Hierarchical Architecture for Network-on-Chip Platform

Liang-Yu Lin, Huang-Kai Lin, Cheng-Yeh Wang, Lan-Da Van*, and Jing-Yang Jou, Fellow, IEEE
Department of Electronics Engineering
Institute of Electronics
National Chiao Tung University
Taiwan, R.O.C.
*Department of Computer Science,
National Chiao Tung University
Taiwan, R.O.C.

ABSTRACT

In this paper, we propose one hierarchical 2-D mesh Network-on-Chip (NoC) platform to support applications with the complexity of several hundreds of tasks or with huge amount of transmission data. Moreover, applying the task binding method by considering communication amount, communication data contention and bandwidth penalty to enhance the system overall performance of the new architecture. Modeling the NoC system data transmission behavior at system level is applied to predict system overall performance and an automatic NoC system performance simulation tool is also built. Therefore, architecture and designers can predict the system performance and obtain all parameters of the designed platform at system abstraction level. The experimental results show that the overall system throughput, the latency, and the saving of redundant transactions are improved by 27%, 14.4% and 21.8% respectively under the communication dominated situation.

1. INTRODUCTION

As the advance of semiconductor technology, it is possible to integrate hundreds of processing elements on a single chip in the next decade. Hence, hundreds of intellectual properties (IPs) including general propose processors and application specific functional blocks will be integrated onto a single chip. When the time comes, communication between the components will be the critical factor for system performance. Both the architecture and designer need a communication-driven system design methodology to make the whole design iteration seamlessly. However, designers will encounter several new problems.

The first problem, the ever-shrinking feature size causes the gate delay scaling down linearly, whereas the wire delay remains constant. Hence, the wire delay will become more critical than the gate delay [1]. Although the wire delay can be conquered by wire pipelining techniques, the problem of timing uncertainty is still hard to treat. On the other hand, the clock skew can not be neglected any more and clock synchronization becomes another problem for designers. It is almost impossible to synchronize all components on a chip with single clock.

Second, traditional shared-bus based network architecture is the most popular architecture in current System-on-Chip (SoC). However, high data transmission contentions between masters reduce the system performance and raise the power consumptions. Moreover, buses only can handle 3 to 10 computation elements and can not scale to higher numbers [2][3][4]. Therefore, communication between computing components becomes the critical factor of system performance.

Last, system design methodology with integrating more computing components and verification become more complex. A communication-driven system design methodology will be applied.

In order to aid these problems, a communication-driven system design methodology was proposed [5]. The methodology separates the system design into two parts: function modeling and architecture modeling. Many design methodologies based on the concept are proposed. M. Sgroi *et al.* propose a Network-on-Chip (NoC) approach to partition the communication into layers to maximize the reuse and provide programmers with an abstraction of the underlying communication framework [6]. D. Ching *et al.* proposed their idea includes integrated modeling, simulation and implementation environment [7] to perform simulation to find the optimal network configuration. In [8], an algorithm called NMAP which can be applied to both the single-path routing and the spilt-traffic routing to map the cores onto NoC architecture under bandwidth constraints is proposed. A simple packet switching communication model to estimate the communication time and to propose a two-step genetic algorithm to map a parameterized task graph onto the 2-D mesh NoC architecture, which minimizes the overall execution time of the task graph is proposed in [9].

In this paper, we propose the idea called "Hierarchical Architecture for NoC" which is similar to the freeways between metropolises to enhance the 2-D topology NoC architecture. The organization of this paper is described as following. In Section 2, we proposed hierarchical 2-D mesh NoC platform and the detail architecture. In Section 3, we present the communication contention-aware task binding and path assignment methodology for the platform. The experiment flow and experimental results are shown and discussed in Section 4. Finally, conclusion is remarked in the last section.

2. ARCHITECTURES

In this work, we use a switch architecture which is based on the latency-insensitive concepts [10][11] and utilize the virtual-circuit switching technique to achieve high bandwidth utilization, guaranteed bandwidth and predictable latency under high communication loading such that the predictable characteristics are able to support real-time applications.

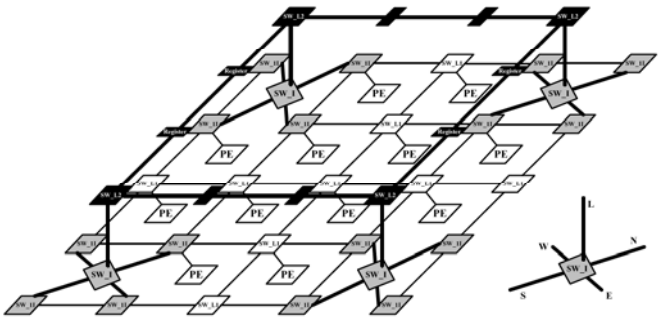

Figure 1. Hierarchical 2-D mesh NoC platform.

Figure 1 shows the proposed hierarchical 2-D mesh NoC platform. In this architecture, two 2-D mesh switch networks are connected by using interchange switches (SW_I). We define the added network as network level-2 (L2) including SW_I and SW_L2; the traditional part is defined as network level-1 (L1). Every three PEs in x-direction and y-direction, the PE is replaced by SW_I and SW_L2. The port names of a SW_I are shown in Figure 2. Some vertical or horizontal physical channels are disconnected in order to release switch ports to connect to SW_I. We also assume that the coordinate of an SW_I is (x, y). If the sum of x and y is odd, the horizontal physical channels will be disconnected; if the sum is even, the vertical ones will be disconnected. The buffers between two SW_L2 are relay stations [12] without switches, and they transmit data forward directly in the next clock cycle. A parameter named the physical channel width ratio R is used to characterize the hierarchical architecture, and is defined as

$$R = \frac{\text{Physical channel width}_{L2}}{\text{Physical channel width}_{L1}} \quad (1)$$

The physical channel width means the available size of the transmission data. If the size of L1 transmission data is equal to one-word, and that of L2 data is equal to four-word, R will be given by four.

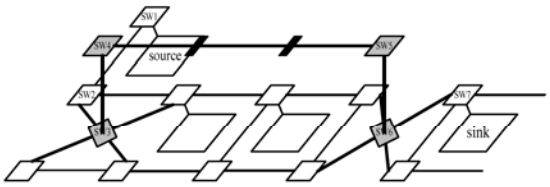

Figure 4. Example of data transmission passing through hierarchy.

For the L1 network, the width of the physical channel is one-word width. On the other hand, for the L2 network, the width is four-word width. Considering an example of a connection path shown in Figure 4, the source PE transmits data to the sink PE through L2. A four-word data can be transmitted from SW2 to SW3 when the four one-word data from SW1 is available. SW4 makes transactions to SW5 until the three four-word data from SW3 is available, because a SW_L2 transmits three four-word data continuously. Hence, for the SW_L2 buffers connected to another SW_L2, the weights of the round-robin scheduling are assigned at least three or six.

3. Methodology of Task Mapping and Path Assignment

In this work, the goal of task mapping is to minimize the overall communication resource usage. In this section, we improve the methodologies of task mapping and path assignment by enhancing the cost functions which used in previous work [12]. In [12], the main idea of the cost function for task mapping is that any pair of connected tasks with heavy traffic loading should be allocated as next to each other as possible. A cost function ξ' is proposed to express the criterion as follows.

$$\xi' = \sum_{\text{pair of processors, X}} D \times \left(1 + \frac{C_{\text{amount, X}}}{C_{\text{amount, MAX}}}\right) \quad (2)$$

Here, D and $C_{\text{amount, X}}$ denote the Manhattan distance between the source PE and the destination PE, and the communication amount between a pair of connected processors, respectively. The first term, ($D \times 1$), of the cost function indicates the resource usage of the virtual channels. It is also the conventional cost used in the FPGA placement algorithm. Not only the distance between a pair of connected PE but also the communication amounts between a pair of connected tasks will affect the system performance. The second term, ($D \times C_{\text{amount, X}} / C_{\text{amount, MAX}}$), of the cost function represents the normalized value that indicates the communication effect of the physical channels, where $C_{\text{amount, MAX}}$ is the maximum value of all $C_{\text{amount, X}}$.

On the other hand, the connection path assignment cost function ξ is enhanced as described in the following.

$$\xi - \sum_{\text{each path, X}} \sum_{\text{each channel, A of X}} \left(1 + \frac{N_{\text{data, A}} + \rho_{\text{A}}}{C_{\text{amount, MAX}}} + B_{\text{penalty, A}}\right) \quad (3)$$

To contrast with equation (2), the distance of the path assignment is the real physical channel number passed by rather than the Manhattan distance used in the task mapping. Furthermore, the transmission data number $N_{\text{data, A}}$, the contention factor ρ_{A} and bandwidth penalty $B_{\text{penalty, A}}$ are included to describe the transmission data number of a path through a physical channel, the total effect of communication contention on a physical channel and the bandwidth constraint of a physical channel introduced by other paths.

The detailed expression of $N_{\text{data, A}}$, ρ_{A} and $B_{\text{penalty, A}}$ are revealed in the following.

$$N_{\text{data, A}} = \begin{cases} C_{\text{amount, X}} & \text{, if A} \in \text{L1} \\ \dfrac{C_{\text{amount, X}}}{R} & \text{, if A} \in \text{L2} \end{cases} \quad (4)$$

where $C_{\text{amount, X}}$ is the communication amount of a path X; $\boldsymbol{R}$ is the physical channel width ratio described in equation (1). Because of the difference between the physical channel width of L1 and that of L2, transmission data number of a physical channel is calculated according to whether this physical channel belongs to.

$$\rho_{\text{A}} = \sum_{\substack{\text{B} \in \text{other paths} \\ \text{in the channel}}} N_{\text{data, B}} \times \Gamma_{B \to A} \quad (5)$$

In equation (5), $N_{\text{data, B}}$ and $\Gamma_{\text{B} \to \text{A}}$ denote the transmission data number of other paths through this physical channel, and the contention density. The calculation of $N_{\text{data, B}}$ is mentioned above, and the contention density is formulated as followed.

$$\Gamma_{\text{B} \to \text{A}} = \frac{t_{\text{overlap, (A, B)}}}{t_{\text{commun, B}}} \quad (6)$$

The contention density of each communication path pair can be derived from the communication profile in time domain as shown in Figure 5.

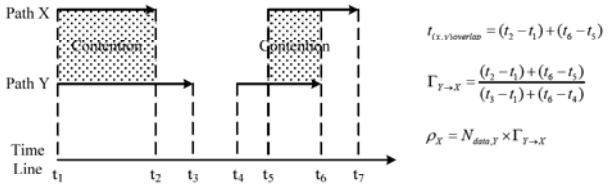

Figure 5. Contention occurrence in time domain.

The arrow means that the path transmits data in a time period, and the time period without an arrow means that there is no data transmission on the path. For example, the arrows of Path X and Path Y overlap in time domain. If there are physical channels both Path X and Path Y passed through, the two paths overlap in spatial domain. When the overlap in time domain and spatial domain is occurred, this

leads to the contention. Therefore, to calculate the contention factor of Path X for a physical channel, pick up other paths in this physical channel from the communication profile and calculate the overlapped transmission data number through the functions above. To contrast with the equation (2), the communication profile is obtained and the contention affected parameters are calculated after the performance analysis. Therefore, the contention factor, ρ_A, is zero during the connection path assignment procedure of the first iteration.

In equation (7), the bandwidth penalty is described as follows.

$$B_{\text{penalty, A}} = \begin{cases} \alpha \times \dfrac{B_{\text{demanded, A}} - B_{\text{provided, A}}}{B_{\text{provided, A}}}, & \text{if } B_{\text{demanded, A}} > B_{\text{provided, A}} \\ 0, & \text{if } B_{\text{demanded, A}} \le B_{\text{provided, A}} \end{cases} \quad (7)$$

where α, $B_{\text{demanded, A}}$ and $B_{\text{provided, A}}$ denote the penalty weight, the demanded bandwidth used by communication paths in a physical channel, and the provided bandwidth of a physical channel, respectively. The demanded bandwidth is decided by the performance constraint of the application. If the bandwidth usage of a physical channel exceeds the provided bandwidth, the performance constraint will be violated.

The connection path assignment cost function proposed in (3) indicates that the efficiency of a communication path is dominated by the distance of the connection path, the selection of the hierarchical or plane physical channels, the path number to share the physical channel and the bandwidth usage.

4. EXPERIMENTAL RESULTS

In this section, we present an enhanced framework based on the previous work [12] and is shown in Figure 4.1.

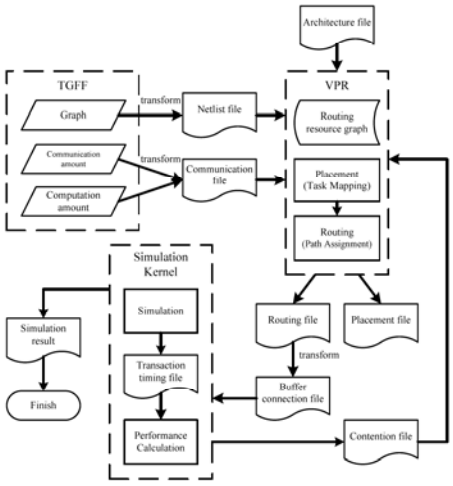

Figure 6. Experimental framework

Hierarchical Bandwidth Scalability Analysis

We assume the physical channel width of L2 is larger than that of L1. Figure 7 and Figure 8 shows the relationship of the latency and the throughput versus R. In this experiment, 100 task graphs are generated from TGFF [13] randomly for each case where each task graph has at least 210 to 250 tasks and the maximum input/output of each task ranges from 7 to 10. The other simulation environment settings are the communication factor of four and buffer size of two. The value of R of hierarchical architecture is set to 1, 2, 4, and 8.

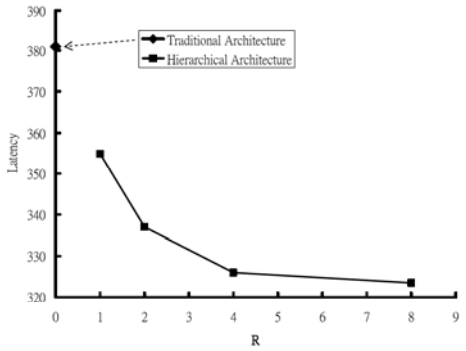

Figure 7. Latency versus different R.

In Figure 4.3, the single node on y-axis denotes the latency of the traditional architecture with R of zero. The curve denotes the relationship between the latency and R. The latency decreases as R increases and is saturated at approximately 320 clock cycles.

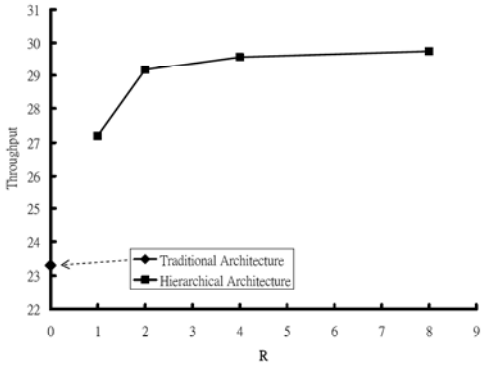

Figure 8. Throughput versus different R.

In Figure 8, the single node on y-axis denotes the throughput of the traditional architecture with R of zero. The curve denotes the relationship between the throughput and R. The throughput increases and tends to saturate while the increasing of R. Through these simulation results, the designer can get trade-off between the physical channel width of L2 and the system performance.

Performance Analysis

100 task graphs generated from TGFF randomly are used for each case, and each task graph has at least 210 to 250 tasks. The maximum inputs/outputs of each task is 7 to 10 which indicates that there are many communication paths between tasks. The communication amount is modified by multiplying the communication factor. In other words, the communication factor is the ratio of the communication amount to the computation amount. The magnitude of the communication factor indicates the communication loading degree of a communication path.

In this experiment, the value of communication factor is set to 0.25, 0.5, 1, 2, and 4. The task graph with the communication factor less than one implies that the application is computation intensive. On the other hand, the communication factor larger than one means that the application is communication intensive. Fail rate is defined as the number of failed transactions over the number of the total transactions. Since a failed transaction needs to transmit the same data again, unnecessary power consumption will be induced. Higher fail rate, more power consumption used for useless transactions; thus,

the total power consumption increases. The relationship between the fail rate and the communication factor by comparing the traditional architecture with the proposed hierarchical one is shown in Figure 9.

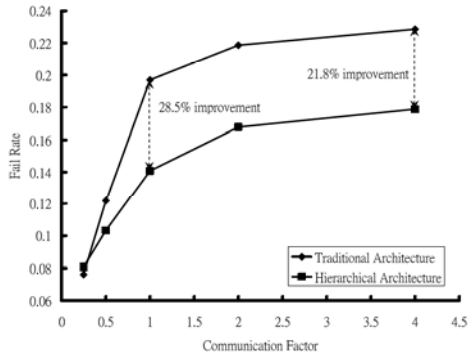

Figure 9. Fail rate versus different communication factor.

The fail rate increases with the increasing of the communication factor and saturates at approximately 22% for traditional architecture and 17% for hierarchical one. The trend of the curve means that the fail rate would be under controlled even under communication intensive conditions. Then, the proposed hierarchical architecture improves the fail rate of 28.5% with communication factor of one and 21.8% for communication factor of four.

Table.1. Comparison between traditional and hierarchical architecture at communication factor = 4 and buffer size = 2.

	Fail Rate	Latency	Throughput
Traditional	0.2285	380.9 (cycles)	23.29
Hierarchical	0.1788	326 (cycles)	29.57
Improvement	21.75%	14.41%	26.96%

Table 1 shows the comparison between the hierarchical and the traditional architecture under the communication factor of four and the buffer size of two. We can see that the hierarchical architecture improves the system performance in terms of fail rate, latency and throughput under the computation intensive applications.

5. CONCLUSIONS

This paper not only proposes a new hierarchical switch-based network platform design but also the new cost functions for task mapping and path assignment methodology adapted to the hierarchical architecture. The hierarchical 2-D mesh architecture performs better than the previous 2-D architecture without hierarchy under the more complex and heavy communication applications. The experimental results indicate that the cost functions and our hierarchical architecture not only increase the system utilization and effectively improves the network throughput but also provide the features of high bandwidth utilization and small latency. Compared with these results of the traditional 2-D mesh architecture, the significant improvements can be attained. The overall system throughput, the latency, and the saving of redundant transactions can be improved by 27%, 14.4% and 21.8% respectively under the communication dominated situation.

REFERENCES

[1] R. Ho, K. Mai, and M. Horowitz, "The future of wires," in Proceedings of IEEE, vol. 89, no. 4, pp. 490-504, April 2001.

[2] Axel Jantsch, and Hannu Tenhunen, "Networks on Chip," Kluwer Academic Publishers, 2003.

[3] Adrijean Adriahantenaina, Herve Charlery, Alain Greiner, Laurent Mortiez and Cesar Albenes Zeferin, "SPIN: a scalable, packet switched, on-chip micro-network," in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, supplements 70-73, 2003.

[4] Cesar Albenes Zeferino and Altamiro Amadeu Susin, "SoCIN: a parametric and scalable network-on-chip," in Proceedings of the 16th Symposium on Integrated Circuits and Systems Design, pp. 169-174, Sep. 2003.

[5] Luca Benini and Giovanni De Micheli, "Networks on chips: a new SoC paradigm," IEEE Computer Magazine, vol. 35, issue 1, pp. 70-78, Jan. 2002.

[6] M. Sgroi, M. Sheets, A. Mihal, K. Keutzer, S.Malik, J. Rabaey, and A. Sangiovanni-Vincentelli, "Addressing the system-on-chip interconnect woes through communication-based design," Proceedings of 38th the Design Automation Conference, pp. 667-672, June 2001.

[7] Doris Ching, Patrick Schaumont and Ingrid Verbauwhede, "Integrated modeling and generation of a reconfigurable network-on-chip," in Proceedings of the 18th International Parallel and Distributed Processing Symposium, pp. 139-145, 2004.

[8] Srinivasan Murali and Giovanni De Micheli, "Bandwidth-constrained mapping of cores onto NOC Architectures," in Proceedings of the Design, Automation and Test in Europe Conference and Exhibition, vol. 2, pp. 896-901, Feb. 2004.

[9] Tang Lei and Shashi Kumar, "A two-step genetic algorithm for mapping task graphs to a network on chip architecture," in Proceedings of Euromicro Symposium on Digital System Design, pp. 180-187, Sep. 2003.

[10] Luca P. Carloni, Kenneth L. McMillan, Alexander Saldanha, and Alberto L. Sangiovanni-Vincentelli, "A methodology for correct-by-construction latency insensitive design," in Proceedings of the International Conference on Computer-Aided Design, pp. 309-315, 1999.

[11] Luca P. Carloni and Alberto L. Sangiovanni-Vincentelli, "Coping with latency in SOC design," IEEE Micro,, vol. 22, issue 5, pp. 24-35, Sep.-Oct. 2002.

[12] L.Y. Lin, C.Y. Wang, P.J. Huang, C.C. Chou, and J.Y. Jou, "Communication-driven task binding for multiprocessor with latency insensitive Network-on-Chip," Asia and South Pacific Design Automation Conference, 2005.

[13] Robert P. Dick, David L. Rhodes, and Wayne Wolf, "TGFF: task graphs for free," in Proceedings of the 6th International Workshop on Hardware/Software Codesign, pp. 97-101, 1998.