An Experimental Analysis of Multicore Cache Organization with Ring Network

Ajay Mathur*, Dr. Rajeev Mathur#
*Dept. of Computer Science & Engineering Government Polytechnic College
Jodhpur, Rajasthan, India
#Dept. of Computer Science & Engineering Lachoo Memorial College of Science & Technology
Jodhpur, Rajasthan, India
[1]ajay_mathur09@)yahoo.com
[2]rajeev.mathur69@gmail.com

Abstract—**Having multiple cores on a single chip gives rise to some problems and challenges. Memory/cache organization is the main challenge, since all designs discussed till date have distributed L1 and in some cases L2 caches which must be coordinated. And finally, using a multicore processor to its full potential is another issue. If programmers don't write applications that take advantage of multiple cores there is no gain, and in some cases there is a loss of performance. Application need to be written so that different parts can be run concurrently (without any ties to another part of the application that is being run simultaneously). This paper analyses the overall architecture of multicore using ring network. The ring structure has been simulated using the simulator multi2sim and performance in terms of L2 cache hit ratio and IPC evaluated. The simulation has been done for the core 2,4,8 and 16.**

Keywords—**Multicore, Cache, Ring, Multi2Sim.**

I. INTRODUCTION

The evolution of parallel machines has changed dramatically. For the first time, major chip manufacturer companies whose primary business is fabricating and selling microprocessors have turned to offering parallel machines, or single chip multicore microprocessors as they have been styled.

There are a number of reasons behind this, but the leading one is to continue the raw performance growth that customers have come to expect from Moore's law scaling without being overwhelmed by the growth in power consumption. As single core designs were pushed to ever higher clock speeds, the power required grew at a faster rate than the frequency. This power problem was exacerbated by designs that attempted to dynamically extract extra performance from the instruction stream, as we will note later. This led to designs that were complex, unmanageable, and power hungry.

II. MULTI2SIM SIMULATOR

Multi2Sim is a simulation framework for CPU-GPU heterogeneous computing written in C. It includes models for superscalar, multithreaded, and multicore CPUs, as well as GPU architectures.

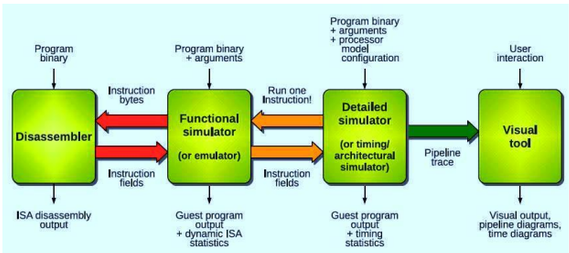

Figure 1.1

The development of the model for a new microprocessor architecture on Multi2Sim consists of four phases, represented in Figure 1.1 in order from left to right. The development phases involve the design and implementation of four independent software modules: a disassembler, a functional simulator, a detailed simulator, and a visual tool.

These four software modules communicate with each other with clearly defined interfaces, but can also work independently. Each software component though requires previous (left) design modules to work as a stand-alone tool. In this manner, the detailed simulator can operate by interacting with the functional simulator and disassembler, and the functional simulator can likewise be used in isolation together with the disassembler. However, it is not possible to use the visual tool without the remaining components.

III. MULTICORE RING ORGANIZATION

An example of ring organization is represented in Figure 3.1, using a 4-core processor. The cores have private L1 data caches, and a common L1 instruction cache is shared every two cores. The [Entry <name>] sections in the memory hierarchy configuration file are responsible for doing the association between CPU cores and data or instruction caches, by assigning values to the InstructionModule and DataModule variables. The network declared between the L1 and L2 is an internal network with

default topology, while the network between the L2 caches and the main memory modules is an external network with custom topology.

In this example, the two L2 caches serve independent sets of higher-level (L1) caches, so each L2 can serve the entire address space. Thus, the AddressRange variable is not given for the L2 modules. Main memory is configured using a banked organization with four banks, using the alternative syntax for the value of AddressRange in the main memory modules.

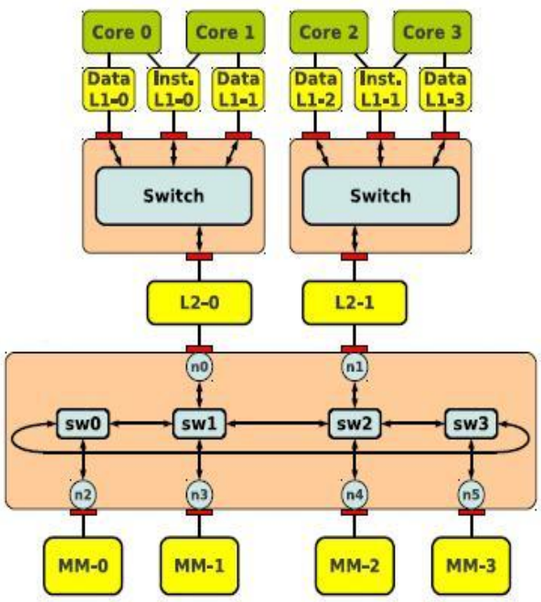

Figure 3.1 Multicore Cache Organization using Ring Network

The ring network net0 is defined to connect L2 cache modules with main memory modules. Two end nodes required for the L2 cache modules, and four additional end nodes are associated with the four main memory modules. Four switches are connected forming a ring, where each of them is connected to one main memory module, and only two of them are connected to L2 caches.

A ring topology contains a cycle of network links, which can cause deadlocks when routing packets between input and output buffers or intermediate switches and end nodes.

TABLE I. RING NETWORK

Core	L2-0	L2-1	Dispatch IPC	Issue IPC	Commit IPC
2	0.6126	NA	3.854	3.313	2.907
4	0.4865	0.4637	6.990	6.020	5.302
8	0.6721	0	6.908	5.951	5.240
16	0.6648	0	6.876	5.930	5.228

IV. SIMULATION RESULTS

As shown in the Figure 4.1, L2-0 cache hit ratio decreases for the change of core from 2 to 4, and further increasing the number of core increases. Also increasing again the core from 8 to 16.

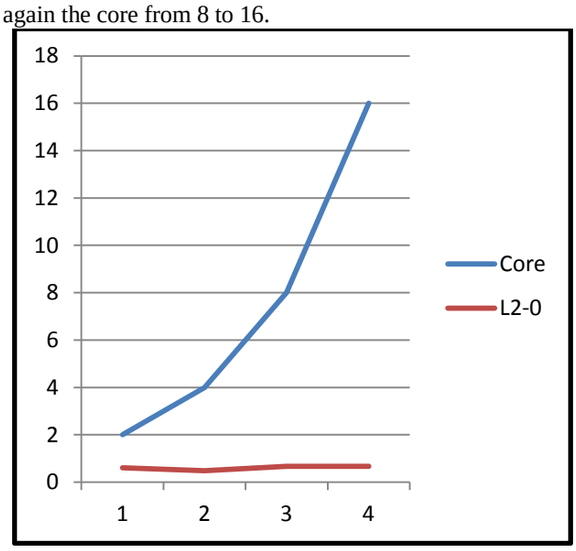

Figure 4.1 No. of Core and L2-0 Cache

Figure 4.2 sows the effect of number of core on the L2-1 cache hit ratio. For the core 2 the L2-1 cache is not applicable. For core 4 to 8 change the L2-1 cache hit ration increases and after then it becomes 0.

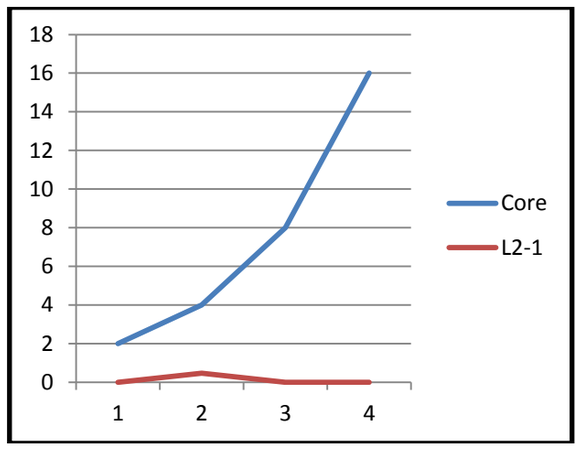

Figure 4.2 No. of Core and L2-1 Cache

The effect on the dispatch IPC of number of core is shown in figure 4.3. As per the graph the Dispatch IPC increases proportionately with number of core upto core 2 and after that there is no further improvement in Dispatch IPC.

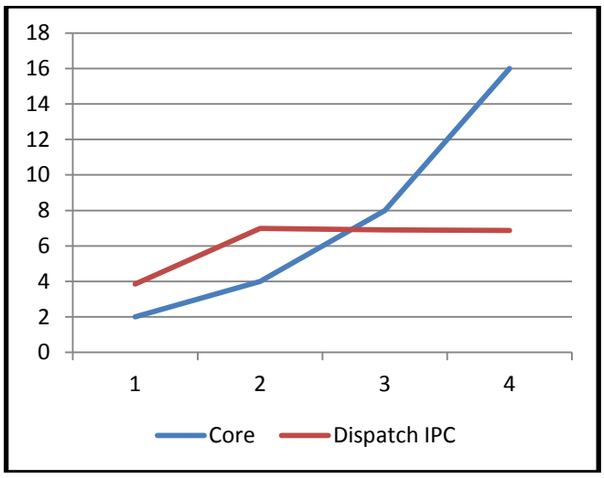

Figure 4.3 No. of Core and Dispatch IPC

The ralation between number of core and issue IPC is shown in figure 4.4. The phenomemen of Dispatch IPC is repeated again here. The Issue IPC increses linerly with the number of core upto core 2 and after that there is no further improvement in Issue IPC.

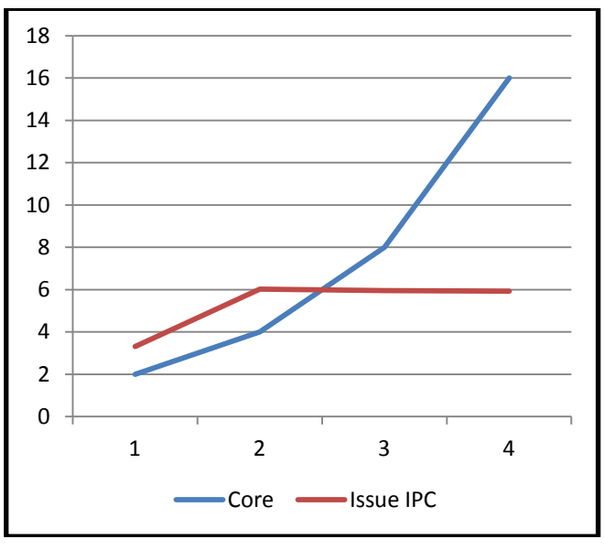

Figure 4.4 No. of Core and Issue IPC

Finally the Commit IPC trend is shown in figure 4.5. As shown in figure Commit IPC is highest at the core 2.

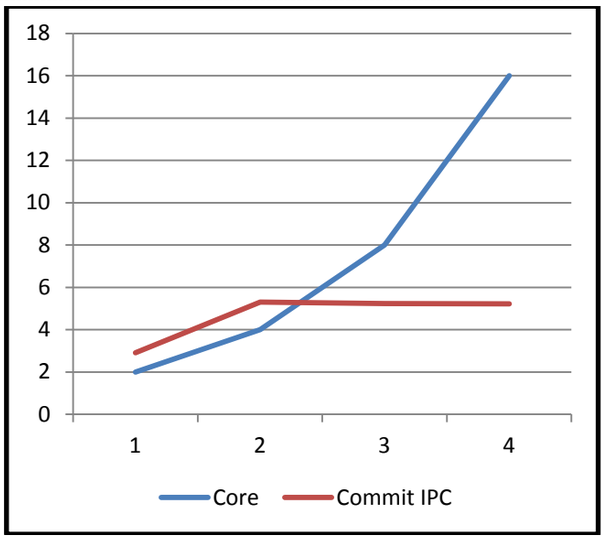

Figure 4.5 No. of Core and Commit IPC

V. CONCLUSION

As per the simulation result it can be concluded that the core 2 CPU provide the enough IPC and Cache hit for ring organization of multicore CPU and there is no improvement in IPC and cache hit for core 4,8 and 16. Hence the ring organization performs best for the number of core 2.

REFERENCES

[1] Group-caching for NoC based multicore cachecoherent systems Wang Zuo; Shi Feng; Zuo Qi; Ji Weixing; Li Jiaxin; Deng Ning; Xue Licheng; Tan Yuan; Qiao Baojun Design, Automation & Test in Europe Conference & Exhibition, 2009. DATE '09. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Investigating the impact of multimedia applications on multicore cache memory subsystems Rani, M.S.; Mridha, M.F.; Asaduzzaman, A.S. Electrical and Computer Engineering (ICECE), 2010 International Conference

[3] Gaining insights into multicore cache partitioning: Bridging the gap between simulation and real systems Jiang Lin; Qingda Lu; Xiaoning Ding; Zhao Zhang; Xiaodong Zhang; Sadayappan, P.

High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium

[4] Multicore Cache Simulations Using Heterogeneous Computing on General Purpose and Graphics ProcessorsKeramidas, G.; Strikos, N.; Kaxiras, S. Digital System Design (DSD), 2011 14th Euromicro Conference

[5] Evaluating the Cache Architecture of Multicore ProcessorsJie Tao; Kunze, M.; Karl, W. Parallel, Distributed and Network-Based Processing, 2008. PDP 2008. 16th Euromicro Conference.

AUTHORS

Ajay Mathur is working as the Lecturer (SG) in Govt. Polytechnic College, Jodhpur (Rajasthan) since 1994. A renowned Computer Engineering Professor, in the field of software as well as hardware, born in Kishangarh (Ajmer) in Rajasthan (India) in 1970.Mathur's family is living in Jodhpur (Rajasthan) since 1994. After obtaining his B.E. in Computer Science & Engineering from Amravati University (Maharashtra) in 1991, he joined Rajasthan Housing board, Kota as System analyst cum programmer. In 1995, he was selected as Lecturer (Computer Engg.) in Govt. Polytechnic College, Jodhpur by Rajasthan Public Service Commission. In 2006, he was promoted as Lecturer (SG) equivalent to Associate Professor. He has already present 10 papers in international seminars and has written 11 books related to computer field with Vardhan publication, Jaipur. He has completed M.Tech. in Computer Engg. from Jodhpur National University in 2011 and pursuing PhD from same University since 2011.

Prof.(Dr.) Rajeev Mathur has secured his BE, M.Tech. in Computer Science and Engineering and Ph.D. from Jai Narayan Vyas University, Jodhpur, India.. He joined as a faculty member in Lachoo memorial college of Science and technology, Jodhpur in 1991and currently is working as Professor and Director in the Department of Computer Science of the same college. His research interests include Content-Based Image and Video Retrieval, Multimedia and Image Databases, Computer Vision and Pattern Recognition, machine learning.