IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 17, Issue 3, Ver. 1 (May – Jun. 2015), PP 71-75
www.iosrjournals.org

Comparative study on Cache Coherence Protocols

Kaushik Roy[#1], Pavan Kumar S.R.[#2], Meenatchi S.[#3]
[1,2]MCA Scholar,[3]Sr. Assistant Professor
[1]kroy.compsc@gmail.com, [2]pavansr77@gmail.com, [3]meenatchi.s@vit.ac.in
[#]*SITE School, VIT University Tamilnadu, India*

Abstract: *In this new age of technology, not only the software but also the computer architecture has been evoluted to support those softwares. The main motive of evolution of architecture day by day is to make the system faster. One of the major steps in this journey of evolution is the multi-core processor architecture. In a multi-core processor system, each core has its own cache module where they are sharing the same memory unit. For that reason, one block in one cache gets invalidate when the same block is updated into any other cache and this is called cache coherence problem. To overcome this, there are lot of research works are going on and the outcome rules or techniques are called cache coherence protocols.The main objective of this paper is to collect all those research works together and represent them in an easy way, so that we could understand their techniques to overcome the particular problem. Here it is presented a comprehensive study of those cache coherence protocols with their pros and cons.*

I. Introduction

In the complex multilevel cache level architecture to bind the cache coherence protocol complexity we need to track the coherence information across all the levels but this leads implausible cost problem. To empower our future processors we need to embed more number of cores per chip multiprocessor. But during this we must face the off chip bandwidth wall, again to overcome this we can use a large on chip cache but because of plain large capacities of hardware the fast access could not be achieved. So commercially it is used three level cache architecture where two are used to maintain private data being closer to processor and the last one shared among cores. Have large number of cores might be helpful in some most of the cases but at the same time it is unachievable to rely on broadcast-based coherence protocols.

II. Comprehensive Studies

The "MESI" protocol (Illinois protocol)is one of the widely used cache coherence protocol and also supported by cache write-back policy. It has four different states-M (Modified): Means the last updated copy belongs to the current cache and the previous copy of the block belongs to memory. Before any memory transaction for that block the same copy should be written back to memory, the state changes to E (Exclusive).If there is and read request from another cache the request will be satisfy by the current cache and the state becomes S (shared).If in this situation any one of the block changes the copy the others goes to I (Invalid).For two cache the allowable state change will be following:

Figure 1: MESI state interchange

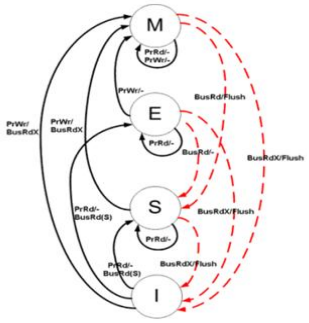

Figure 2: State Diagram of MESI

In the paper "A New Kind of Hybrid Cache Coherence Protocol for Multiprocessor with D-Cache" a new protocol is proposed called "MECSIF" based on "MESI" by some group of scientists.[22]The main benefits of this protocol are: reduces Level1 cache miss ratio ,overcomes the unnecessary broadcasting of snoopy a protocol and also enhances the data access rate by processor compared to "MESI".A small size directory-DCache is used to overcome the broadcasting problem of snoopy protocol and also to reduce inter-linked bus traffic, an additional C(coherence) bus deports directory requests. Every the first level cache is connected with DCache and bus are connected with CBus.

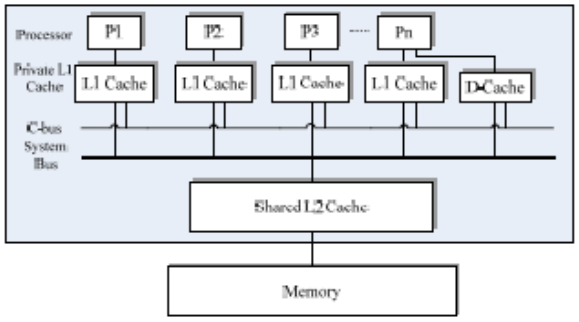

Figure 3: System architecture of DCache

D-Cache directory contains following:

Address: address of data block
Condition: state of data block
Processor-number: processor code of data block

The allowable operations are as followings-

i) Cache controller using C-bus sendsdirectory request to the D-Cache.
ii) When there occurs a directory miss, failure message is sent back, otherwise, send message requesting data to system bus.
iii) Remote cache replies with a message on behalf of data copy block for any request then system bus receives the returned request.
iv) When the local cache got message it concentrates on the copy of data block.

The states of cache data block copy are more or less same as "MESI" but it is extended to three more additional states.

"PC (Primary Clean)": There exits two copies of a block .One of the in PC state means; it is the master copy of two."SC (Slave Clean)": There exits two copies of the block and this one is the slave copy.

"F (Forwarding)": The copy of data block has not yet been modified once but one cache has it at least. Processor responds to only the data requests coming from remote cache, which contains data block copy of same state. Other states are as per "MESI" protocol. In the figure 4 the "MECSIF" protocol has been explained with seven states where direction of arrows defines the changing of state:

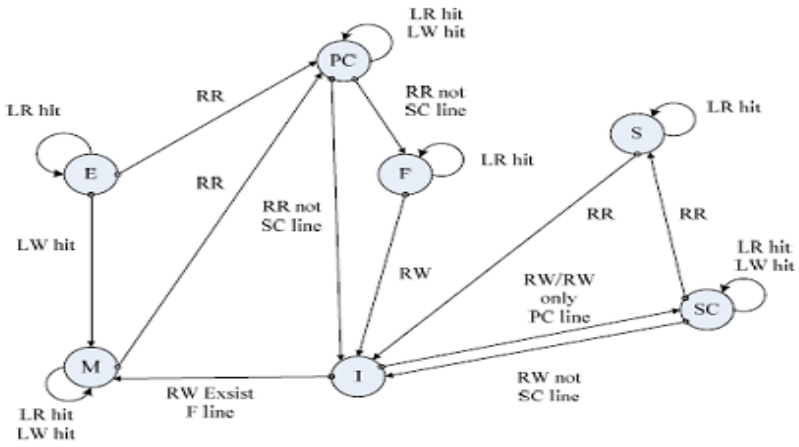

Figure 4: State diagram of MECSIF

When there is "LR (Local Read)", it means there needs a read on local cache. The "LW (Local Write)" means, there is a write to local cache. The "RR (Remote Read)" means a read is required on remote cache and the "RW (Remote Write)" means; we need to write to remote cache.

A team of scientists of University of Cantabaria ("Lucia G. Menezo, Valentin Puente, Jose Angel Gregorio") [11] proposed a new protocol named as "MOSAIC" based on token coherence correctness. The sates of this protocols are M: Modified, O: Owened, S: shared, A: Allocated, I: Invalidate, C: Costracting. The simplicity of Token Coherence protocol and the power efficiency of traditional Directory based coherence is innated in this proposed protocol. Simply Token coherence protocol means assigning a number of tokens per cache blocks and requires all of them to write the block or at least one token to read that. In this protocol a private data block will not extrude just because there is not enough capacity in directory to hold its coherence information. If a block is extruded from directory the private copy never gets invalidate. So it means after a miss in directory there might be a valid copy of that block in any private levels in off chip memory. So after subsequent miss to ensure coherence correctness an on chip reconstruction of directory entry is initiated. This whole process finishes when all the associated coherence information of that block has been collected. To perform this process there needs the token counting. The benefits of this "MOSAIC" are: decreases the number of misses in private cache which will be reduced not only average access time but also bandwidth consumption.

75% of the total memory request is done for the private data of any application program where as cache coherence protocol is unable to identify between a shared and private block. There is no requirement of resolving cache conflicts for the private data blocks. Without a conflict free private data access there may arise memory latency, directory needs to hold the addition states for private data which consumes a lot of space and message flow for each of private data block access reduces system efficiency and consumes a huge amount of power. Scientists "Wang Shaogang, Xu Weixia, Pang Zhengbin, Wu Dan, Dai Yi and Lu Pingjing" introduces a new cache coherence protocol called "PMESI(Private-Modified-Exclusive-Shared-Invalid)"[8] . So this "PMESI" bypasses the memory accesses for private data blocks by reducing access latency and allows to change the state of a blockbetween shared and private state. In traditional "MESI" protocol memory accesses the DRAM twice –first access to find out the exact location of the newest copy and second access to fetch the memory data. It is easy to understand that there will be no conflicting request for a private data, the DRAM accesses can be reduced for those cases. So here is an additional state called private (P) which means the requesting cache owns it exclusively. In OS kernel level, the page data structure of physical memory introduces a counter subfield that helps us to check how many virtual thread space has been mapped onto it; in shorts whether it is shared or private. If the counter value is more than one that means that is a shared block otherwise it's a private one. In this case each page table entry has a P-flag which is cleared when shared counter is greater than one and there is an additional B-flag. It means for the temporal state, raised conflicts need to be handled by accessing directory by private cache transaction, when B-flag is set. The authorization of changing this two flag field is belongs to OS according to the counter value. Memory request type is satisfied by virtual-to-physical address translation and in memory request path it reduces the delay. Simulation result shows for this technique that 54% memory references coherently may be improved and program execution time can be trimmed by 9%.

Problem also can be occurred when some nodes are connected in a network and each of them are multi-processor system. So in that cases Token passing is a good way to communicate with each other where processors passes tokens around system and according to the protocol depending on the number of token determines whether a given block is legal to access or not. "ArunRaghavan, ColinBlundell and Milo M. K. Martin" proposed in their paper a new protocol named as PATCH [14] which supports direct requests and also

without a non-scalable interconnection, it can predict the destination sets through the combinationalstrategy of "token passing" and "standard directory protocol". As it is very complex model to implement practically, commercial manufacturers do not use.

Whereas traditional 3-HOP protocol [15][16][17][18][19] , known as Indirection protocols are widely used because of its simple implementation architecture for homemade processor system.In the figure 5(a) a "classic 3-HOP protocol" has been shown .The unavailability of newest data in home node initiates a request, that will be forwarded to owner from local by the home itself and that request is serviced by owner. Because of out of order execution the cache coherence problem is solved in serial processing. In "traditional directory based protocols" conflict is determined by home node. Now to resolve this conflict different group of scientists proposed different methods.

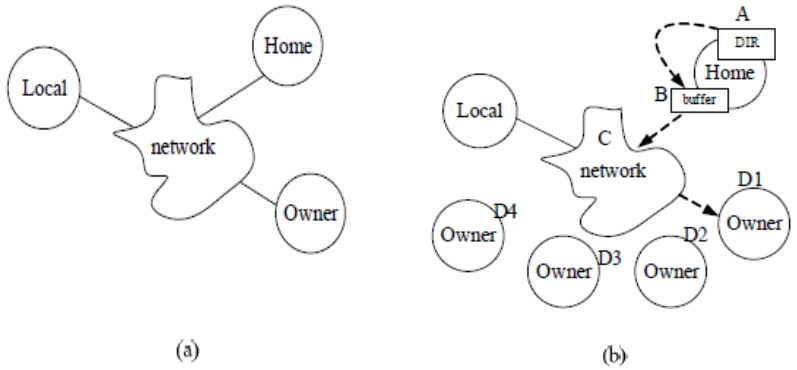

Figure 5 a) Basic-directory based protocol **b)** Different conflict Solution positions

"J. Laudon and D. Lenoski" proposed "SGI origin" model [18]. Here homenode is used to resolve the raised conflicts(shown in figure 5(b)).Whenever there is an request for an any block arrives in home that block state is assigned as "busy" and all subsequent requests are queued until the first request is deactivated.QP1 replies on "Transaction-in-Transit Table"(TTT) buffer to solve the conflict(figure 5(b) position B).

Whether as "Gharachorlooy, Madhu Sharma, Simon Steely, and Stephen Van Doren" proposed GS320 model[19] where conflicts are solved using global switch position(figure5 (b) position C). When there is a race condition, the early request race mechanism delays forwarded requests that reach their targets early in the global switch. May be the positions of solving conflicts are different but still they are central system based.

"Scientists Luiz AndrCBarroso, KouroshGharachorloo, Robert McNamarat, AndreasNowatzyk, ShazQadeert, Barton Sano, Scott Smith, Robert Stets, and Ben Verghese" proposed to solve those conflicts at the end of the system in their PIRANHA model [20]. So the solution points change from A, B, C to all the D positions that means responsibility goes to the owners. So it is easy to understand that several cache lines can be segmented to several owners to find a solution for their respective conflicts. Piranha model also introduces the optimization of "clean-exclusive", forwarding the replies from respective remote owners, exclusive replies and neglecting the use of negative acknowledgment (NAK) messages to directory-based protocols. But it should be also considered that the deadlock solution in this model takes sufficient buffering in the network whereas GS320 limits this one by global switch.

"Huang Yongqin, Yuan Aidong, Li Jun, Hu Xiangdong" proposed another method called "Novel-Directory Based Non-Busy, on-Blocking cache coherence (NB2CC)"[21] .It is the outcome of the combination of some traditional protocols, like relaxed memory model,avoiding protocol deadlock and basic process of request reces which leads this protocol to high efficiency and concurrency at low cost. Many techniques, such as Avoiding Deadlock, P2P Order in VC1, No Negative ACK used in GS320 and Piranha, are implemented in a simple but novel way in NB2CC in a more effective and competitive way. In this model serial processing is done in two steps: detecting a conflict first and then get a solution of that. Conflict detection is done at homenode end whereas solution done at distributed owners end. The benefits of this model are unnecessary ordering requirements can be eliminated to achieve more concurrency and pipeline performance at the time of conflicts and secondly the overhead can be much more reduced.

III. Conclusion

Cache coherence is a great matter of consideration for a multi core or multi-processor system. The world of technology is enhancing day by day. Most of the time we use to consider only the software development field as the signature of advancement, but at the same time we need to work at hardware level also to provide proper support of those software itself. Nowadays most of the software use to execute in multi-threadedenvironment, so without a multi core architecture this facility cannot be provided. For a single processor, cache coherence can be solved most of time very easily and also in less amount of time. But for multi core or processor there are lot of conditions need to be considered, otherwise it may lead to an unreliable system

architecture where most of the CPU cycles will be wasted to resolve the coherency instead of executing the pending tasks. So the main goal of all type of protocols is to resolve the coherence problem using minimum no of CPU cycles. So every day when the field of software is growing, at the same time the field of Hardware is growing parallel. Every year scientists are proposing about hundreds of cache coherence protocols and still this procedure continues. We here try to cover about from the beginning protocol to the latest one as per their requirement in different situations with different conditions. Still thousands are left there, but due to the space constrain we are able to describe only some of them.

References

[1]. Fradik Dahlgren and Per Stenstrom,"Using write caches to Improve Performance of cache coherence Protocols in Shared Memory Multiprocessors", Journal of Parallel and distributed Computing, 1995.

[2]. Anant Agarwal, RichardSimoni, John Hennessy and Mark Horoiwitz, "An Evolution of Directive schemes of for cache coherence", IEEE 1998.

[3]. Milo M.K. Martin, Mark D. Hill, David A. Wood,"Token Coherence: A New Framework for shared-memory Multiprocessors", IEEE 2003

[4]. Liqun Cheng, John B. Carter, DonglaiDai,**"**An Adaptive Cache Coherence Protocol Optimized for Producer-Consumer Sharing", 2007

[5]. Muthukumar.S and Dhinakaran.K, "Hybrid Cache Coherence Protocol for Multi-Core Processor Architecture", International Journal of Computer Applications in May 2013

[6]. Elver, M.,Nagarajan, V.,"TSO-CC: Consistency directed cache coherence for TSO", IEEE 2014

[7]. Xiaoyue Pan and Jonsson, B."Modeling cache coherence misses on multicores ", IEEE 2014

[8]. Wang Shaogang, Xu Weixia, Pang Zhengbin, Wu Dan, Dai Yi, Lu Pingjing , "Optimizing Private Memory Performance By Dynamically Deactivating Cache Coherence", 2012 IEEE 14th International Conference on High Performance Computing and Communications

[9]. B. A. Cuesta, A. Ros, M. E. G´omez, A. Robles, and J. F. Du-ato, "Increasing the effectiveness of directory caches by de-activating coherence for private memory blocks" , SIGARCH Comput. Archit. News.

[10]. D. Kim, J. Ahn, J. Kim, and J. Huh,"Subspace snooping: filtering snoops with operating system support", ser. PACT '10, 2010.

[11]. Lucía G. Menezo Valentin Puente Jose Angel Gregorio,"The Case for a Scalable Coherence Protocol for Complex On-Chip Cache Hierarchies in Many-Core Systems", 2013 IEEE

[12]. M. M. K. Martin, M. D. Hill, and D. J. Sorin, "Why on-chip cache coherence is here to stay", Communications of the ACM, Jul. 2012.

[13]. N. Kurd, J. Douglas, P. Mosalikanti, and R. Kumar,"Next generation Intel® micro-architecture (Nehalem) clocking architecture" , IEEE, 2008

[14]. ArunRaghavan, Colin Blundell and Milo M. K. Martin,"Token Tenure: PATCHing Token Counting Using Directory-Based Cache Coherence", in the Proceedings of the 41st International Symposium on Microarchitecture (MICRO-31), November 2008.

[15]. J. Kuskin et al, "The Stanford FLASH Multiprocessor", 21st Annual International Symposium on Computer Architecture, 1994.

[16]. D. Lenoski, J. Laudon, K. Gharachorloo, A. Gupta, and J. L.Hennessy ,"The Directory-Based Cache Coherence Protocol for the DASH Multiprocessor",17th Annual International Symposium on Computer Architecture May 1990

[17]. Yuan Aidong, Dong Jianping, "Analysis of directory-based cache coherence protocol", Computer Engineering, 2004

[18]. J. Laudon and D. Lenoski, "The SGI Origin: A ccNUMA Highly Scalable Server", In 24th Annual lnternational symposium on Computer Architecture, June 1997

[19]. KouroshGharachorlooy, Madhu Sharma, Simon Steely and Stephen Van Doren,"Architecture and Design of AlphaServer GS320", ASPLOS, 2000.

[20]. Luiz AndrCBarroso, KouroshGharachorloo, Robert McNamarat,AndreasNowatzyk, ShazQadeert,Barton Sano, Scott Smith, Robert Stets, and Ben Verghese, "Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing", Proceedings of the 27th Annual International Symposium on Computer Architecture, June 2000.

[21]. Huang Yongqin, Yuan Aidong, Li Jun, Hu Xiangdong, "A Novel Directory-Based Non-Busy, Non-Blocking Cache Coherence", 2009 International Forum on Computer Science-Technology and Applications,IEEE

[22]. Jingmei Li, Haiyang Guan , Yanxia Wu, Pengfei Yang, Jianpei Zhang , Jing Li, Nan Ding, Chaoguang Men, Chaoyu Wang **,"**A New Kind of Hybrid Cache Coherence Protocol for Multiprocessor with D-Cache", 2011 International Conference on Future Computer Science and Education,IEEE 2011.