Performance Evaluation of Multi-core processors with Varied Interconnect Networks

Ram Prasad Mohanty, Ashok Kumar Turuk, Bibhudatta Sahoo
Department of Computer Science and Enginee'ring
National Institute of Technology
Rourkela 769008, India
Email: {611cs103,akturuk,bdsahu}@nitrkl.ac.in

***Abstract*—Multi-core and heterogeneous processor systems are widely used now a days. Even mobile devices have two or more cores to improve their performance. While these two technologies are widely used, it is not clear which one would perform better and which hardware configuration is optimal for a specific target domain. In this paper, we proposed an interconnect architecture Multi-core processor with Hybrid network(MPHN). The performance of MPHN has been compared with: (i) Multi-core processor with internal network, and (ii) Multi-core processor with Ring network. This architecture substantially minimizes the communication delay in multicore processors.**

***Index Terms*—Multi-core processor; interconnect; performance analysis; impact of cache size.**

I. Introduction

In today's Scenario the usage and application of multi-threaded or multi-core processor system in most of the computer industry is a common implementation across the globe. Also, in mobile device applications, it is becoming more common to see cell phones with multi-cores. 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 [1]. Multi-threaded and multi-core systems also requires more space (area) than a single threaded or single core system [2]. Multi-threaded and multi-core technologies are conceptually different. A thread is the smallest unit of processing that can be scheduled by an operating system, and multiple threads can exist within the same process and share the resources but are executed independently. However, cores in multi-core system have hardware resources for themselves and use them each for processing [3]. In this paper, we evaluate the performance of (i) Multi-core Processor with internal Network, (ii) Multi-core Processor with Ring network, and (iii) Multi-core Processor with hybrid Network in order to find better configuration of performance computing.

II. Architecture and Background

Various work in current literature has explored the multi-core architecture utilising various preformance metrics and application domain. D.M. and Ranganathan [4] have analysed a Single-ISA heterogeneous multi-core architecture for multi-threaded workload performance. The objective was to analyze the performance of multi-core architectures for multithreaded workloads. This section details the benefits of variation in the interconnection network in the multi-core architecture with multithreaded workloads.

A. Exploration of Multi-core Architecture

Various works has analyzed the performance in both single core and multi-core architectures. Julian et al. [5] 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 [6] 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. Few of the architecture of this type have been described below:

1) Multi-core Processor with Internal Network: In Fig. 1 the architecture of a multi-core processor with internal network has been shown. This architecture has a private L1 cache dedicated for each core, to unify the instruction and data requests. These L1 caches are connected to two L2 cache by using a switch [7].

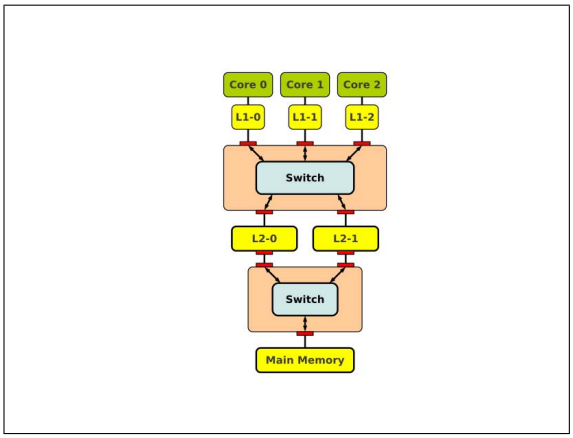

Fig. 1: Multi-core Processor with Internal Network

2) Multi-core Processor with Ring Network: Multi-core processor with Ring Network is shown in Fig. 2. Here each core have private L1 cache. While the L2 uses the interconnect which is implemented using a bidirectional ring to communicate with the memory controller.

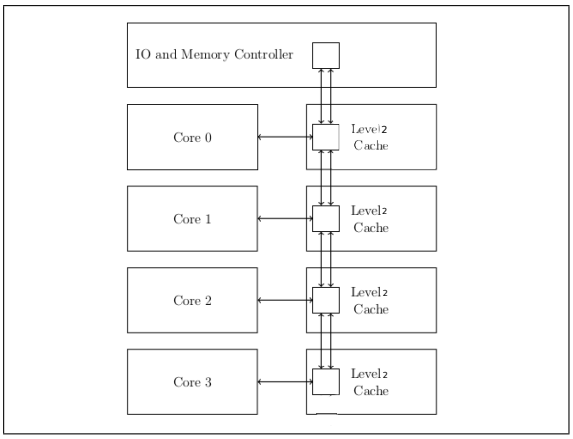

Fig. 2: Multi-core Processor with Ring Network

3) Switches: The network model as shown in the different architecture above consists of set of nodes and switch nodes. Few links that connects to the input and output buffers of a pair of nodes is also included in this network. The nodes existing in this network is classified as the end nodes and switch nodes. An end node is able to send or receive packets from or to another end node. Switches are used only to forward the packets between the switches or end nodes [8]. A link is particularly used for connection between the end node and switch. The internal architecture of a switch is displayed in the Fig: 5. This switch includes a crossbar which connects each input buffers with every other output buffer. A packet at any input buffer can be routed to any of the output buffer at the tail of the switch [9]. The paper has been organized as

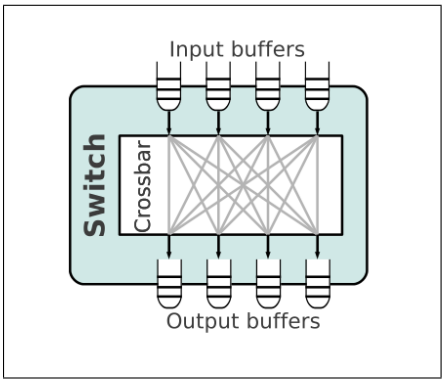

Fig. 3: Switch Architecture Model

follows: 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..

III. Proposed Work

In this paper, we proposed an interconnect architecture that substantially minimizes the communication delay in multcore processors. The performance of the proposed architecture has been compared with existing multicore processor architectures detailed in Section 2.

A. Multi-core Processor with Hybrid Network (MPHN)

The proposed architecture of multi-core processor uses a hybrid network to connect main memory modules with multiple cores. Typical 4 core multi-core processor architecture is shown in Fig 3. This architecture has private L1 caches as shown in Fig. 4.The two L2 caches serve the independent higher level L1 caches. 4 cores of the processor shares one level 2 cache bank. This level 2 cache bank communicates with the main memory banks using the proposed hybrid interconnect. The hybrid interconnect is created using four switches connected to each other with bidirectional links.

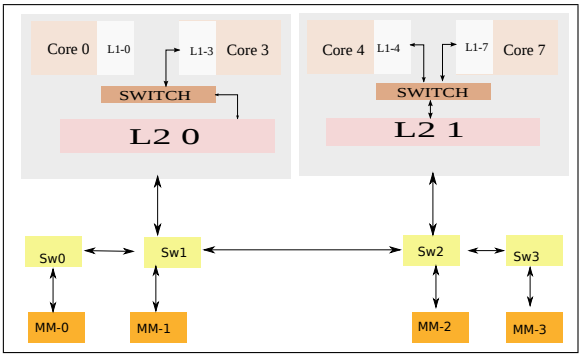

Fig. 4: Multi-core Processor with Hybrid Network

B. Parameters used for Performace Analysis

The parameters of interest to analyze the performance are outlined as follows:

- Execution Time
  - It is the product of the clock cycle time with the sum of CPU cycles and the memory stall cycles [10][11].

$$CPUtime : \hat{CPU}_{time} = clk * t \quad (1)$$

  - where, CPU_{time} is CPU time.
  - clk is CPU clock cycles for a program.
  - t is clock cycle time.
- Speedup
  - Amdahl started with the intuitively clear statement that program speedup is a function of the fraction of a program that is enhanced and to what extent that fraction is enhanced. Over all speedup is computed by the ratio of the old and new execution time.

$$speedup : \hat{s} = (\frac{p_0}{p_1}) = (\frac{e_o}{e_n}) \quad (2)$$

  - where, $\hat{s}$ is Speedup,
  - p_0 is Performance for entire task using the enhancement when possible,
  - p_1 is Performance for entire task without using the enhancement.

- e_o is Old Execution Time.
- e_n is New Execution Time.

- Speedup (Overall)
  - It is the ratio of the execution times. The ratio of the Old exectuion time with the New execution time.

$$Speedup_{overall} = \hat{s}_o = \frac{1}{((1-f_e)+(\frac{f_e}{s_e})} \quad (3)$$

  - where, s_o is Overall speedup,
  - f_e is Fraction enhanced,
  - s_e is Speedup enhanced.

- Execution Time (new):
  - New execution time is dependant on the fraction of program enhanced and the speedup of this enhanced program. Here the way of computing the new exectuion time by using the old execution time and the speedup of enhanced program.

$$ExecutionTime(new) : \hat{e} = et_o((1 - f_e) + (\frac{f_e}{s_e})) \quad (4)$$

  - where, $\hat{e}$ is New Execution Time,
  - $et_o$is Old Execution Time,
  - f_e is Fraction enhanced,
  - s_e is Speedup enhanced.

IV. Simulation Results

For the simulation work, we take the help of the following:

- SPLASH2 benchmark suite.
- multi2Sim 4.0.1 simulator.

A. Construction of Workload

With the advancement of processor architecture over time, benchmarks that were used to compute the performance of these processors are not as practical today as they were before due to their incapability to stress the new architectures to their utmost capacity in terms of clock cycles, cache, main memory and I/O bandwidth [12]. Hence new and enhanced benchmarks have to be developed and used. Bienia et al. [13] have compared the SPLASH-2 and PARSEC on chip-multiprocessors. SPLASH-2 is found to be such a benchmark that has concentrated workloads based on real applications and is a descendant of the SPLASH benchmark. Other such benchmark includes CPUSPEC2006, and Mini Bench. The SPLASH-2 has 11 programs. All experiments were run on systems with 32 bit LINUX operating system and Intel Core 2 Duo processors using the multi2Sim simulator. Here we have used the fft program of the benchamark suite for the performance evaluation.

B. Approach for Simulation

The simulator used for experimentation is Multi2Sim [9] which has been developed integrating some significant characteristics of popular simulators, such as separate functional and timing simulation, SMT and multiprocessor support and cache coherence. Multi2Sim is an application-only tool intended to simulate x86 binary executable files. The simulation is run by several multi-core architecture which have been detailed . In each group, the simulation case is determined by varying the configuration of cache, cores or network connections [4].

C. Simulation Results

Using the Multi2Sim simulator the multi-core architecture with internal networks as shown in Fig: 1 was modelled.

On this model the number of cores was kept constant as 8 and the L1 and L2 cache size was varied while keeping the number of cores constant. The L2 cache size was varied first by keeping the L1 cache size constant and then in the other set The L1 cache size was varied and the same set of L2 cache size was used and the execution time was analysed. The results thus obtained is displayed in the Table: I

TABLE I: Execution time for MPIN

L1 Cache Size	L2 Cache Size	Execution Time
32 KB	512KB	3549.20
	1 MB	3546.17
	2 MB	3542.17
	4 MB	3540.27
	8 MB	3539.11
64 KB	512KB	3542.90
	1 MB	3536.37
	2 MB	3534.81
	4 MB	3532.72
	8 MB	3531.76
128 KB	512KB	3529.91
	1 MB	3528.13
	2 MB	3527.11
	4 MB	3526.52
	8 MB	3525.44
256 KB	512KB	3527.27
	1 MB	3526.15
	2 MB	3525.91
	4 MB	3524.03
	8 MB	3523.91
512 KB	512KB	3525.00
	1 MB	3524.11
	2 MB	3520.80
	4 MB	3518.24
	8 MB	3511.45

Multi-core architecture with ring network as given in Fig: 2 was modelled with 8 cores using the multi2sim simulator. The number of cores was kept constant while varying the L2 cache size for every L1 cache size in the set as shown in Table: II and the execution time is analysed whose details is provided in the same table.

Multi-core architecture with hybrid network as shown in Fig: 4 was modelled by keeping the number of cores as 8. Here also we kept the number of cores constant The L2 cache size was varied for every L1 cache size in the set as shown in Table: II.Here the execution time is analysed.

The analysis for these results has been depicted in the following figures for each Architecture seperately. Fig: 5 shows the graph for CPU execution time for multi-core processor with Internal Network.

Fig: 6 shows the bar graph for CPU execution time for multi-core architecture with External network. Fig: 7 shows the graph for CPU execution time for multi-core architecture with Hybrid network. Cache size has great impact on the performance of the multicore processor architecture. This can be deduced from the results obtained. But as the cache size

TABLE II: Execution time for MPEN

L1 Cache Size	L2 Cache Size	Execution Time
32 KB	512KB	3651.38
	1 MB	3649.20
	2 MB	3647.11
	4 MB	3645.23
	8 MB	3641.33
64 KB	512KB	3637.26
	1 MB	3636.33
	2 MB	3634.21
	4 MB	3631.33
	8 MB	3629.18
128 KB	512KB	3627.11
	1 MB	3624.91
	2 MB	3623.11
	4 MB	3621.79
	8 MB	3618.41
256 KB	512KB	3616.33
	1 MB	3614.28
	2 MB	3612.41
	4 MB	3610.11
	8 MB	3607.63
512 KB	512KB	3603.41
	1 MB	3601.22
	2 MB	3597.18
	4 MB	3594.23
	8 MB	3593.18

TABLE III: Execution time for MPHN

L1 Cache Size	L2 Cache Size	Execution Time
32 KB	512KB	3293.64
	1 MB	3289.47
	2 MB	3287.99
	4 MB	3285.67
	8 MB	3283.81
64 KB	512KB	3282.47
	1 MB	3280.64
	2 MB	3278.41
	4 MB	3276.81
	8 MB	3273.45
128 KB	512KB	3272.00
	1 MB	3270.18
	2 MB	3268.10
	4 MB	3266.94
	8 MB	3265.61
256 KB	512KB	3263.71
	1 MB	3259.11
	2 MB	3257.68
	4 MB	3255.61
	8 MB	3253.64
512 KB	512KB	3249.76
	1 MB	3247.63
	2 MB	3246.51
	4 MB	3243.97
	8 MB	3242.66

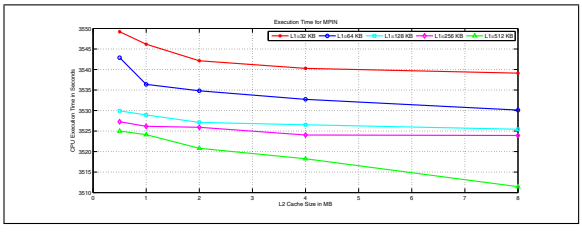

Fig. 5: Execution Time for MPIN

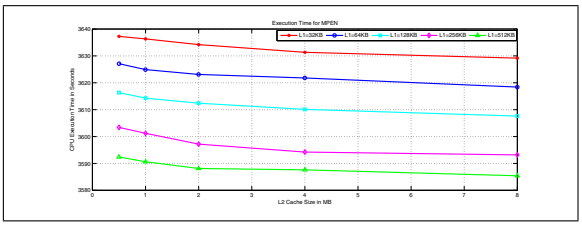

Fig. 6: Execution Time for MPEN

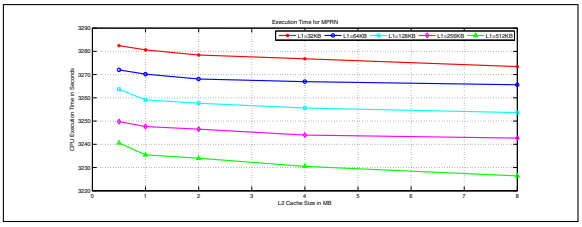

Fig. 7: Execution Time for MPHN

increases, the growth in the performance is lowered beyond certain size of the cache. We can also see that the Multi-core processor with hybrid network provides the lowest execution time as compared to the other two architectures.

The speedup has been computed by the expression given by the expression 2. Here the Old execution time is the execution time optained by executing the same benchmark program for a single core processor. The speedup obtained has been depicted in the figures given. The speedup for Multi-core architecture with internal network has been given in the Fig: 8. The speedup for multi-core architecture for External Network has been depicted in the Fig: 9. The Speedup for multi-core architecture with hybrid network has been given in the Fig: 10.

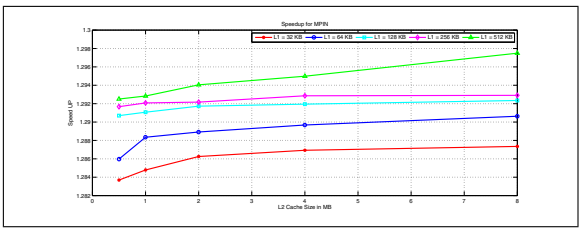

Fig. 8: Speedup for MPIN

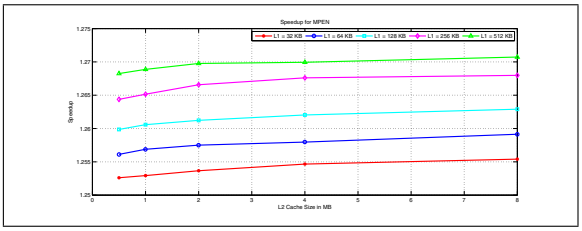

Fig. 9: Speedup for MPEN

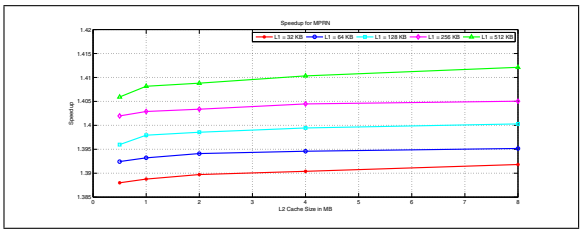

Fig. 10: Speedup for MPHN

V. Conclusions & Future Work

Communication delay is one of the big issues to address in developing future processor architectures. In this paper, we discussed the impact of cache size, interconnect on multicore processor performance. Our conclusions apply to multicore architectures composed of processors with shared L2 cache and running parallel applications. We conducted an experimental evaluation of our proposed interconnect architecture and two other existing architectures, considering number of cores, and cache size. We showed that multicore processor with hybrid interconnect achieve better performance with respect to the two existing architectures.

Few of the factors, not directly addressed in this paper, may affect design choices, such as power,area, and reliability. These factors may, as well, affect the design choice. Nevertheless, evaluating all these alternatives is left for future research.

References

[1] 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.

[2] B. B. Brey, *The Intel Microprocessors*. Prentice Hall Press, 2008.

[3] D. Geer, "Chip makers turn to multicore processors," *Computer*, vol. 38, no. 5, pp. 11–13, 2005.

[4] R. Kumar, D. Tullsen, P. Ranganathan, N. Jouppi, and K. Farkas, "Single-isa heterogeneous multi-core architectures for multithreaded workload performance," in *ACM SIGARCH Computer Architecture News*, vol. 32, no. 2. IEEE Computer Society, 2004, p. 64.

[5] J. Bui, C. Xu, and S. Gurumurthi, "Understanding performance issues on both single core and multi-core architecture," Technical report, University of Virginia, Department of Computer Science, Charlottesville, Tech. Rep., 2007.

[6] R. Ubal, J. Sahuquillo, S. Petit, P. López, Z. Chen, and D. Kaeli, "The multi2sim simulation framework."

[7] R. Ubal, B. Jang, P. Mistry, D. Schaa, and 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.

[8] K. Hwang, *Advanced computer architecture*. Tata McGraw-Hill Education, 2003.

[9] R. Ubal, J. Sahuquillo, S. Petit, and 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.

[10] C. Hughes and T. Hughes, *Professional multicore programming: design and implementation for C++ developers*. Wrox, 2011.

[11] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach*. Elsevier, 2012.

[12] S. Akhter and J. Roberts, *Multi-core programming*. Intel Press, 2006, vol. 33.

[13] C. Bienia, S. Kumar, and 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.

[14] M. Monchiero, R. Canal, and A. González, "Design space exploration for multicore architectures: a power/performance/thermal view," in *Proceedings of the 20th annual international conference on Supercomputing*. ACM, 2006, pp. 177–186.

[15] D. Gove, *Multicore Application Programming: For Windows, Linux, and Oracle Solaris*. Addison-Wesley Professional, 2010.

[16] J. Fruehe, "Multicore processor technology," 2005.

[17] P. Gorder, "Multicore processors for science and engineering," *Computing in science & engineering*, vol. 9, no. 2, pp. 3–7, 2007.

[18] L. Peng, J. Peir, T. Prakash, Y. Chen, and D. Koppelman, "Memory performance and scalability of intel's and amd's dual-core processors: a case study," in *Performance, Computing, and Communications Conference, 2007. IPCCC 2007. IEEE Internationa*. IEEE, 2007, pp. 55–64.

[19] B. JUN-FENG, "Application development methods based on multi-core systems," *American Journal of Engineering and Technology Research Vol*, vol. 11, no. 9, 2011.

[20] D. Hackenberg, D. Molka, and W. E. Nagel, "Comparing cache architectures and coherency protocols on x86-64 multicore smp systems," in *Proceedings of the 42Nd Annual IEEE/ACM International Symposium on microarchitecture*. ACM, 2009, pp. 413–422.

[21] B. Schauer, "Multicore processors–a necessity," *ProQuest Discovery Guides1–14*, 2008.

[22] R. Merritt, "Cpu designers debate multi-core future," *EETimes Online, February*, 2008.

[23] S. Balakrishnan, R. Rajwar, M. Upton, and K. Lai, "The impact of performance asymmetry in emerging multicore architectures," in *ACM SIGARCH Computer Architecture News*, vol. 33, no. 2. IEEE Computer Society, 2005, pp. 506–517.

[24] W. Knight, "Two heads are better than one [dual-core processors]," *IEE Review*, vol. 51, no. 9, pp. 32–35, 2005.

[25] G. Tan, N. Sun, and G. R. Gao, "Improving performance of dynamic programming via parallelism and locality on multicore architectures," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 20, no. 2, pp. 261–274, 2009.

[26] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, and A. Gupta, "The splash-2 programs: Characterization and methodological considerations," in *ACM SIGARCH Computer Architecture News*, vol. 23, no. 2. ACM, 1995, pp. 24–36.

[27] R. P. Mohanty, A. K. Turuk, and B. Sahoo, "Analysing the performance of multi-core architecture," in *1st International Conference on Computing, Communication and Sensor Networks-CCSN,(2012), vol. 62, PIET, Rourkela, Odisha, pp. 258-264, 2012.*, 2012.

[28] D. Stasiak, R. Chaudhry, D. Cox, S. Posluszny, J. Warnock, S. Weitzel, D. Wendel, and M. Wang, "Cell processor low-power design methodology," *Micro, IEEE*, vol. 25, no. 6, pp. 71–78, 2005.

[29] J. A. Brown, R. Kumar, and D. Tullsen, "Proximity-aware directory-based coherence for multi-core processor architectures," in *Proceedings of the nineteenth annual ACM symposium on Parallel algorithms and architectures*. ACM, 2007, pp. 126–134.

[30] D. Olson, "Intel announces plan for up to 8-core processor," *Slippery Brick, March*, 2008.

[31] X. Wang, G. Gan, J. Manzano, D. Fan, and S. Guo, "A quantitative study of the on-chip network and memory hierarchy design for many-core processor," in *Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*. IEEE, 2008, pp. 689–696.

[32] L. Cheng, J. B. Carter, and D. Dai, "An adaptive cache coherence protocol optimized for producer-consumer sharing," in *High Performance Computer Architecture, 2007. HPCA 2007. IEEE 13th International Symposium on*. IEEE, 2007, pp. 328–339.

[33] R. Kumar, V. Zyuban, and D. M. Tullsen, "Interconnections in multi-core architectures: Understanding mechanisms, overheads and scaling," in *Computer Architecture, 2005. ISCA'05. Proceedings. 32nd International Symposium on*. IEEE, 2005, pp. 408–419.