Journal of Information & Communication Technology
Vol. 4, No. 1, (Spring 2010) 01-10

Snoopy and Directory Based Cache Coherence Protocols: A Critical Analysis

Samaher Al-Hothali*
Department of Computer Science and Engineering, Yanbu University College, Saudi Arabia.

Safeeullah Soomro*
Institute of Business and Technology, Biztek, Karachi, Pakistan.

Khurram Tanvir *
Ruchi Tuli *
Department of Computer Science and Engineering, Yanbu University College, Saudi Arabia.

ABSTRACT

The computational systems (multi and uni-processors) need to avoid the cache coherence problem. The problem of cache coherence is solved by today's multiprocessors by implementing a *cache coherence protocol*. The cache coherence protocol affects the performance of a distributed shared memory multiprocessor system. This paper discusses several different varieties of cache coherence protocols including with their pros and cons, the way they are organized, common protocol transitions, and some examples of systems that implement those protocols

Keywords : Cache coherence, Snoopy protocols, Directory-based protocols, Shared memory, coherence problem.

1. INTRODUCTION

Shared-memory multiprocessors have been considered for research quite considerably. Shared memory multiprocessors are famous because of the simple programming model they implement. Address space is shared among multiprocessors so that they can communicate to each other through that single address space. Same cache block in multiple caches would result in a system with caches because of sharing of data. This problem doesn't affect the read process but for writes when one processor writes to one location, this change has to be updated to all caches [1]. Cache coherence is a term that refers to ensure consistent data in all caches in case of data write.

A distributed algorithm is used to tackle the cache coherence problem known as cache coherence protocol [1]. There are different cache coherence protocols that differ from each other in the scope of places that are updated by write operation. These protocols can impact the performance of a multiprocessor system which is mostly hard to estimate. The performance of a system is directly proportional to the latency of microprocessor accesses.

* The material presented by the authors does not necessarily portray the viewpoint of the editors and the management of the Institute of Business and Technology (Biztek) or Computer Science and Engineering, Yanbu University College, Saudi Arabia.

* Samaher Al-Hothali : alhothali.samaher@gmail.com
* Khurram Tanvir : khurram.tanvir@yuc.edu.sa
* Ruchi Tuli : ruchi.tuli@yuc.edu.sa
* Safeeullah Soomro : afeeullah.soomro@biztekian.com

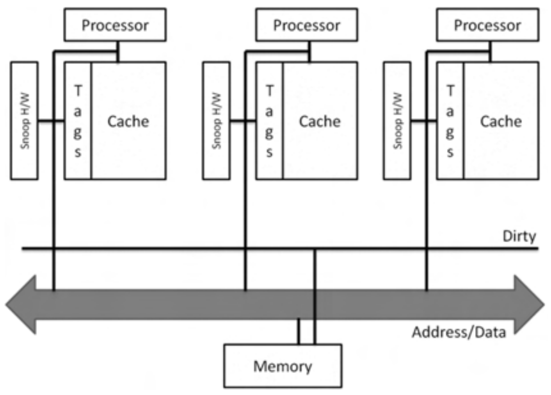
The latency of an access is mostly dependant on congestion in the system that is directly related to the amount of communication traffic. The first step in evaluating the overall performance is to analyze the processor data sharing behavior and determine its total effect on the cache coherence communication costs. This paper gives a guideline for determining the communication costs of different protocols and comparing different protocols along with estimating the effects of different system and application parameters on the overall performance of a system. Moreover improving the latency of accesses and reducing the traffic can thus reduce the cost of the system by reducing the bandwidth requirements.

This paper further discusses several different varieties of cache coherence protocols including their advantages and disadvantages, their organization, and some examples of machines that implement each protocol.

2. LITERATURE REVIEW

Communication between processors results in data coherence in snoopy protocol (Fig.1). Processor announces to all other processors by a broadcasting mechanism. The processor snoops the bus whenever shared data is present in its transaction. Whether or not an action is to be taken is decided through an algorithm (e.g. write-update or write-invalidate) [2].

Fig. 1
Snoopy Protocol



Cache coherence protocols are major factors in achieving high performance through thread-level parallelism on multi-core systems. Among them, the token coherence protocol is the most efficient cache coherence protocol in maintaining the memory consistency [3].

Cache coherence protocols are classified based on the technique by which they implement cache coherence: *Snooping and Directory based protocols*. In *Snooping based protocols*, address lines of shared bus are monitored by cache for every memory access by remote processors. The action is taken when locally saved data is changed by the transaction started by the remote processor. *Directory based protocols* have a main directory containing information on shared data across processor caches. The directory works as a look-up table for each processor to identify coherence and consistency of data which is currently being updated [4].

A directory-based protocol is a smart way of implementing cache consistency on an arbitrary interconnection network. While the resulting protocol is complex, it is indeed tractable. Moreover, the hardware needed to implement such a protocol is quite reasonable for the scale of machine in which it is expected to be used [5].

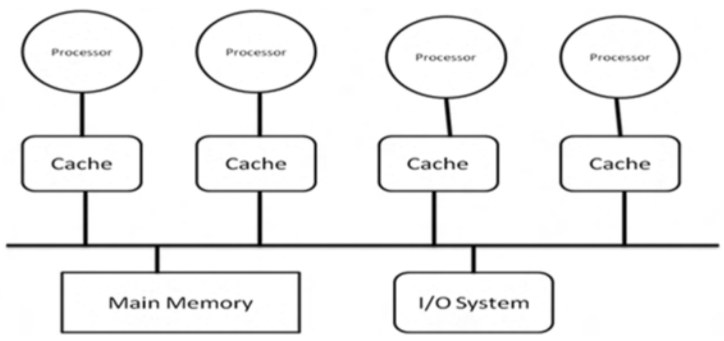
Which protocol is best is hard to answer. The question is complex since traditionally any cache coherence protocol implementation was attributed to a specific machine, and vice versa. Since each multiprocessor architecture hardwired a cache coherence protocol, comparing protocols meant comparing the performance on different machine architectures. This approach is not famous since differences in machine architecture or other implementation designs inevitably complicate the protocol comparison [6].

3. ANALYSIS

CACHE COHERENCE PROBLEM

In this section we will discuss cache coherence problem in centralized (Fig. 2), and distributed (Fig. 3) shared memory.

Fig 2
Centralized Shared Memory Architecture



This type of architecture is useful for multiprocessor (Fig. 2), It is not useful if the number of processors is equal to the bandwidth. It is used in large arrays of multiprocessors (Fig. 3). When multi-users have a common memory resource, the problem increases to keep consistent data. It is true for CPUs in a multiprocessing system. As shown in Fig. 4, if the lower user has a copy of a memory block from a previous read and the upper user changes that memory block, the bottom will end up with an invalid cache of memory without any change notification.

Fig. 3
Multiple Caches of Shared Resource

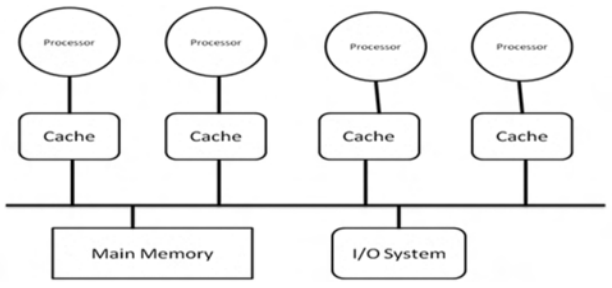


Fig. 4
Cache coherence Problem for a single Memory Location Z read and written by two processors

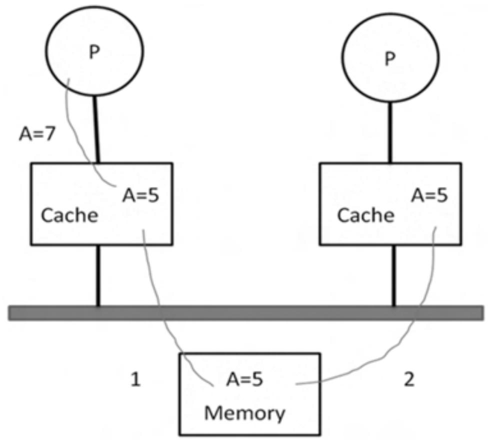


Table 1
Coherence problem for a single memory location X read and written by two processors A and B

Time	Even	Cache contents for CPU A	Cache contents for CPU B	Memory contents for location Z
0				3
1	CPU A reads Z	3		3
2	CPU B reads Z	3	3	3
3	CPU A stores 0 Into Z	0	3	0

4. CLASSES OF CACHE COHERENCE PROTOCOLS

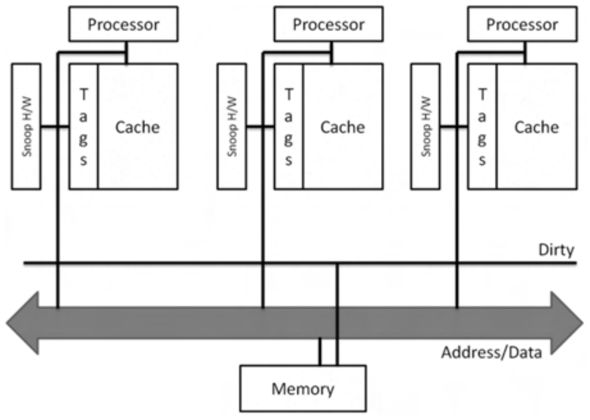
There are two main classes of cache coherence protocols, snoopy protocols and directory-based protocols.

5. SNOOPY PROTOCOLS

In bus based multiprocessor systems, appropriate coherence actions can be taken if coherence is detected. These are called snoopy protocols. The name snoopy comes from snoop, because each cache snoops bus transactions to watch memory transactions of other processors (Fig. 6). Snoopy protocols require the use of a broadcast medium in the machine and hence apply only to small-scale bus-based multiprocessors. This type of protocol is most commonly used method in commercial multiprocessor. Various snoopy protocols have been proposed.

The two primary categories of Snoopy protocols are: Write Through / Write invalidate and Write Through / Write invalidate .

Fig. 5
Snoopy Protocol

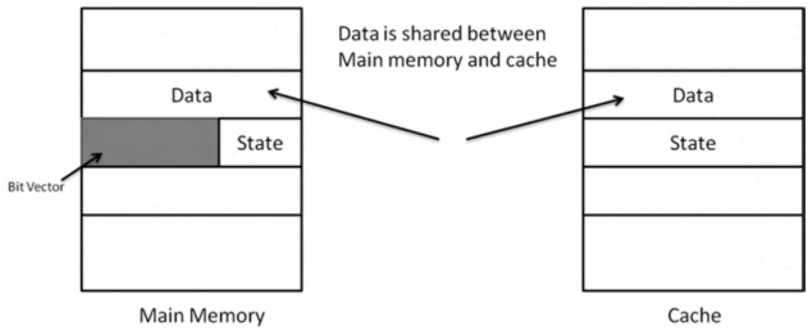


6. DIRECTORY BASED PROTOCOLS

A cache coherence protocol that does not use broadcasts will take care about the locations of all cached copies of every block of shared data and store it. These cache locations can be centralized or distributed and are called a **directories**. For each block of data there is a directory entry that contains a number of pointers. The purpose of this number is to mention the locations of block copies. Each directory entry also contains a dirty bit to specify whether a unique cache has a permission or not to write the associated block of data.

There are three primary categories of directory-based protocol: full-map directories, limited directories, and chained directories (Fig. 7).

Fig. 6
Directory Based Protocols

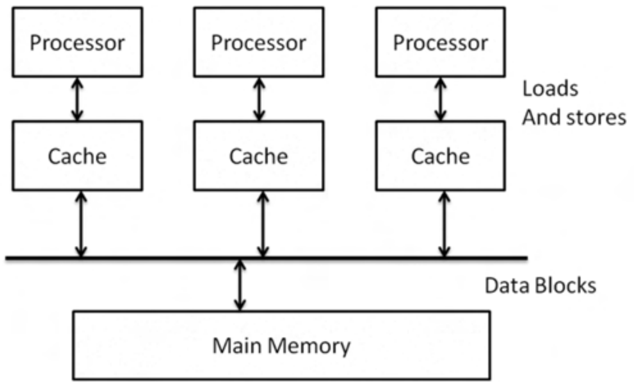


7. ORGANIZATION OF DIFFERENT CACHE COHERENCE PROTOCOLS

In this section we will discuss the performance and architecture for each protocol.

8. ORGANIZATION OF SNOOPY PROTOCOLS

Fig. 7
Snoopy Protocol Organization



There are two primary methods to maintain the coherence: write invalidate and write update. In write invalidate all other caches with a copy are invalidated. The advantage of this method is its simple implementation and the disadvantage is that it would result into cache miss. The other method is *write update* or *write broadcast* protocol, while the data item is written all cached copies of the data item will update.

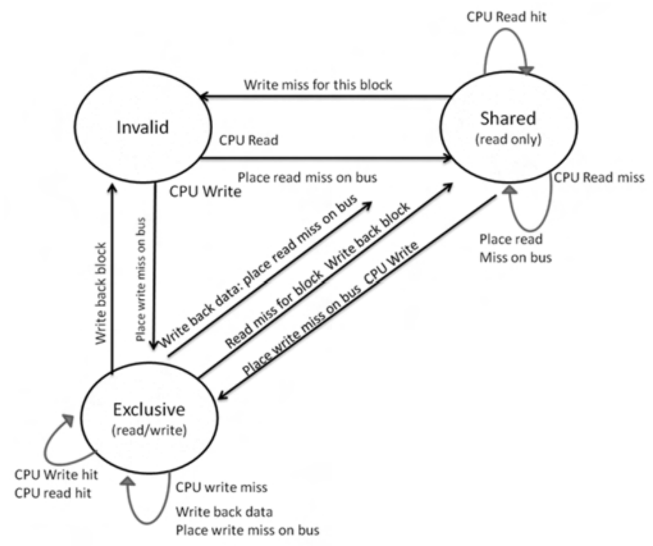
Table 2
Requests from the processor and the bus that responds to these based on their type

Request	**Source**	**State ofaddressed cache block**	**Function**
Read hit	Processor	Shard or exclusive	Read data in cache.
Read miss	Processor	Invalid	Put read miss on bus.
Read miss	Processor	Shared	Address conflict miss; put read miss on bus.
Read miss	Processor	Exclusive	Address conflict miss; write back block, then put read miss on bus.
write hit	Processor	Exclusive	Write data in cache.
write hit	Processor	Shared	Place write miss on bus.
write miss	Processor	Invalid	Place write miss on bus.
write miss	Processor	Shared	Address conflict miss; place write miss on bus.
write miss	Processor	Exclusive	Address conflict miss; write back block, then place write miss on bus.
Read miss	Bus	Shared	No action, allow memory to service read miss.
Read miss	Bus	Exclusive	Attempt to share data; place cache block on bus and change state to shared.
write miss	Bus	Shared	Attempt to write shared block; invalidate the block.
write miss	Bus	Exclusive	Attempt to write block that is exclusive elsewhere; write back the cache block and make its state invalid.

The advantages of this method causes no cache misses. The disadvantage is that it consumes considerably more bandwidth because it must broadcast all writes to shared cache lines (Each update must be global). Due to this most multiprocessor systems nowadays implement a write invalidate protocol.

In snooping protocol cache continuously snoops the bus, watching the addresses. It sees if the address on the bus is in their cache and if so, it takes respective actions depending on the request either by processor or bus. The cache coherence mechanism receives requests from the processors and the bus and responds to these, according to the type of request, if it hits or misses in the cache, and the state of the cache block specified in the request. In Fig. 8, set of state transitions for a single cache block with all requests and functions for respective request are shown in table 3.

Fig. 8
coherence state transition diagram with the state transitions induced by the local processor and the bus activities.



9. MACHINES THAT USE SNOOPY PROTOCOLS

Snoopy protocols are extensively used in commercial multiprocessor system like Pentium 4 (Fig. 9) and PowerPC (Fig. 10) .

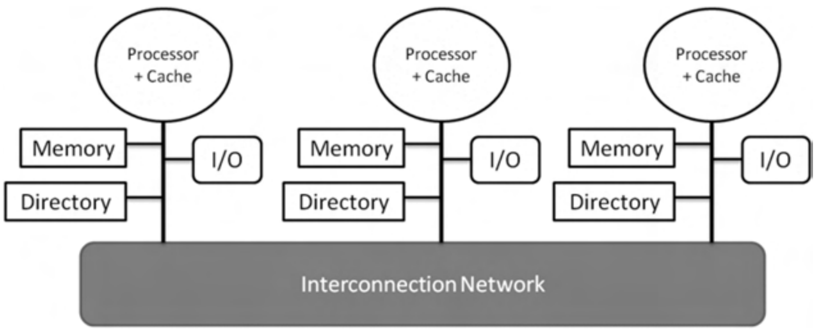
Fig. 9
Pentium 4

Fig. 10
Power PC

10. ORGANIZATION OF DIRECTORY BASED PROTOCOLS

Directory based protocol take care about the locations of all cached copies of every block of shared data and store it in a place called a **directory**. This protocol considers the system as shown below.

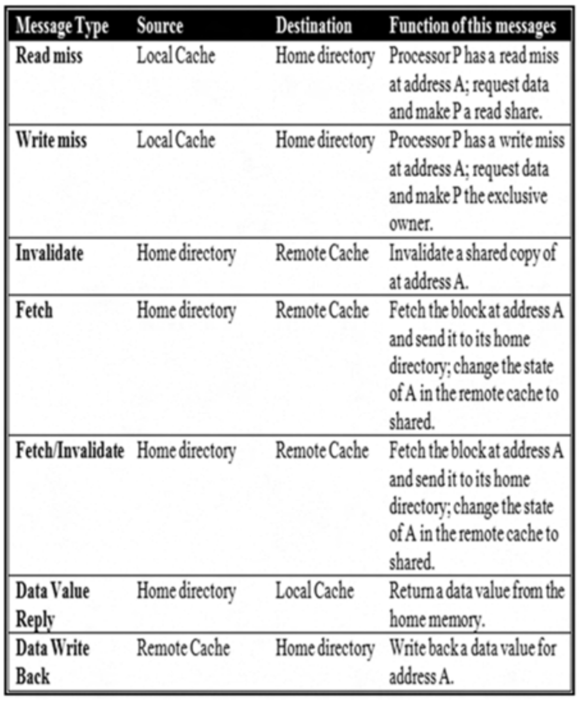
Fig. 11
Organization of Directory Based Protocol



As in Fig.11 the directory is added to each node to implement cache coherence. There are two primary operations that a directory protocol must implement: handling read miss and handling a write to a shared, clean cache block. To implement these operations directory must track the state of each cache block. These states could be the following: Shared: One or more processors have the block cached, and the value in memory is up to date. *Uncached:* No processor has a copy of the cache block. *Exclusive:* Exactly one processor has a copy of the cache block, and it has written the block, so the memory copy is out of date. The processor is called the owner of the block.

In directory- based protocol, the communication between processors and directories by sending the messages. Different messages as shown below are sent among nodes. The nodes are classified as Local node: It is the node where requests originate. Home node: It is the node where the memory location and directory entry of an address reside. Remote node: Copies exist at third node, called remote node. A remote node is the node that has a copy of a cache block, whether exclusive or shared. A remote node may be same as either the local node or the home node. The possible messages sent among nodes to maintain coherence, along with the source and destination node, the contents (where P = requesting processor number, A= requested address, and D = data contents) , and the function of the message, listed in table 3 [2].

Table 3
Possible Messages sent among nodes to maintain coherence



Message Type	Source	Destination	Function of this messages
Read miss	Local Cache	Home directory	Processor P has a read miss at address A; request data and make P a read share.
Write miss	Local Cache	Home directory	Processor P has a write miss at address A; request data and make P the exclusive owner.
Invalidate	Home directory	Remote Cache	Invalidate a shared copy of at address A.
Fetch	Home directory	Remote Cache	Fetch the block at address A and send it to its home directory; change the state of A in the remote cache to shared.
Fetch/Invalidate	Home directory	Remote Cache	Fetch the block at address A and send it to its home directory; change the state of A in the remote cache to shared.
Data Value Reply	Home directory	Local Cache	Return a data value from the home memory.
Data Write Back	Remote Cache	Home directory	Write back a data value for address A.

11. MACHINES THAT USE DIRECTORY BASED PROTOCOLS

There are two types of cache coherence protocols that are proposed: broadcast-based snoopy protocols and directory-based protocols. Each protocol has its advantage and disadvantage.

12. SNOOPY PROTOCOLS

Each cache snoops bus transactions to see memory transactions of other processors, and it needs the use of a broadcast medium in the machine. The main advantage of snoopy protocols is the low average miss latency, for cache to cache misses.

The cache coherence overhead and the speed of shared buses limit the bandwidth required to broadcast messages to all processors. Another problem of snoopy protocols is their inefficiency from the point of view of power dissipation.

13. DIRECTORY BASED PROTOCOLS

Directory based cache coherence protocols have the potential to scale shared-memory multiprocessors to a large number of processors. For this reason, we are interested to know more about the advantages and disadvantages of this class of protocols.

One important advantage of directory based protocols is that they scale much better than snoopy protocols. The second and more important advantage of directory protocols is the ability to exploit arbitrary point-to-point interconnects.

Directory based protocols have two primary disadvantages. First, the directory access and the extra interconnect traversal is on the critical path of cache to cache misses. The second disadvantage of directory based protocols; it involves the storage and manipulation of directory state. This disadvantage was more pronounced on earlier systems that used dedicated directory storage (SRAM or DRAM) that added to the overall system cost.

Table 4
Snoopy Protocols vs Directory Based Protocols

Protocols	**Advantages**	**Disadvantages**
Snoopy protocols	1. low average miss latency, especially for cache-to-cache misses.	1. The cache coherence overhead and the speed of shared buses limit the bandwidth needed to broadcast messages to all processors. 2. Not efficient from the point of view of power dissipation.
Directory-based protocols	1. Scale much better than snoopy protocols. 2. Ability to exploit arbitrary point-to-point interconnects.	1. The directory access and the extra interconnect traversal is on the critical path of cache to cache misses. 2. It involves the storage and manipulation of directory state.

14. RESULTS

Snoopy protocols are inherently faster provided enough bus bandwidth is available, because all transactions are a request/response observed by all processors. The scalability is one of the drawbacks of snoopy protocols. Each request must be sent to all nodes in a system, meaning that if the system gets larger, the size of the (logical or physical) bus and the bandwidth it provides should grow. In contrast, directory based protocols have tendency to have longer latencies, but use much less bandwidth because messages are point to point and not broadcast. So many of the big systems use this type of cache coherence.

15. FUTURE SCOPE

The concept of cooperative caching can be introduced instead of cache allotted to each processor so that caches can be used more efficiently [7].

16. CONCLUSION

Protocols for cache coherence are critical to multiprocessor systems. In general, the directory based protocol is more used for larger systems to increase their performance; while snooping protocol is used for smaller systems.

17. REFERENCES

[1] Juan Gomez-Luna Herruzo and Jose Ignacio Benavides, MESI Cache Coherence Simulator for Teaching Purposes. CLEI ELECTRONIC journal pp1-7, 2009.
[2] John L. Hennessy, David A. Patterson, David Goldberg, Computer Architecture: A quantitative Approach, fourth edition, P579.
[3] Yong J. Jang and Won W.R. Evaluation of Cache Coherence Protocols onMulti-Core Systems with Linear Workloads, ISECS International Colloquium on Computing, Communication, Control, and Management, pp 1-4, 2009.
[4] Aanjhan Ranganathan, Experimental Analysis ofSnoop Filters for MPSoC Embedded Systems, Ecole Polytechnique Federale de Lausanne, pp10.
[5] Richard Simoni, Implementing a Directory-Based Cache Consistency Protocol, DARPA, pp 1-2, 1990 .
[6] M. Heinrich, J. Hennessy, and A. Gupta, The Performance and Scalability of Distributed Shared Memory Cache Coherence Protocols, IEEE Transactions on Computers, pp 1-7, 1999.
[7] Wong Pak Shing, An effective model of cache coherence protocol with VHDL simulation. Object oriented computing pp16-18, 2008.