Understanding the Impact of the Interconnection Network Performance of Multi-core Cluster Architectures

Norhazlina Hamid*, Robert Walters, Gary Wills
Electronics and Computer Science, University of Southampton, Southampton, United Kingdom.

* Corresponding author. Email: nh3g11@ecs.soton.ac.uk
Manuscript submitted April 20, 2015; accepted August 20, 2015.
doi: 10.17706/jcp.11.2.132-139

Abstract: Increasing the speed of single-core processors created more heat and produce higher power consumption. Multi-core architectures are proposed for their capability to provide more processing power than single-core processors, without increasing heat and power usage. This paper introduces simulation models of a new architecture for large-scale multi-core clusters to improve the communication performance within the interconnection network. The simulation models are built based on two different flow control mechanisms to identify their impact on the interconnection network performance of the multi-core cluster.

Key words: Multi-core cluster, flow control mechanism, interconnection network, store-and-forward flow control mechanism, wormhole flow control mechanism.

1. Introduction

The exponential growth in computing performance quickly led to more sophisticated computing platforms. This rapid growth increased the demand for faster computing performance; every new enhancement in processors leads to greater performance demands. Moore's Law predicts that the number of transistors on a computer microprocessor will double every two years or so, providing regular leaps in computing power [1]. Over more than four decades, this has driven the impressive growth in computer speed and accessibility. But lately, Moore's Law has begun to show signs of falling, which insists on the emergence of multi-core processors [2].

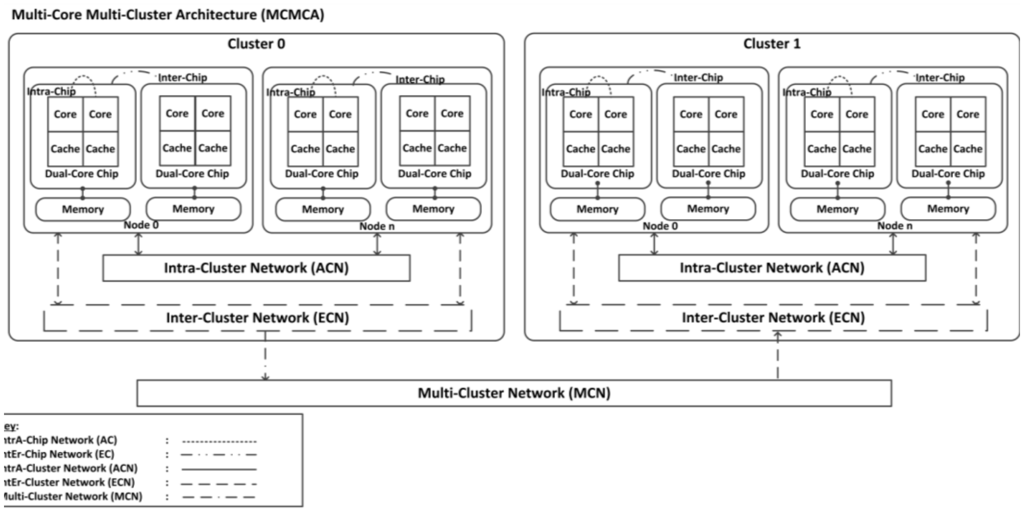
In the past, it was a trend to increase a processor's speed to get better performance. Transistor size has been reduced to increase the number of transistors that can be applied to processor functions and reduce the distance signals must travel [3]. These allowed processor clock frequencies to soar by having more transistors to work with. However, Lei, Qi and Panda [4] identified that it becomes more difficult to speed up processors nowadays by increasing frequency. As processor frequencies increase, the amount of heat produced by the processor increase with it [5]. The solution is to reduce the transistor size because smaller transistors can operate at lower voltages, and this allows the processor to produce less heat. Unfortunately, David Geer [6] demonstrated that as a transistor gets smaller, it will be less able to block the flow of electrons. Thus, smaller transistors keep using the electricity even when they aren't switching which wastes the power. However, transistor can't shrink forever and chip manufacturers have struggled to cap power usage and heat generation which slowing the processor performance. For these reason, computer engineers are building a processor with more processing cores which means placing two or more processing cores on the same chip [7].

Multi-core processors have been widely deployed in clusters for parallel computing as reported in the Top500 supercomputer list [8]. Multi-core cluster architectures have been major designs and have provided an important contribution in computing technology for provisioning additional processing power in high performance computing and communications. These architectures provide an effective solution to improve cluster performance while keeping power consumption manageable. The ability to work on multiple problems simultaneously by taking advantage of parallelism allows for faster execution of applications.

In recent years, many architectures based on multi-core clusters have been proposed to predict and evaluate communication performance [9]-[11]. Previous work on modelling either concentrated on inter-node communication network or focused on high performance multi-core architecture design without considering the effect of interconnection networks on the performance. Perhaps the work by Furhad *et al*. in their paper "An Analysis of Reducing Communication Delay in Network-on-Chip Interconnect Architecture" [12] is the most related to this paper, but with a different interconnection network structure and their analysis only being based on wormhole flow control.

2. The New Architecture

A new architecture known as the multi-core multi-cluster architecture (MCMCA) is introduced in Fig. 1. The structure of MCMCA is derived from a multi-stage clustering system (MSCS) [13] which is based on a basic cluster using single-core nodes. The MCMCA is built up of a number of clusters where each cluster is composed of a number of nodes. Each node of a cluster has a number of processors, each with two or more cores. Cores on the same chip share the local memory and the cluster nodes are connected through the interconnection network.


Multi-Core Multi-Cluster Architecture (MCMCA)
Cluster 0
Cluster 1
Inter-Chip
Intra-Chip
Core
Cache
Dual-Core Chip
Memory
Node 0
Node n
Intra-Cluster Network (ACN)
Inter-Cluster Network (ECN)
Multi-Cluster Network (MCN)
ey:
ntrA-Chip Network (AC)
ntEr-Chip Network (EC)
ntrA-Cluster Network (ACN)
ntEr-Cluster Network (ECN)
Multi-Cluster Network (MCN)
Fig. 1. Multi-core multi-cluster architecture (MCMCA).

There are five communication networks in MCMCA. Two of them are commonly found in any multi-core cluster architecture, the intra-chip network (AC) and the inter-chip network (EC). The new communication networks introduced in this paper are the intra-cluster network (ACN), the inter-cluster network (ECN) and the multi-cluster network (MCN).

The communication between two processor cores on the same chip is the intra-chip network (AC). Messages will be divided into a number of cores by the AC network, which acts as a connector between two or more processor cores on the same chip. Dividing the messages into a number of cores, in theory, results in more than twice the performance with a lower communication delay. An inter-chip network (EC) provides for communications between processors in different chips, but still within the same node. Messages travelling to different chips in the same node will communicate via the intra-chip (AC) and inter-chip (EC) networks to reach their destination. Intra-cluster network (ACN) is an interconnection network to connect nodes within a cluster. Messages that cross between nodes in the same cluster will connected by ACN via the intra-chip (AC) and the inter-chip (EC) networks to complete their journeys.

The longest route for messages to travel will involve ECN and MCN. Messages travelling from their source to their destination between clusters communicate via two interconnection networks. An inter-cluster network (ECN) is used to transmit messages between clusters. The clusters are connected to each other via the multi-cluster network (MCN). When messages reach the other cluster, they will travel using the ECN of the target cluster before arriving at its destination. The same process will continue to the other clusters until all the packets exit the network.

3. The Methodology

3.1. Analysis Assumptions

Both analyses are built on the basis of the following assumptions which are used in similar studies [14], [15]:

1) Each processor generates packets independently, following a Poisson distribution with a mean rate of lambda (λ) and the inter-arrival times are exponentially distributed.
2) The destination of each message is any node in the system with uniform distribution.
3) The number of processors and cores in all clusters are the same and the cluster nodes are homogeneous.
4) The communication switches are input-buffered and each channel is associated with a single packet buffer.
5) The message length is fixed.

3.2. Simulation Model

A simulation model of Multi-core Multi-cluster Architecture was developed using OMNeT++ network simulation software. OMNeT++ [16] is a C++-based discrete-event simulator, which uses the process-interaction approach and has been publicly available since 1997. The structure of the model, including its modules and their interconnection is defined in OMNET++'s topology description language, NED. NED consists of simple module declarations, compound module definitions and network definitions.

The model was built at run time to form a topology based on a scalable fat-tree topology [17] that represents the geometric structure and the Up*/Down* routing [18], a deadlock-free deterministic routing for interaction between the modules. The research experiment focused on buffered flow control mechanism, with store-and-forward flow control representing packet-buffer flow control and wormhole flow control indicating flit-buffer flow control, in order to determine the allocation of the network's resources.

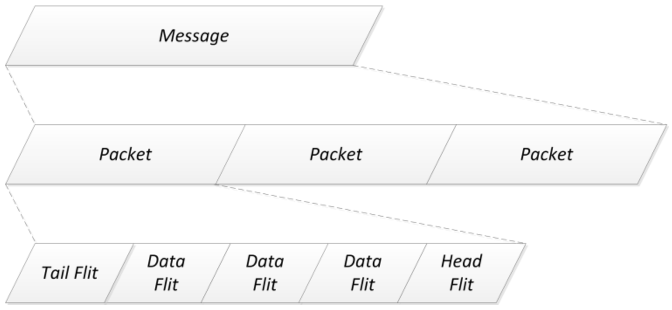


Fig. 2. Illustration of message, packets and flits.

Store-and-forward flow control is a packet switching mechanism where the message to be transmitted is partitioned into a sequence of packets. Each packet is sent separately to the destination according to routing information. Store-and-forward flow control mechanism allocates channel bandwidth and buffer resources to packets, and the resources are used by one packet at a time. With store-and-forward flow control mechanism, each node along a route waits until a packet has been completely received or stored and then forwards the packet to the next node.

Wormhole flow control mechanism allocates buffers and physical channels to flits instead of packets. A packet is decomposed into one or more flits. A flit, the smallest unit on which flow control is performed, can advance once buffering in the next switch is available to hold the flit. The flits of a packet are delivered in a pipeline fashion. Illustration of message, packet and flit is shown by Fig. 2.

The simulation can behave with different inputs and specified parameters of the model, such as the number of cores (nc), number of clusters (C), number of messages to be generated (λg), message length (M) and inter-arrival time (λ). Messages are generated at each node by a *generator* module based on the assumptions that the network traffic follows the uniform Poisson process distribution and the message destinations are uniformly distributed by using a uniform random number generator. Each packet is time-stamped after its generation and the message completion time is defined by a *sink* module on each node to compute message latency.

In each run, 10,000 messages were generated and a total of 100,000 messages were used to gather statistics for every simulation experiment. The simulation experiments were conducted in three phases: warm-up, measurement and drain, to measure steady-state performance. The network has necessarily reached a steady-state once the network is warmed up. This means that the statistics of the network are stationary and no longer changing with time, which will determine an accurate estimation [19].

The communication between processors relies on message passing between the source and destination processors/nodes. The message passes over a channel that directly connects two nodes and might have to pass through several such nodes based on the designate flow control before it reaches its destination. Therefore, each communication involves a lot of latency. Communication networks in MCMCA are divided into internal-cluster and external-cluster. The communication latency in internal-cluster includes messages travelling in the intra-chip network (AC), inter-chip network (EC) and intra-cluster network (ACN) while messages travelling in an external-cluster communicate via two interconnection networks, the inter-cluster network (ECN) and multi-cluster network (MCN). Both clusters will be determined by four factors:

1) Average waiting time at the source node
2) Average transmission delay for a message to cross the networks
3) Average time for the last packet of the message to reach its destination
4) Average waiting time at transfer switch (external-cluster only)

4. Results and Discussion

In order to illustrate the feasibility and the accuracy of the simulation model, a set of experiments were conducted using several system configurations as listed in Table 1 and Table 2. Two different flow control mechanism are used to investigate the impact on interconnection network performance.

Table 1. Simulation Input

Items	**Quantity**
No. of cores (nc)	1, 2, 4
Message Length (M) and Flit Length (F)	32 flits, 256 bytes
No. of cluster, m-port n-tree	8, 8, 2

Fig. 3 and Fig. 5 depicts average message latency vs. traffic generation rate curve based on store-and-forward flow control from case I and case II while Fig. 4 and Fig. 6 shows average message latency with the same traffic generation rate for the wormhole flow control with single-core and multi-core processor.

Table 2. Interconnection Network Parameter

Parameter	**Internal-cluster**		**External-cluster**	
	Case I	Case II	Case I	Case II
Network latency	0.01s	0.02s	0.02s	0.01s
Switch latency	0.01s	0.01s	0.01s	0.05s
Network Bandwidth	1000 b/s	800 b/s	500 b/s	600 b/s

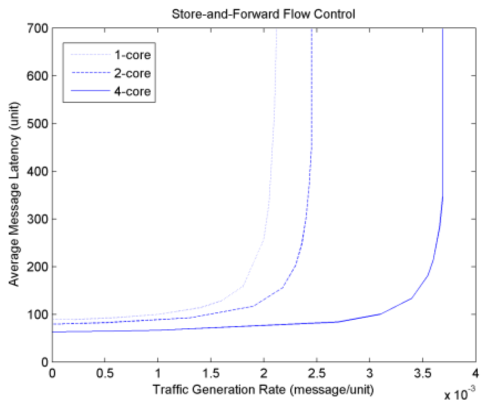


Fig. 3. Average message latency based on store-and-forward flow control.

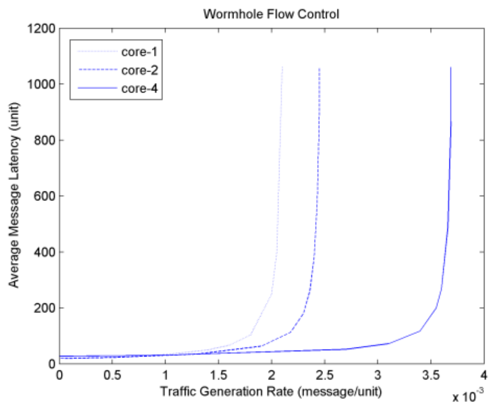


Fig. 4. Average message latency based on wormhole flow control.

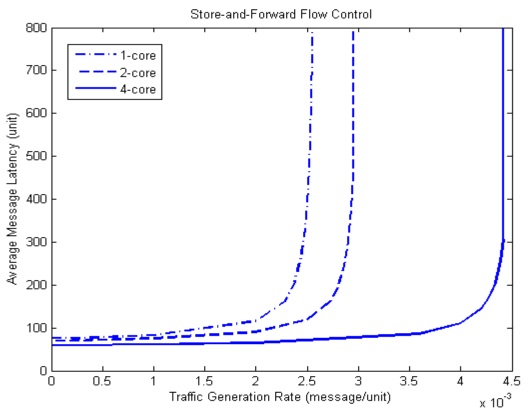


Fig. 5. Average message latency based on store-and-forward flow control.

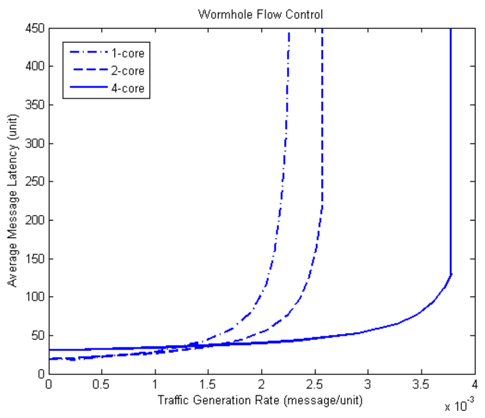


Fig. 6. Average message latency based on wormhole flow control.

All the figures demonstrated that as the traffic increases, the contention latency begins to dominate and the vertical asymptotes of the latency curves are determined by the saturation throughputs of the different flow control mechanism. The increased contention causes the latency to increase as messages must wait for the buffers and channels. All figures also show that the traffic starts to saturate at about 50% of traffic rate capacity with single-core and dual-core processor but higher throughput is achieved with quad-core processors. Store-and-forward flow control fares better, improving the performance by almost 51% compared to wormhole flow control. While wormhole flow control performs better at light traffic, it does not achieve optimal throughput due to the probability of blocking happen while traversing the network.

Generally, as the number of processor cores increases, the load allocation time increases, which can affect the pipelining of the router. The zero-load latency tends to increase and may slightly decrease the saturation throughput in the internal-cluster because of the additional time required to allocate the task into a number of cores. The zero-load latency assumption is that a packet has never contended for network resources with other packets. It gives a lower bound on the average latency of a packet through the network. The fact that multiple messages can be in transmission and attempt to use the same network link at the same time will cause a problem in the network. If this problem happens, some of the message transmission must be blocked while other messages are allowed to proceed which affect the performance. Fig. 7 and Fig. 8 demonstrate an average message latency in internal-cluster and external-cluster with the same system configuration as in Table 1 and Table 2 with single-core and multi-core processor. For the same traffic rate, both flow control mechanisms achieve lower latency in internal-cluster with single-core and dual-core processor, but the latency increases massively with quad-core processor due to higher message distribution in the internal-cluster as shown in Fig. 7. Since more messages can be loaded in internal-cluster with multi-core processors, the external-cluster latency decreases as in Fig. 6.

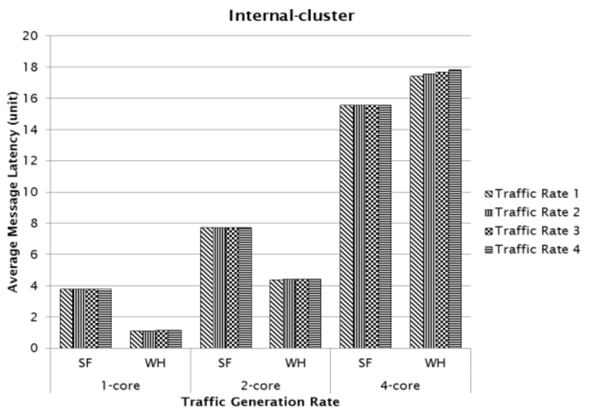


Fig. 7. Average Message Latency in Internal-cluster.

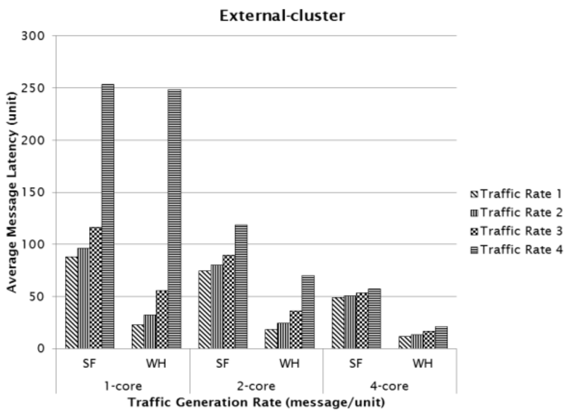


Fig. 8. Average Message Latency in External-cluster.

5. Conclusions

This paper has presented a new architecture for building and measuring the performance of interconnection networks in Multi-Core Multi-Cluster Architecture (MCMCA). The simulation models in this paper are general and applicable to predict the performance of a multi-core cluster. The experiments are produced based on two different flow control mechanisms to provide a comparative analysis of multi-core cluster architecture with a varying number of parameter and system configurations. Higher performance of large clusters can be achieved by optimizing the interconnection network communication. The choice of routing algorithm used in a network sets a limit on the achievable throughput and a lower-bound on the packet latency. The results show that the wormhole flow control performs better at light traffic, however it does not achieve optimal performance throughput. Though store-and-forward flow control produces larger latency, the saturation point is greater.

Acknowledgment

The authors acknowledge the award of a Malaysia Fellowship Training scholarship (HLP), Public Service Department of Malaysia, to Norhazlina Hamid to allow this research to be undertaken.

References

[1] Intel. (1997). Moore's Law and Intel Innovation. From http://www.intel.com/about/companyinfo/museum/exhibits/moore.htm?wapkw=moore+laws
[2] Karmakar, N. (2011). *Multi-core Architecture*. North Maharashrta University, India.

[3] Schauer, B. (2008). Multicore processors-a necessity. *ProQuest Discovery Guides*, 1-14.

[4] Chai, L., Gao, Q., & Panda, D. K. (2007). Understanding the impact of multi-core architecture in cluster computing: A case study with intel dual-core system. *Proceedings of Seventh IEEE International Symposium on Cluster Computing and the Grid* (pp. 471-478).

[5] Pase, D. M., *et al*. (2005). *A Comparison of Single-Core and Dual-Core Opteron Processor Performance for HPC*.

[6] Geer, D. (2005). *Industry Trends: Chip Makers Turn to Multicore Processors*. 11-13.

[7] Burger, T. W. (2005). *Intel Multi-core Processors: Quick Reference Guide*. From https://software.intel.com/en-us/articles/intel-multi-core-processors-quick-reference-guide

[8] Admin. (2014). *Top500 Supercomputer Sites*. From http://www.top500.org/lists/2014/06/

[9] Soryani, M., Analoui, M., & Zarrinchian, G. (2013). Improving inter-node communications in multi-core clusters using a contention-free process mapping algorithm. *The Journal of Supercomputing*, *66(1)*, 488-513.

[10] Shainer, G., Lui, P., Hilgeman, M., Layton, J., Stevens, C., Stemple, W., *et al*. Maximizing application performance in a multi-core, NUMA-aware compute cluster by multi-level tuning. *Supercomputing*. 226-238.

[11] Khanyile, N. P., Tapamo, J. R., & Dube, E. (2012). An analytic model for predicting the performance of distributed applications on multicore clusters. *IAENG International Journal of Computer Science, 39(3)*, 312-320.

[12] Furhad, H., Haque, M. A., Kim, C. H., & Kim, J. M. (2013). An analysis of reducing communication delay in network-on-chip interconnect architecture. *Wireless Personal Communications*, *73(4)*, 1403-1419.

[13] Shahhosein, H. S., Naderi, M., & Buyya, R. (2000). Shared memory multistage clustering structure, an efficient structure for massively parallel processing systems. *Proceedings of the Fourth International Conference/Exhibition on High Performance Computing in the Asia-Pacific Region*: *Vol. 1* (pp. 22-27).

[14] Hamid, N., Walters, R. J., & Wills, G. B. (2015). Performance evaluation of multi-core multi-cluster architecture (MCMCA). *Delivery and Adoption of Cloud Computing Services in Contemporary Organizations*, 219.

[15] Hamid, N., Walters, R. J., & Wills, G. B. (2015). An analytical model of multi-core multi-cluster architecture (MCMCA). *Open Journal of Cloud Computing (OJCC), 2(1)*, 1-12.

[16] Varga, A., & Hornig, R. (2008). An overview of the OMNeT++ simulation environment. *Proceedings of The 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*. Marseille, France.

[17] Lin, X. (2003). An efficient communication scheme for fat-tree topology on infiniband networks. Department of Information Engineering and Computer Science, Feng Chia University Taiwan.

[18] Robles-Gómez, A., Bermúdez, A., & Casado, R. (2011). A deadlock-free dynamic reconfiguration scheme for source routing networks using close up*/down* graphs. *IEEE Transactions on Parallel and Distributed Systems*, *22(10)*, 1641-1652.

[19] Dally, W. J., & Towles, B. P. (2004). *Principles and Practices of Interconnection Network*: *Morgan Kaufmann.*

Norhazlina Hamid received a bachelor in information technology (Hons) from Northern University of Malaysia (UUM) in 2000 and an MSc in information technology from MARA University of Technology (UiTM) in 2003. She is now a final year PhD student in the School of Electronics and Computer Science of University of Southampton.

Robert Walters worked for almost fifteen years working in commercial banking, before leaving to study mathematics with computer science at University of Southampton. After completing his degree, he worked for several years as a software developer before returning to Southampton as a research fellow in 1996. Since then he has completed his PhD in 2003 and is currently employed as a lecturer in the School of Electronics and Computer Science of University of Southampton.

Gary Wills is a senior lecturer in computer science at the University of Southampton. He graduated from the University of Southampton with an Honours degree in electromechanical engineering, and then a PhD in industrial hypermedia systems. He is a chartered engineer and a member of the Institute of Engineering Technology and a fellow of the Higher Educational Academy. He is also a visiting professor at the Cape Peninsular University of Technology, SA. Gary's main research interests are in personal information environments (PIEs) and their application to industry, medicine and education.