Stockholm, Sweden
{yazhang, zhonghai, axel} @ kth.se

***Abstract*—We introduce a novel hierarchical cluster based cache coherence scheme for large-scale NoC based distributed memory architectures. We describe the hierarchical memory organization. We show analytically that the proposed scheme has better performance than traditional counterparts both in memory overhead and communication cost.**

Keywords: Cache coherence, Hierarchical directory, Network cluster, Network-on-Chip

I. INTRODUCTION

Network-on-Chip (NoC) is considered to be a promising scheme for future MPSoCs. NoC can provide high bandwidth, low latency, parallel communication and scalable design exploration space [1, 2]. The increasing integration density will allow NoCs to integrate more and more components. The memory system will consume large part of on-chip area. The organization of memory system will greatly affect the performance of NoC. There exist plenty of shared data among nodes. Because of the inherent contention in accessing shared recourses, it may incur long communication latencies, which will limit the performance of NoC.

Introduction of private caches will be helpful in reducing average latencies [3]. Especially in the NoC context, caches also increase memory effective utilization and reduce communication volume. But a new problem emerges. Multiple copies of the same data block can exist in different caches. If processors are allowed to update their own copies freely, inconsistent view of the memory is unavoidable, leading to program malfunction. This is the well-known cache coherence problem [4, 5].

In order to provide a coherent view of memory, the hardware must track where the latest version of any particular memory address can be found, and refresh the data when a load from the memory address occurs. The cache coherence scheme should efficiently support the communication of large numbers of small, cache-line-sized packets of data between processors. All this must be done with minimal latency, since individual load and store instructions are dependent upon each communication event. The cache coherence solution is not only necessary for correct program execution, but it can also have a very significant impact on system performance. Therefore it is important to design cache coherence solutions as efficiently as possible.

In bus-based MPSoCs, the snoopy protocol is suitable. It makes use of the broadcasting and serialization properties of buses, resulting in a cheap solution. However, coherence actions on the shared bus additionally increase the bus traffic and make the bus saturation more acute. Only systems with small to medium number of processors can be efficiently supported by snoopy protocols. It is not a scalable cache coherence solution for network-based multiprocessors.

Directory-based cache coherence protocol is a more scalable scheme [5, 6, 7]. The global system-wide status information relevant for coherence maintenance is stored in a directory. It is suitable for multiprocessors with general interconnection networks. However, there still exist weaknesses in this scheme. The first is the memory overhead of directory which will limit the system scalability. The second is long transaction delay, especially in write invalidation round trips. There also exists the risk of traffic contention and hot spots, which will further decrease the performance. A cache coherence protocol, which can efficiently utilize the high bandwidth of a NoC and provide good scalability and low latency, is important in order to improve the system performance.

The rest of this paper is organized as follows. Section 2 analyzes and compares traditional directory based cache coherence schemes. Section 3 proposes a hierarchical cluster based scheme for NoCs. Section 4 discusses the performance and scalability of the proposed cache coherence scheme. Section 5 describes related work in hierarchical directory based cache coherence protocols in traditional multiprocessor field and recent work of cache coherence schemes for NoCs. Section 6 concludes this paper and proposes future work.

II. FLAT DIRECTORY BASED CACHE COHERENCY

Traditional flat directory protocols [4] fall under three primary categories: *full-map directory*, *limited directory*, and *chained directory*.

The full-map protocol uses directory entries with one bit per processor and a dirty bit. Each bit represents the status of the block in the corresponding processor's cache (present or absent). If the dirty bit is set, then one and only one processor's bit is set, and that processor has permission to write into the block. The full-map protocol can provide a good performance of centralized directory-based cache coherence. However, it is not scalable with respect to memory overhead, which increases with $O(N^2)$, where N is the number of processors.

Limited directory protocols are designed to solve the directory size problem. Restricting the number of simultaneously cached copies of any particular block of data limits the growth of the directory to a constant factor. The directory size increases with $O(N \times \log_2 N)$. However, limited directory based protocols suffer from extraneous cache invalidating and updating due to insufficient pointers.

The chained directory realizes the scalability of limited directory without restricting the number of shared copies of data blocks. However, it takes precious memory space from cache and the invalidation delay linearly depends on the number of copies.

For large-scale shared-memory NoCs, both the memory overhead and communication delay will become unacceptable. And the traditional schemes do not make good use of high communication bandwidth of NoCs. A novel cache coherence protocol that efficiently utilizes the underlying communication bandwidth of NoC should be explored.

III. Hierarchical Cluster Based Cache Coherency

A. *Hierarchical directory organisation*

To decrease both the memory overhead and communication latency, we suggest a hierarchical cluster based cache coherence scheme. The nodes of the NoC are organized into clusters. In each cluster, there exists at least one central node which is named *HEAD*. The L2 cache and an associated directory are located in the HEAD node. The memory hierarchy is shown in Fig. 1.

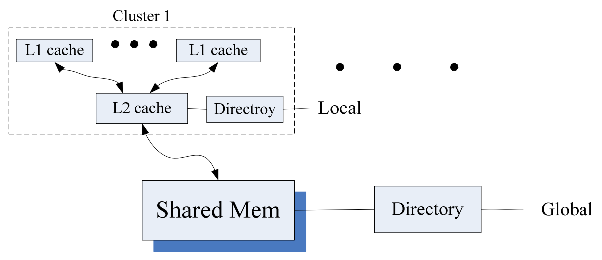

Figure 1. Memory hierarchy

The directory in L2 cache is named *local directory*. It only keeps its members' L1 cache states. The directory in the shared memory is called *global directory*. It holds the information that indicates which cluster has a valid copy of the memory block.

The cache coherence issue is divided into two levels, *inter-cluster* and *intra-cluster* coherence. The inter-cluster cache coherence is kept in the global directory in the memory. The intra-cluster coherence is maintained within the cluster. The L2 cache is the super set of its inner cluster L1 caches. To explore the spatial and temporal locality of data, the L2 cache can perform pre-fetching of the related data which will help to decrease the read miss in the cluster.

B. *Cluster structure*

A cluster is a hub based unit. The HEAD of the cluster has the average minimum distance to all its members. The minimum size of a cluster is 2x2, as shown in Fig. 2. The dark node is the HEAD of the cluster where the L2 cache and local directory are located. The distance between the HEAD and the north/west node is one hop. The distance to the corner is two hops. It is easy to construct a base 2 (2Nx2N) NoC.

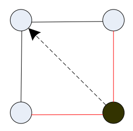

Figure 2. 2x2 cluster unit

Other two basic clusters are 2x3 and 3x3 ones, as shown in Fig. 3.

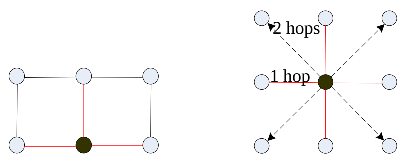

Figure 3. 2*3 and 3*3 basic cluster units

With the basic 2x2, 2x3 and 3x3 cluster units, it is easy to construct an arbitrary NxN 2D mesh NoC, e.g. 5= 3+2, 6=3+3, 7=2+2+3, 8=4x2, 9=3x3, 10=3x2+2x2, 11=3x3+2, 12=4x3, and so on. A 5x5 mesh is constructed as shown in Fig. 4.

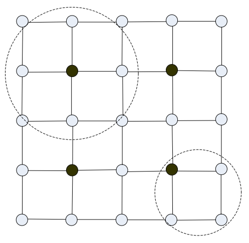

Figure 4. A 5x5 clsuter based mesh

IV. Analysis and Discussion

A. *Communication cost*

We give an example to assist explaining the gain of communication cost of the proposed scheme. Fig. 5 shows an 8x8 2D mesh NoC, which is divided into four clusters. The network uses dimension-order XY routing. Suppose a coarse 4x4 cluster unit is used and in each cluster there exists one HEAD. Fig. 5 shows a write event. There are 9 sharers, R1~R9, in the network besides the local node (L). Once the home node (H) sends out the invalidation to the sharers, it first sends them to the HEADs (3. INV), and then waits for the acknowledgement from the HEADs. Each HEAD will forward the invalidation to its members (4. FWD) and the members will send back the acknowledgements (5. ACK). Finally, the HEADs send back the response for the invalidation procedure (6. ACK). The home node will send acknowledgment to the HEAD of the requested cluster (7. ACK) and the HEAD will finally send the write permission to the local node (8. ACK).

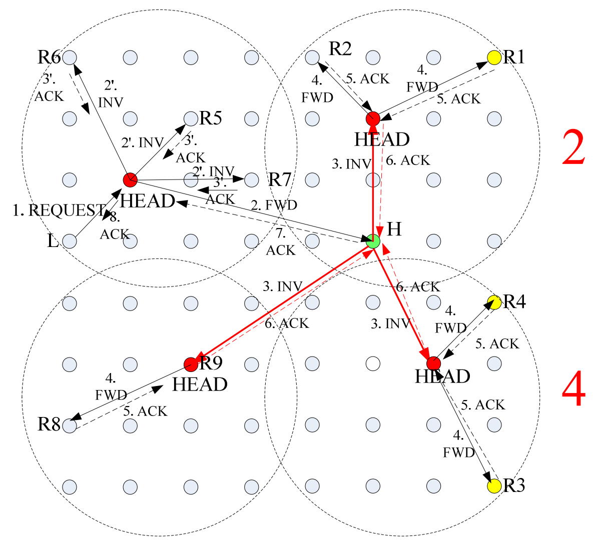

Figure 5. Clustered perspective

1) *Longest Manhattan Distance (LMD):* Since the HEAD is usually located nearer to the initiator than the longest target node, the LMD of the network will be decreased. In this example, if the flat full-map directory structure is used, the LMD is 8 hops (home node to R8). If the hierarchical cluster based cache coherence scheme is adopted, it decreases to 5 (HEAD in cluster 1 to home node).

2) *Long distance travel (LD):* In the example, we consider more than 6 hops travel to be a long distance travel. Since the HEAD acts as a hub, the long distance travel will be replaced with several shorter ones. For instance, the number of LDs is 3 in the flat directory scheme, which becomes 0 in our hierarchical scheme. Considering an access taking a round trip, it will be 6 versus 0 .

3) Total number of hops: Since the read requests or write invalidation requests can be combined to be one read or write message, it will save the amount of global communicatio. In this example, the communication cost of the flat full-map directory protocol is 104 hops. When the hierarchical cluster based scheme is adopted, the communication cost is only 74 hops. In total, 30 hops are reduced in this write event.

4) Critical path traffic: For a 2D mesh, we use the deterministic XY routing. The packets are transferred following the X axis first and Y axis second. Due to the imbalance of load, some paths may become a critical path since they may have more traffic than others. Contention and queuing on critical paths will lead to traffic jam and longer delay in the network. Reducing both the number of critial paths and traffic loads on them is important for the decreasing of the communication latency. Since several transmissions could be combined, it can potentially reduce the critical path traffic in the hierarchical cluster based scheme. In this example, for a flat full-map scheme, the INV request message sent from Head to R2, R5, R6, R7, R8 and R9 will take the same X path. This leads to be a very busy traffic on this path, especially the hop from Head to the first left node. Introducing HEAD could help to make the traffic on the critical path less loaded. There is no heavily traffic path in our hierarchical cluster based cache coherence scheme. If there are more sharers, the critical path traffic will be decreased more significantly.

B. Memory overhead

Next, the directory overhead of the hierarchical cache coherence protocol is studied. Suppose,

$$Each\ shared\ memory\ size\ =\ M_i\ byte$$

$$L1\,cache\ size\ =\ c_1\ byte\ ,\ \ L2\,cache\ size\ =\ c_2\ byte$$

$$Memory\ block\ size\ =\ L1\,cache\,line\ =\ m_1\ byte\ ,\ \ L2\,cache\,line\ =\ m_2\ byte$$

$$Number\ of\ \ L1\,caches\,(\,processors)\ =\ N_1$$

$$Number\ of\ \ L2\,caches\,(heads)\ =\ N_2$$

$$Number\ of\ \ members\ in\ each\ cluster\ =\ N_c$$

Here, we assume the members in each cluster are the same for simplicity. However, it can be different. Both the global and local directories are full mapped ones. The number of entries in the global directory equals to the size of L2 cache instead of L1 caches. And directories associated with the L2 cache store the entries of its cluster members.

In individual shared memory, the global directory overhead is as follows,

$$O_{hc-in\ mem} = \frac{M_i}{m_2}(N_2 + 1)\,bit \qquad (1)$$

Here, “hc” stands for hierarchical cluster based cache coherence protocol. The total overhead in the shared memory is,

$$O_{hc-in\ mem\ total} = (M_1 + M_2 + \cdots + M_n)\frac{N_2 + 1}{m_2}bit \qquad (2)$$

In each L2 cache, the local directory overhead is,

$$O_{hc-in\,L2} = \frac{c_2}{m_2}(N_c + 1)\,bit \qquad (3)$$

The total directory overhead in HEAD is (suppose the L2 caches are of uniform size),

$$O_{hc-in\,L2\,total} = N_2\frac{c_2}{m_2}(N_c + 1)\,bit \qquad (4)$$

So the total overhead of the cluster based hierarchical cache coherence is equal to (2) + (4),

$$O_{hc} = (M_1 + M_2 + \cdots + M_n)\frac{N_2 + 1}{m_2} + N_2\frac{c_2}{m_2}(N_c + 1)\,bit \qquad (5)$$

For a mesh NoC, if $N_2 = N_1 / 4$, $m_1 = m_2$, $c_2 = 4c_1$, then the directory overhead for the hierarchical cluster based scheme is,

$$O_{hc} = (M_1 + M_2 + \cdots + M_n)\frac{N_1 + 4}{4m_1} + \frac{N_1 c_2}{4m_1}(N_c + 1)\,bit \qquad (6)$$

And the overhead of the flat full mapped one is,

$$O_{full} = (M_1 + M_2 + \cdots + M_n)\frac{N_1 + 1}{m_1}bit \qquad (7)$$

To compare the memory overhead of the flat full map directory with our hierarchical cluster based cache coherence scheme, (7) – (6),

$$O_{full} - O_{hc} = (M_1 + M_2 + \cdots + M_n)\frac{3N_1}{4m_1} - \frac{N_1}{4m_1}c_2(N_c + 1)\,bit$$

$$\because 3(M_1 + M_2 + \cdots + M_n) >> c_2(N_c + 1)\ ,$$

$$\therefore O_{hc} << O_{full}$$

V. Related Work

A. Hierarchical directory based cache coherency schemes for multi-processor system

In the Stanford Dash Multiprocessor [9] a mixed bus-network communication scheme is used. The local system consists of processors with caches and a bus etc. In this work, a hierarchical directory cache coherence protocol is realized. The L1 caches of local system’s processors are constructed to be a cluster. The memory hierarchy is divided into: processor level, local level, home cluster level and remote cluster level. However, it does not consider clustering at the network level as we propose in the paper.

Wallach [10] constructs a hierarchical cache coherence protocol. The directory is organized in a recursive structure. It is designed to be a general solution for k-ary n-cube. However, the structure is not flexible enough to compose an arbitrary network. There does not exist a higher cache hierarchy besides L1 cache.

Acacio et al. [11] study the hierarchical directory architecture to deal with the memory overhead of directory by reducing the width of the directory entries. A binary tree with subtrees directory structure is constructed. The directory hierarchy consists of a small full-map first level directory and a compressed second-level directory. The results indicate that the performance is as good as the nonscalable full-map directory with a very significant reduction of the memory overhead.

The former hierarchical cache coherence studies are mainly focused on the memory overhead issue of directory. Few contribute on cache hierarchy to improve the system performance by decreasing communication for cache coherence at the same time. Researchs that simultaneously consider the cache hierarchy, cache coherence protocol and underlying communication structure and properties together are even fewer.

B. Cache coherency schemes for NoCs

There are much fewer works on cache coherence schemes in the NoC context. Petrot et al. [12] propose a software based solution for NoC. The memory is statically partitioned into two types of segments: local segments and shared segments. A segment is a global entity defined by a base address and a size in the address space. Based on the knowledge of applications, programmer intervenes to map shared variables in shared segments and local variables in local segments. Only local data are cacheable. The shared data could not be cached. Although there is little hardware cost for the cache coherence. The performance benefit of the caching scheme is not significant, because the shared data are the main part that assumes on chip communication and causes long travel delay. Only caching local data will improve the performance but the gain is limited.

Girao et al. [13] realize a traditional flat full map directory cache coherence scheme. The communication cost is modeled and analyzed. However, the simulation is not accurate enough and self-contained. The communication scheme and application mapping are not introduced. Also, it does not consider the NoC communication properties and the scalability of the relevant cache coherence protocol.

Eisley et al. [14] propose an approach to cache coherence for chip multiprocessors where the coherence protocol and directories are all embedded within network routers. This scheme can reduce the hop count for read and write by reading from the nearest node and sending invalidation to all the nodes directly. However, the design of the router becomes more complicated. The coupling between processing nodes and communication nodes becomes tight.

Bolotin et al. [15] introduce a priority based scheme for cache coherence control packets. The short control messages that constitute the coherency protocol could be allowed to bypass long data packets. It is helpful to extend and accelerate the operation. However, there is little discussion on the structure of the directory based cache coherence protocol itself.

The solution of cache coherence protocol which fits for NoCs is still an open problem. Because of different constraints, the solutions found for general multi-processor systems may not be suitable for NoCs. A sophisticated cache coherence protocol co-designed with the underlying communication architecture should be explored.

VI. CONCLUSION AND FUTURE WORK

In this paper, we propose a hierarchical cluster based cache coherence protocol. The memory overhead of the directory is far less than the traditional flat full-map scheme. Without limiting the number of sharers, the performance is superior to the flat full map scheme.

We will analyze communication overhead more quantitatively by means of experiments so as to study the benefits and overhead of the proposed hierarchical cluster based cache coherence protocol for NoCs.

In fact, the directory structure used here is a two level tree like directory hierarchy. The members in the clusters are the leaf nodes. The local directory is the first level, and the global directory is the second level directory. So it is a recursive structure and could be scaled further to multi-level directory tree if the NoC size grows. We will investigate at what size, the traditional protocol will be sufficient, and when the hierarchical structure should be used and also the levels and structure of the directory hierarchy. We will make a comparative study, applying the full mapped directory, limited directory, and cluster based scheme for various network sizes in the range of 16 to 512 nodes.

ACKNOWLEDGMENT

This work has been supported partially by the FP7 EU project Mosart under contract number IST-215244, the National 863 program of China under Grant No.2008AA01Z135, and the National Nature Science Foundation of China under Grant No. 60876017.

REFERENCES

[1] Mikael Millberg, Erland Nilsson, Rikard Thid, Shashi Kumar, Axel Jantsch, "The Nostrum backbone-a communication protocol stack for Networks on Chip", 17th International Conference on VLSI Design, 2004. Page(s):693 – 696

[2] Axel Jantsch, Hannu Tenhunen, editors, "Networks on Chip", Kluwer Academic Publishers, 2003

[3] John L. Hennessy, David A. Patterson, "Computer Architecture: A Quantitative Approach Fourth Edition", Morgan Kaufmann Publishers, 2007

[4] Milo Tomasevic, Veljko Milutinovic, "A survey of hardware solutions for maintenance of cache coherence in shared memory multiprocessors", Proceeding of the Twenty-Sixth Hawaii International Conference on System Science, 1993, Volume i, 5-8 Jan 1993 Pages: 863-872 vol.1

[5] John Hennessy, Mark Heinrich, Anoop Gupta, "Cache-coherent distributed shared memory: perspectives on its development and future challenges", Proceedings of the IEEE, Volume 87, Issue 3, March 1999 Pages: 418-429

[6] David Chaiken, Craig Fields, Kiyoshi Kurihara, Anant Agarwal, "Directory-based cache coherence in large-scale multiprocessors", Computer, Volume 23, Issue 6, June 1990 Pages: 49-58

[7] Milo Tomasevic, Veljko Milutinovic, "Hardware approaches coherence in shared-memory multiprocessors" part 1, IEEE Micro, Volume 14, Issue 5, October 1994 Pages: 52-59

[8] David E. Culler, Jaswinder Pal Singh, Anoop Gupta, "Parallel Computer Architecture: A Hardware/Software Approach", Morgan Kaufmann Publishers, 1999

[9] Daniel Lenoski, James Laudon, Kourosh Garachorloo, Wolf-Dietrich Webbser, Anoop Gupta, John Hennessy, Mark Horowitz, and Monica S. Lam, "The Stanford Dash Multiprocessor", Computer, Volume 87, Issue 3, March 1992 pages: 418- 429

[10] Deborah A. Wallach, "PHD: A Hierarchical Cache Coherent Protocol", master thesis, MIT, September 1992

[11] Manuel E. Acacio, Jose Gonzalez, Jose M. Garcia, Jose Duato, "A Two-Level Directory Architecture for Highly Scalable cc-NUMA Multiprocessors", IEEE Transactions on parallel and distributed systems, Volume 16, No. 1, January 2005

[12] F. Petrot, A.Greiner, P. Gomez, "On cache coherency and memory consistency issues in NoC based shared memory multiprocessor SoC architectures", 9th EUROMICRO Conference on Digital System Design: Architectures, Methods and Tools, 2006, Pages: 53-60

[13] Gustavo Girao, Bruno Cruz de Oliveira, Rodrigo Soares, Ivan Saraiva Silva, "Cache Coherency Communicaton Cost in a NoC-based MPSoC Platform", Proceedings of the 20th annual conference on Integrated circuits and systems design, September 2007, Pages: 288 - 293

[14] Noel Eisley, Li-Shiuan Peh, Li Shang, "In-Network cache coherence", The 39th Annual IEEE/ACM International Symposium on Microarchitecture 2006

[15] E. Bolotin, Z. Guz, I. Cidon, R. Ginosar, A. Kolodny, "The power of priority: NoC based distributed cache coherency", First International Symposium on Networks-on-Chip, May 2007, Pages: 117-126