Architecting Fabricated Implant in Interconnect for multi-core processor

Ram Prasad Mohanty, Ashok Kumar Turuk, and Bibhudatta Sahoo

National Institute of Technology, Computer Science Department,
Rourkela, Odisha, India
ramprasadmohanty@gmail.com
{akturuk,bdsahu}@nitrkl.com
http://www.nitrkl.ac.in

Abstract. In this paper, we analyze the effect of interconnect on the multi-core processors and have proposed a novel highly scalable on-chip interconnection mechanism for multi-core processors. As the number of cores increases, traditional on-chip interconnect like bus and crossbar proves to be low in efficiency as well as suffer from poor scalability. In order to get rid of the scalability and efficiency issues in these traditional interconnects, ring based design has been proposed. But with the steady growth in number of cores have rendered the ring interconnect too infeasible. Thus, novel interconnect designs are proposed for the future multi-core processors for enhancement in the scalability. In this paper, we analyze and compare the interconnect of two existing multi-core processors named Multi-core Processor with Internal Network(MPIN) and Mult-core processor with Ring Network(MPRN). We have also proposed a highly scalable and efficient interconnect named as *fabricated Implant in Interconnect* for multi-core processors. The placement of cores and cache in a network is proved to be highly crucial for system performance. The benchmark results are presented by using a full system simulator. Results show that, by using the proposed on-chip interconnect, compared with the MPIN and MPRN, the execution time are significantly reduced for three applications.

Keywords: Multi-core Processor, Performance analysis, Interconnect, Cache Dependency

1 Introduction

Multi-core processors with greater number of cores and complex on-chip interconnect are recent trend since the past few years. The constraints with respect to power consumption, chip clock frequency and heat dissipation have made the chip designers to evolve from improvement in the single-core processors to integration of multiple cores on a single chip. A recent trend of enhancement in performance is to enhance the number of cores per chip. [1] This enhancement in the number of cores lead to the proposal of concept of network-on-chip

(NoC). Before this concept was proposed, system-on-chips (SoCs) took the aid of complex traditional interconnects like bus structures for connection between the cores to memory and I/O. The traditional bus structures were improved to be used as interconnect in the Multi-core processors. But with enhancement in the number of cores these bus designs were not able to sustain the interconnect scaling as well as complexity. Eventually NoC was used as a solution to the scalability issues [2] [3]. Multi-threading/multi-core technology increases performance, but doing so requires more power than single threading/core computers. Power was not an issue at the beginning of computer era. However, power has become a critical design issue in computer systems [4]. Multi-threaded and multi-core systems also requires more space (area) than a single threaded or single core system [5]. Cores in multi-core system have hardware resources for themselves and use them each for processing [6]. In this paper, we analyze the effect of interconnect on the multi-core processors and have proposed a novel highly scalable on-chip interconnection mechanism for multi-core processors.

The paper has been organized as follows: section II gives a brief description on the existing architectures and related works done in Multi-core Processor Technology, section III describes the proposed work, section IV provides a detailed description of the simulation results and section V gives a concluding remark and the future direction of our work.

2 Architecture and Background

Various work in current literature has explored the multi-core architecture utilizing various performance metrics and application domain. D.M. and Ranganathan [7] have analyzed a single-ISA heterogeneous multi-core architecture for multi-threaded workload performance. The objective was to analyze the performance of multi-core architectures for multi-threaded workloads. This section details the benefits of variation in the interconnection network in the multi-core architecture with multi-threaded workloads.

Various works has analyzed the performance in both single core and multi-core architectures. Julian et al. [8] determined the relationship between performance and memory system in single core as well as multi-core architecture. They utilized multiple performance parameters like cache size, core complexity. The author have discussed the effect of variation in cache size and core complexity across the single core and multi-core architecture.

Multi-core architecture with multiple types of on-chip interconnect network [9] are the recent trend of multi-core architecture. This type of architecture have different type of interconnect networks with different core complexity and cache configuration.

3 Proposed Work

In this paper, we propose an interconnect architecture for multi-core processors. We name the proposed architecture as Multi-core Processor with Fabricated

Implant in Interconnect (MPFII). A block diagram of the proposed interconnect is shown in Figure 1.It can be configured with different numbers of cores the Figure shows an example with 8 cores. Each core is a out of order super-scalar SMT core capable of running more than one thread at once.

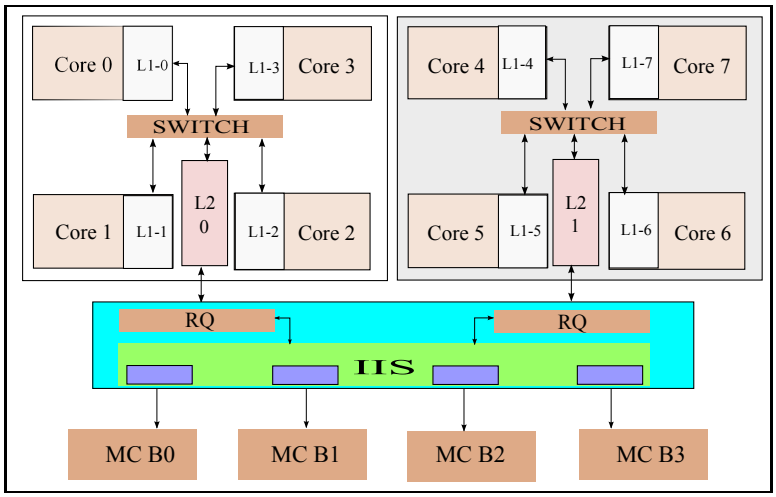

Fig. 1. Multi-core Processor with Fabricated Implant in Interconnect

Each core has a private L1 cache and is shared between the multiple threads in that core. The L1 cache has the following configuration: Block size is 256 bytes for both data and instruction cache. Associativity: L1 caches are 2-way associative, so cache lines from the L2 cache can be mapped to any L1 cache line. Replacement Algorithm : LRU replacement. Ports: Number of ports for each L1 cache is 2. Each L2 cache shares the full address range thus isolating any coherent issues to a local region with which the L2 cache is associated with.

The MPFII uses a hybrid mode of connection. It uses a crossbar switch to connect every four L1 data and instruction cache to its corresponding L2 cache. L1 cache communicate with the L2 cache through the switch. No two L1 cache can communicate with each other. Each L2 cache has the following configuration: Block Size is 256 bytes. Associativity: This cache is 4-way set associative. Replacement Algorithm : LRU Latency: 20 cycles for a L2 cache miss. Ports: Number of ports for each L2 cache is 4. Each core communicates with the corresponding L2 cache through a non-blocking crossbar which can let simultaneous message passing as long as each message is headed for a unique output. Each switch is having input and output buffers that lets the message to be stored temporarily at times of contention. Main memory is divided into four memory banks. L2 cache communicate with the main memory through a On-Chip network called fabricated implant interconnect. This interconnect consists of request queue for individual L2 cache. These request queue (RQ)

are connected to the intelligent interface section (IS) of the interconnect. This intelligent IS is capable of mapping a particular request with the corresponding memory bank. If the next request in the queue is requesting for a memory bank, which is being accessed by some other request then this request status is updated to waiting state until the memory bank is free to be reassigned. Information is exchanged between each connected part in the form of packets. A transmission of packet initiates at the end of one core or memory and finishes at the destination memory. Floyd-Warshall algorithm has been used to initialize the routing table which is based on the established connection or links between two connected parts of the architecture. This routing table records the shortest path for each and every pair of connected parts in the network. When a core requires a data from the memory, it first searches for the data in the private L1 cache. If it is not able to find the data in this level it communicates this request to the shared L2 cache through the crossbar. IF the data is not found in the L2 cache too then a request is communicated to the interconnection network. In this network the request queue stores the request. This request is handled by the interface section and it maps the request to the corresponding memory bank. The interface section handles the requests in the request queue simultaneously. If one request is for a memory bank which is having not free, the request is placed on hold until the required memory bank is released. Coherence is enforced with the aid of directory-based MOESI protocol at all caches connected to the upper link of the interconnect with respect to the single cache connected to the lower link [10]. Each block of the L2 cache has a directory entry which contains two fields. The first field is an identifier that specifies the single upper level cache which is the owner of the block that is , it holds the information whether an L1 cache has the block in exclusive, owned or modified state.The second one is a bitmap with as many bits as upper level caches, with those bits set to one that corresponds to the caches having a copy of the block.

3.1 Advantage

Once a memory address is presented each memory module returns with one word per cycle. It is possible to present different addresses to different memory modules in order to enable parallel access of multiple words simultaneously or in a pipelined fashion. Memory banking enhances the parallelism as well as effectively improves the effective memory bandwidth [11]. This parallelism is effectively implemented in the MPFII as shown in the Fig: 1. This leads to an improved performance over few existing architectures. The scalability of the interconnect is enhanced because of the usage of lower configurations of the crossbar. Multiple lower configured crossbar proves to be economical as well as more scalable as compared to high configurations of crossbar [12].

4 Simulation and Results

For the simulationwe have used SPLASH2 benchmark suite [13] [14] [15] and multi2Sim 4.0.2 simulator [9] [16]. We have compared the proposed architecture

with multi-core processor with internal network and the multi-core architecture with ring network. The metrics considered for comparison are execution time and speedup. We varied the cache size and clock speed to evaluate the above architectures [7].

4.1 Impact of Cache Size

To study the impact of cache on the performance of multi-core processors, the number of cores in each architecture was kept constant as 32 and the size of L1 and L2 cache was varied. L2 cache size was varied first keeping the L1 cache size constant. Then L1 cache size was varied keeping L2 cache size constant. The execution time for FFT, cholesky, and barnes benchmark program of the SPLASH2 benchmark suite was analyzed.

Fig: 2, Fig: 3 and Fig: 4shows the CPU execution time for multi-core architecture with Proposed Interconnect on execution of FFT, cholesky and barnes program of the Splash2 benchmark suite. With the enhancement in the cache size the number of misses reduced thus resulting in the reduction in the total CPU execution time. But after certain size the impact reduced. Beyond the size of 512 KB for L1 cache the execution time almost remained constant. Similarly beyond L2 = 8 MB the execution time almost remained constant.

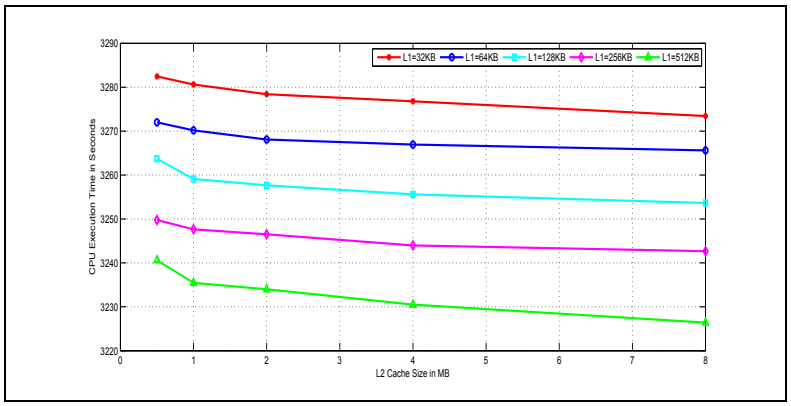

Fig. 2. Execution Time for MPFI

4.2 Performance comparison of Proposed architecture with Existing Architectures

The performance of MPFII has been successfully compared with few of the existing architectures Fig: 5shows the execution time for MPIN, MPRN, and MPFII on execution of the FFT benchmark program. Here the L1 cache size has been kept constant as 512 KB, and L2 cache size as 8 MB. The number of

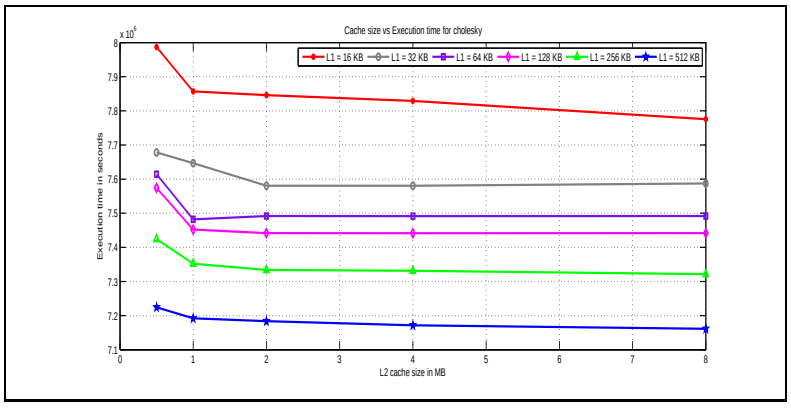

Fig. 3. Execution Time of MPFII for cholesky

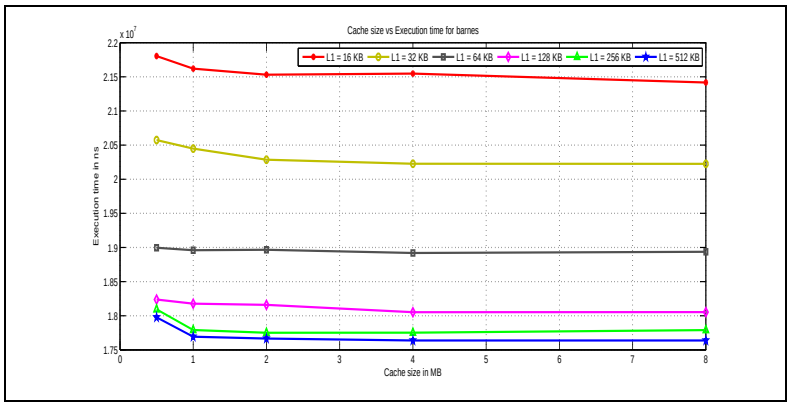

Fig. 4. Execution Time of MPFII for barnes

cores has been varied from 2 - 128 cores as has been described in the previous section. But here we have only shown the variation in performance obtained by keeping the number of cores as 64 and 128. The execution time is the lowest for the proposed interconnect as compared to MPIN and MPRN. The novel interconnect is highly scalable thus able to handle the requests from multiple cores successfully and hence able to reduce the execution time as compared to the other two existing architectures.

5 Conclusions & Future Work

The problem of performance evaluation of Multi-core Organization being most challenging, is interesting too. Keeping a view of the literature the Various Multi-core Organization systems are modeled and few of them have been analyzed.

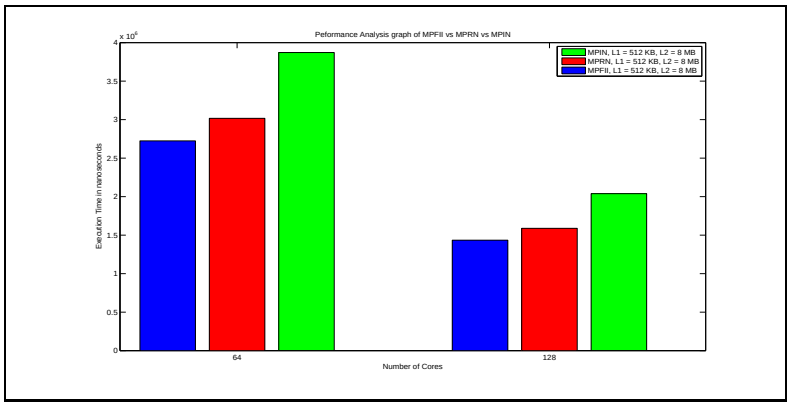

Fig. 5. MPIN vs MPRN vs MPFII on executing FFT

The Multi-core processors are developed to consume lower power and dissipate lower heat. But till date few of the issues still remains unsolved. The utilization of the processor can be highly efficient only when the applications being executed is multi-threaded. Only a few applications exists which are multi-threaded and can be executed parallel. At the same time only a few programmers have the idea and intellect to write programs that can utilize the multi-core processor properly. The interconnect network as well as the memory system also requires much more advancement. In the current work, our primary objective to reduce the delay in the core to memory or memory to memory communication. The second objective was to analyze the performance of multi-core architecture with internal network, ring network and proposed interconnect. We could successfully achieve both the objective by being able to enhance the performance with the proposed architecture. And secondly by analyzing the performance by varying the L1, L2 cache size, the number of cores as well as the core frequency. The performance of the processor is dependent on the cache size but only by increasing the cache size the performance of the processor is not enhanced. This can be concluded by the simulation results obtained. By varying the interconnect network we are able to get a better performance for the proposed architecture as compared to the existing architectures detailed in Section II.

References

1. T. C. Xu, P. Liljeberg, H. Tenhunen, Explorations of optimal core and cache placements for chip multiprocessor, in: NORCHIP, 2011, IEEE, 2011, pp. 1–6.
2. M. Buckler, W. Burleson, G. Sadowski, Low-power networks-on-chip: Progress and remaining challenges, in: Low Power Electronics and Design (ISLPED), 2013 IEEE International Symposium on, IEEE, 2013, pp. 132–134.
3. T. Huang, Y. Zhu, M. Qiu, X. Yin, X. Wang, Extending amdahls law and gustafsons law by evaluating interconnections on multi-core processors, The Journal of Supercomputing (2013) 1–15.

4. D. Pham, S. Asano, M. Bolliger, M. Day, H. Hofstee, C. Johns, J. Kahle, A. Kameyama, J. Keaty, Y. Masubuchi, et al., The design and implementation of a first-generation CELL processor, in: Solid-State Circuits Conference, 2005. Digest of Technical Papers. ISSCC. 2005 IEEE International, IEEE, 2005, pp. 184–592.
5. B. B. Brey, The Intel Microprocessors, Prentice Hall Press, 2008.
6. D. Geer, Chip makers turn to multicore processors, Computer 38 (5) (2005) 11–13.
7. R. Kumar, D. Tullsen, P. Ranganathan, N. Jouppi, K. Farkas, Single-ISA heterogeneous multi-core architectures for multithreaded workload performance, in: ACM SIGARCH Computer Architecture News, Vol. 32, IEEE Computer Society, 2004, p. 64.
8. J. Bui, C. Xu, S. Gurumurthi, Understanding Performance Issues on both Single Core and Multi-core Architecture, Tech. rep., Technical report, University of Virginia, Department of Computer Science, Charlottesville (2007).
9. R. Ubal, J. Sahuquillo, S. Petit, P. López, Z. Chen, D. Kaeli, The Multi2Sim Simulation Framework.
10. R. Ubal, B. Jang, P. Mistry, D. Schaa, D. Kaeli, Multi2Sim: a simulation framework for CPU-GPU computing, in: Proceedings of the 21st international conference on Parallel architectures and compilation techniques, ACM, 2012, pp. 335–344.
11. K. Hwang, Advanced computer architecture, Tata McGraw-Hill Education, 2003.
12. W. J. Dally, B. P. Towles, Principles and practices of interconnection networks, Access Online via Elsevier, 2004.
13. S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, A. Gupta, The SPLASH-2 programs: Characterization and methodological considerations, in: ACM SIGARCH Computer Architecture News, Vol. 23, ACM, 1995, pp. 24–36.
14. S. Akhter, J. Roberts, Multi-core programming, Vol. 33, Intel Press, 2006.
15. C. Bienia, S. Kumar, K. Li, PARSEC vs. SPLASH-2: A quantitative comparison of two multithreaded benchmark suites on chip-multiprocessors, in: Workload Characterization, 2008. IISWC 2008. IEEE International Symposium on, IEEE, 2008, pp. 47–56.
16. R. Ubal, J. Sahuquillo, S. Petit, P. López, Multi2Sim: A Simulation Framework to Evaluate Multicore-Multithread Processors, in: IEEE 19th International Symposium on Computer Architecture and High Performance computing, page (s), 2007, pp. 62–68.
17. R. P. Mohanty, A. K. Turuk, B. Sahoo, Performance evaluation of multi-core processors with varied interconnect networks, in: Advanced Computing, Networking and Security (ADCONS), 2013 2nd International Conference on, IEEE, 2013, pp. 7–11.