http://dx.doi.org/10.5573/JSTS.2011.11.4.318 JOURNAL OF SEMICONDUCTOR TECHNOLOGY AND SCIENCE, VOL.11, NO.4, DECEMBER, 2011

Hierarchical Multiplexing Interconnection Structure for Fault-Tolerant Reconfigurable Chip Multiprocessor

Yoonjin Kim

Abstract—**Stage-level reconfigurable chip multiprocessor (CMP) aims to achieve highly reliable and fault tolerant computing by using interwoven pipeline stages and on-chip interconnect for communicating with each other. The existing crossbar-switch based stage-level reconfigurable CMPs offer high reliability at the cost of significant area/power overheads. These overheads make realizing large CMPs prohibitive due to the area and power consumed by heavy interconnection networks. On other hand, area/ power-efficient architectures offer less reliability and inefficient stage-level resource utilization. In this paper, I propose a hierarchical multiplexing interconnection structure in lieu of crossbar interconnect to design area/power-efficient stage-level reconfigurable CMP. The proposed approach is able to keep the reliability offered by the crossbar-switch while reducing the area and power overheads. Experimental results show that the proposed approach reduces area by up to 21% and power by up to 32% when compared with the crossbar-switch based interconnection network.**

Index Terms—**Chip multiprocessor (CMP), reconfigurable architecture, interconnection, fault-tolerant computing, low power**

I. Introduction

With the recent popularity of multi-core systems, traditional fault-tolerant core-level approaches have been able to leverage the inherent redundancy present in large chip multiprocessors (CMPs) [1, 2]. However, both the historical designs and their modern incarnations, because of their emphasis on core-level redundancy, incur high hardware overhead and can only tolerate a small number of defects [2]. With the increasing defect rate in semiconductor technology, it will not be uncommon to see a rapid degradation in throughput for these systems as single device failures cause entire cores to be decommissioned, often times with the majority of the core still intact and functional [3, 4]. In contrast, the works in [5, 6] argue the case for reconfiguration and redundancy at a finer granularity. These works present stage-level reconfigurable CMP architecture designed as a network of pipeline stages, rather than isolated cores in a CMP. Within the stage-level reconfigurable CMP, pipeline stages can be selected from the pool of available stages to act as logical processing cores. Such a reconfigurable CMP contributes to the area of recovery and reconfiguration by proposing a radical architectural shift in processor design. Motivated by the need for finer-grain reconfiguration, net-worked pipeline stages were identified as the effective trade-off between cost and reliability enhancement.

However, such a reconfigurable CMP architecture suffers from its significant area and power consumption by heavy interconnections among the pipeline stages with many cores [5]. On the other hand, the architecture [6] having less interconnections shows less flexibility and weaker fault-tolerant capability than the former case [5]. Therefore, efficient interconnection structure is necessary to provide cost-effective stage-level reconfigurable CMP with great flexibility.

This work addresses cost-effective interconnection

Manuscript received Jul. 13, 2011; revised Oct. 3, 2011.
Dept. of Computer Science, Sookmyung Women's University, Korea
E-mail : ykim@sookmyung.ac.kr

design approach to the stage-level reconfigurable CMP with great flexibility. In this paper, I present a novel approach to reduce power and area by using hierarchical multiplexing interconnection structure in CMP in-stead of crossbar-switches for stage-level reconfiguration. The proposed approach does not affect the performance, functionality and reconfigurability of the original structure. The validation of the proposed approaches is demonstrated through gate level implementation and obtain area, delay and power cost.

This paper is organized as follows. After mentioning the related work in Section II, I describe crossbar-switch based stage-level reconfigurable CMP architecture and its area/power breakdown in Section III. In Section IV, I propose a new multiplexing hierarchical inter-connection structure. I show the experimental results in Section V and conclude the paper in the Section VI.

II. RELATED WORKS

For tolerating permanent faults, architectures must have the ability to reconfigure, where reconfiguration can refer to a variety of activities ranging from decommissioning non-functioning, non-critical processor structures to swapping in cold spare devices. In a reconfigurable architecture, recovery entails isolating defective module(s) and incorporating spare structures as needed. Support for reconfiguration can be achieved at various granularities, from ultrafine grain systems [7, 8] that have the ability to replace individual logic gates to coarser designs that focus on isolating entire processor cores [1, 2, 9-14, 21]. This choice presents a trade-off between complexity of implementation and potential lifetime enhancement [15, 16]. Stage level reconfiguration is positioned as a good candidate for system recovery as it scales well with the increase in area available for redundancy. Logically, stages are a convenient boundary because pipeline architectures divide work at the level of stages (e.g., fetch, decode, etc.). Similarly, in terms of circuit implementation, stages are an intuitive boundary because data signals typically get latched at the end of every pipeline stage. Both these factors are helpful when reconfiguration is desired with a minimum impact on the performance.

Recently, two kinds of stage-level reconfigurable CMP have been proposed. Gupta et al. [5] have developed StageNet (SN) that is a flexibility-oriented architecture and the pipeline stages are connected by cross-bar-switch based interconnection network. The cross-bar switches are the main components in the SN that provides distinct feature for stage-level reconfiguration. Even though the crossbar switches play an important role for high flexibility in stage-level reconfigurable CMP, it is evident that the crossbar switch interconnections are very area/power-critical resources in an entire system with the increased number of pipeline-stage or cores [17]. Furthermore, the SN has been proposed to withstand the rapidly increasing device failure rates expected in future technologies (32 nm and beyond). However, it also means that the crossbar switch-based reconfigurable CMP would spend much power caused by huge interconnect wires because wire power consumption become more and more significant with advances in fabrication technology [18].

Romanescu et al. [6] have proposed Core Cannibalization Architecture (CCA), that also exploits stage level reconfigurability. CCA is an area/power oriented architecture because it allows only a subset of pipelines to lend their stages to other broken pipelines, thereby avoiding full crossbar interconnection. Unlike StageNet, CCA pipelines maintain all feedback links and avoid any major changes to the micro-architecture. Although these design choices reduce the overall complexity, fewer opportunities of reconfiguration exist for the CCA as compared to the StageNet. In addition, their experimental results only show performance and area of CCA consisting of a few cores without evaluating power consumption.

III. PRELIMINARIES

In this section, I present an example (*StageNet*) of crossbar-switch based stage-level reconfigurable CMP. I have first designed a CMP similar to the *StageNet* as the base architecture and its area and power breakdown are shown in this section. This base architecture will be used for quantitative comparison with proposed cost-effective approach.

1. Crossbar-Switch based Stage-Level Reconfigurable CMP

StageNet (SN) architecture [5] has been positioned as a good candidate for system recovery as it scales well with the increase in area available for redundancy. Logically, stages are a convenient boundary because pipeline architectures divide work at the level of stages (e.g., fetch, decode, etc.). Similarly, in terms of circuit implementation, stages are an intuitive boundary because data signals typically get latched at the end of every pipeline stage. These factors are helpful when reconfiguration is desired with a minimum impact on performance. Stage level reconfiguration is to decouple the pipeline stages from each other with removing all direct point-to-point communication between the stages and replacing them by a switch based interconnection network. Fig. 1(a) shows an example SN composed of 5 cores.

The network is formed by replacing the direct connections at each pipeline stage boundary by a crossbar switch interconnection. Within this architecture, pipeline stages can be selected from the pool of available stages to act as logical processing cores. Fig. 1(b) shows a horizontal slice of stage-level reconfigurable CMP. The slice is a basic building block for the *SN* architecture. It consists of a decoupled pipeline micro-architecture that allows convenient reconfiguration at the granularity of

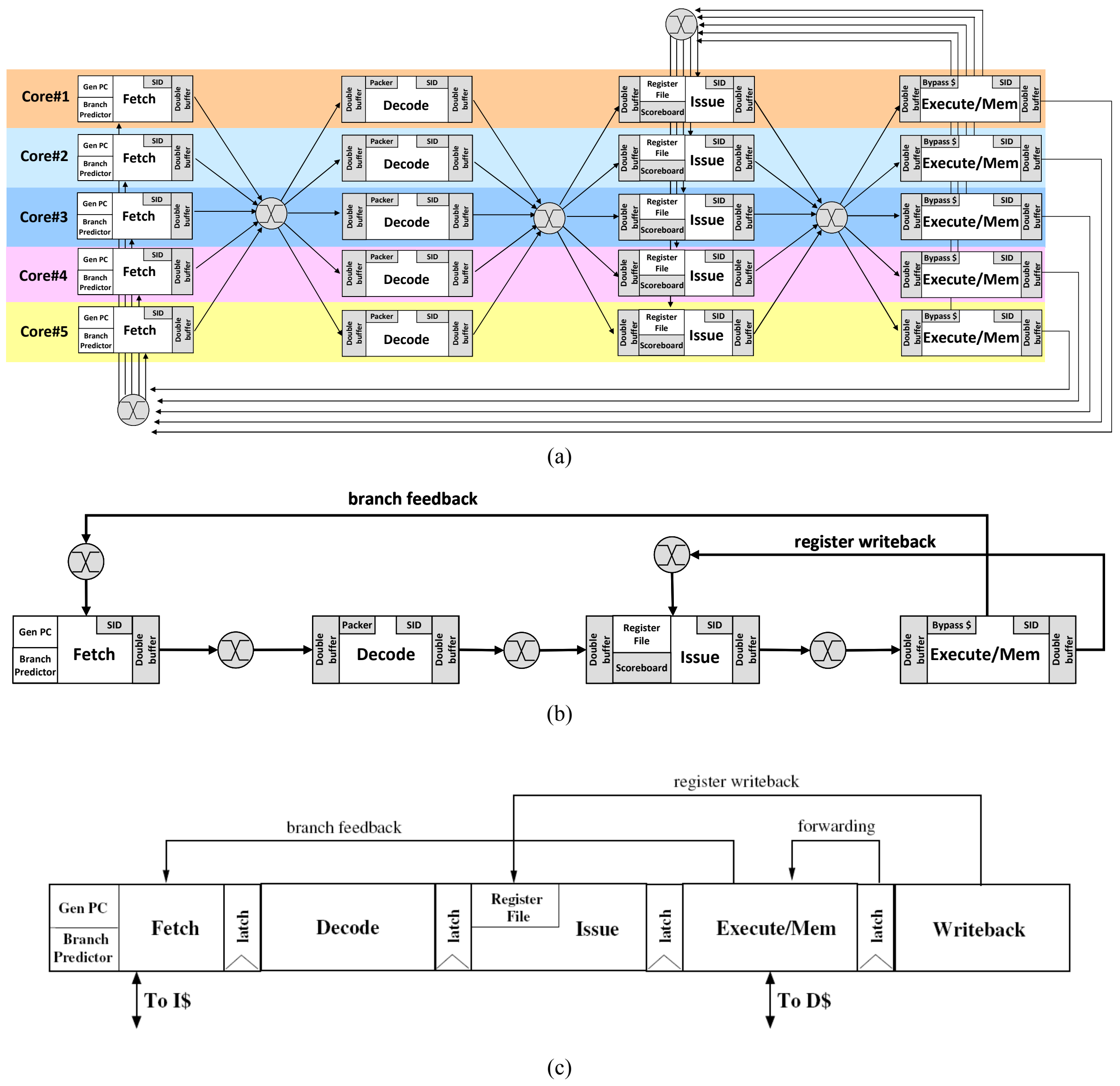


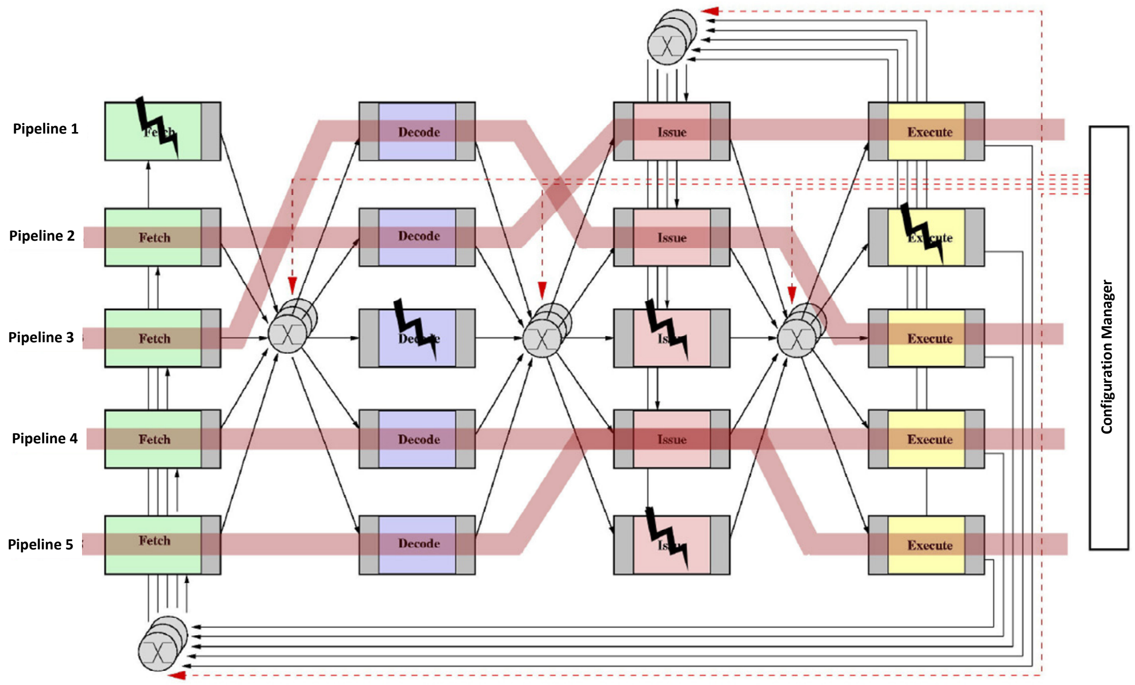
Fig. 1. Structure of *StageNet*. (a) An example of *StageNet* architecture composed of 5 cores, (b) A horizontal slice of *StageNet*, (c) A conventional 5-stage in-order pipeline [5].

stages. As a basis for the *SN*, a simple embedded processor core is used, consisting of five stages namely, fetch, decode, issue, execute/memory, and writeback as shown in Fig. 1(c). The conventional pipeline latches are replaced with a combination of a crossbar switch and buffers. To minimize the performance loss from inter-stage communications, full crossbar switches are used since these allow non-blocking access to all of their inputs. The full crossbar switches have a fixed channel width and, as a result, transfer of an instruction from one stage to the next can take a variable number of cycles. However, this channel width of the crossbar can be varied to tradeoff performance with area.

Fig. 2 shows an example when some faults occur in the *SN* as Fig. 1(a). The configuration manager is invoked for making route with other core and it is designed as a firmware/kernel module because resource allocation (routing) policy is more flexible in software. The top three slices have a minimum of two stages alive of each type, and, thus, two logical pipelines are formed. The logical pipelines are highlighted using the shaded path indicating the flow of the instruction streams. It is noteworthy that all top three pipelines in Fig. 2 have at least one failed stage, and therefore, a multi-core system in a similar situation would have lost all three pipelines. Hence, the *SN*'s ability to efficiently borrow stages from different slices, gives it the competitive edge over a traditional multi-core. In addition, if the stages can be time-multiplexed by multiple logical pipelines, then the same number of logical pipelines can be maintained. Fig. 2 has the bottom two slices sharing the issue stage. Therefore, the *SN* exploits the inherent redundancy present in a multi-core by borrowing/sharing stages from adjacent cores. As nodes (stages) wearout and eventually fail, the *SN* will exhibit a graceful degradation in performance, and a gradual decline in throughput.

2. Area and Power Breakdown of Crossbar-Switch based Stage-Level Reconfigurable CMP

Along with the benefits of the *SN* architecture as mentioned in the previous section, the *SN* architecture has certain area and power overheads associated with itself. Area and power overhead primarily arises from the crossbar-switch based interconnection network between the stages. To study the impact of the crossbar-switch based interconnections on entire power and area of *SN* architecture, I have implemented 4 cases of base

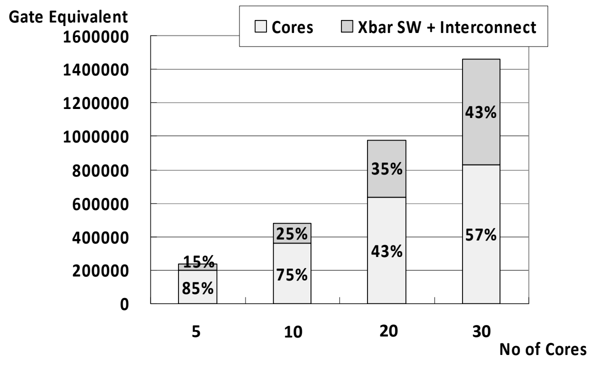


Number of faults: 5 Traditional CMP: 1 Stage-level Reconfigurable CMP: 4

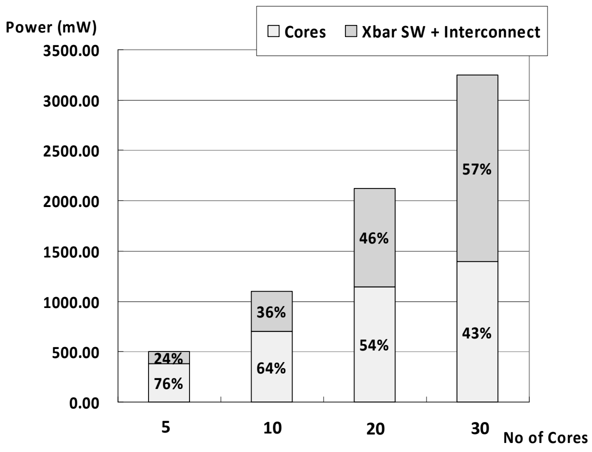
Fig. 2. An example scenario when 4 cores have faults in *StageNet* [5].

architecture similar to the *SN* with variation in the number of cores – 5, 10, 20, and 30 cores. 64-bit crossbar-switches and Leon2 processors [19] have been used for the base architecture implementation – Leon2 is a synthesizable VHDL model of a 32-bit processor compliant with the SPARC V8 architecture consisting of five pipeline stages. I have designed them at RT-level using VHDL and synthesized gate-level circuits from the VHDL descriptions to analyze area and power cost. The synthesis has been done using Design Compiler [20] with 90 nm technology.

Fig. 3(a) illustrates the impact of the crossbar-switch based interconnections varying the number of cores on area cost. In the case of architecture including 5 cores, the crossbar-switch based interconnection network only occupied about 15% of the total area. However, the architecture consisting of 30 cores shows 43% area of the crossbar-switch based interconnection network. In addition, Fig. 3(b) shows the power overhead with



(a)



(b)

Fig. 3. Area and power breakdown with increasing number of cores. (a) Area, (b) Power.

increasing the number of cores. Even though the architecture including 5 cores only shows 24% power consumption[1] by the crossbar- switch based interconnections, the interconnection network spends about 57% of the total power consumed in the architecture consisting of 30 cores. Based on the area/power breakdown, it is evident that the crossbar-witch based interconnection network is very area/power-critical resources in an entire system with the increased number of cores. Therefore, reducing power consumption caused by the crossbar- switch based interconnections a serious concern for reliability of the stage-level reconfigurable CMP.

IV. MOTIVATION

As mentioned in the previous section, crossbar-switch based stage-level reconfigurable CMP is a highly reconfigurable and adaptable multi-core computing substrate. The crossbar-switch can guarantee any combination of one-to-one connection between the inputs and outputs. Fig. 4 shows routing capability of the crossbar-switch between pipeline stages in the same core. The switch can make route for incoming/outgoing instruction stream between two cores as well as pass the pipeline stream from the previous stage to the next stage. Therefore, number of incoming/outgoing instruction stream from/to other core is 1/1 in Fig. 4. However, stage-level reconfiguration is only necessary when some pipeline stages are broken. It means that routes of the instruction streams are dependent on stage-level fault occurrence.

For study about dependency between stage-level reconfiguration and fault occurrence, I classify all of the fault occurrence cases between two stages as shown in Fig. 5. Fig. 5(a)[2] shows that no fault occurs in two stages. Even though the crossbar-switch has capability to make route with other cores, it is sufficient to only pass the pipeline stream from the previous stage to the next

[1] The synthesis results with 90 nm technology show microwatt –level static power dissipation whereas dynamic power varies from 200 mW to 3500 mW. It means that dynamic power is most of the entire power. Therefore, I have only analyzed dynamic power dissipation in this paper.

[2] Number of incoming from other core can be 1 if the *SN* architecture supports 'stage sharing'. However, 'stage sharing' is optional technique of the *SN* architecture and it only shows very trivial improvement [5]. Therefore, I do not consider stage sharing for the *SN* operation.

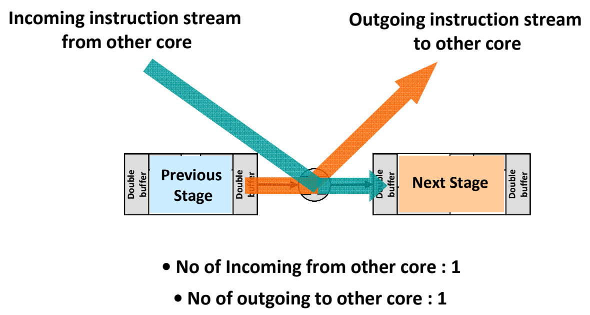

Incoming instruction stream from other core
Outgoing instruction stream to other core
Double buffer
Previous Stage
Double buffer
Double buffer
Next Stage
Double buffer
• No of Incoming from other core : 1
• No of outgoing to other core : 1

Fig. 4. Routing capability of crossbar-switch between two pipeline stages in the same core.

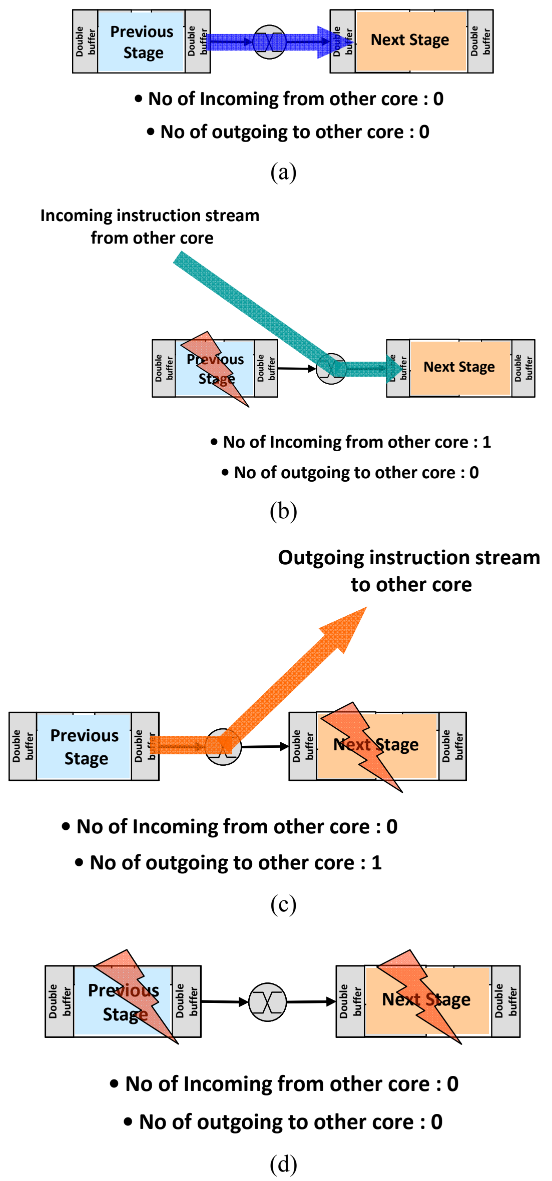

Double buffer
Previous Stage
Double buffer
Double buffer
Next Stage
Double buffer
• No of Incoming from other core : 0
• No of outgoing to other core : 0
(a)
Incoming instruction stream from other core
Previous Stage
Next Stage
• No of Incoming from other core : 1
• No of outgoing to other core : 0
(b)
Outgoing instruction stream to other core
Previous Stage
Next Stage
• No of Incoming from other core : 0
• No of outgoing to other core : 1
(c)
Previous Stage
Next Stage
• No of Incoming from other core : 0
• No of outgoing to other core : 0
(d)

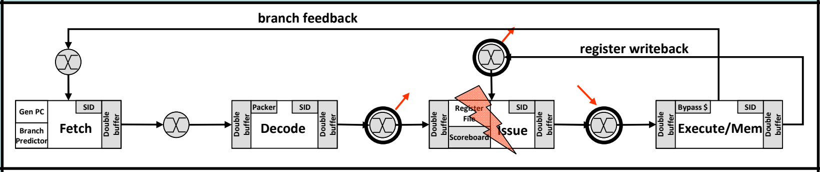
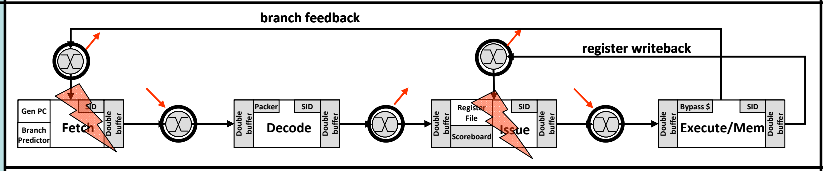
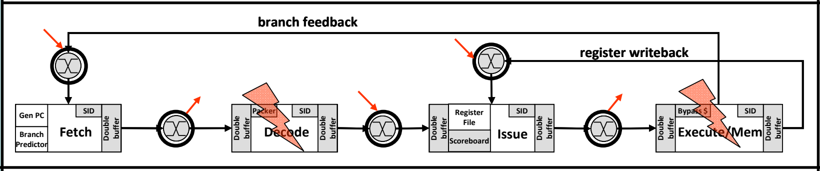
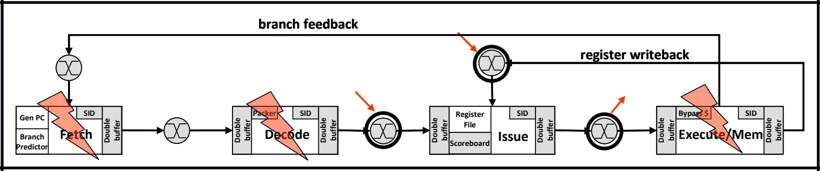
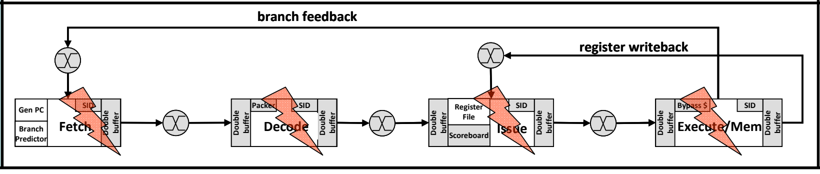
Fig. 5. Dependency between route of instruction stream and stage-level fault occurrence. (a) No fault, (b) A fault in the previous stage, (c) A fault in the next stage, (d) two faults in the both stages.

stage. Therefore, the number of incoming/outgoing instruction stream from/to other core is 0/0 in Fig. 5(a). In the case of Fig. 5(b), the previous stage has a fault and the switch selects an incoming instruction stream from the other core and sends it to the next stage - the number of incoming/outgoing instruction stream from/to other core is 1/0. In the similar manner as Fig. 5(b), Fig. 5(c) shows that the number of incoming/outgoing instruction stream from/to other core is 0/1. Finally, Fig. 5(d) shows two faults occur in the both stage. In this case, the crossbar-switch is not utilized for passing or routing any instruction stream. Based on the observation of Fig. 5, the crossbar switches only routes either incoming pipeline flow or outgoing pipeline flow under fault occurrence even though the crossbar switches have functionality to route the both cases as shown in Fig. 4. It means that crossbar switches show redundancy and it gives a hint to reduce power/area of the interconnection structure keeping flexibility and performance.

V. HIERARCHICAL MULTIPLEXING INTERCONNECTION STRUCTURE

Based on Fig. 5, I can extend the observation to an entire core in the crossbar-switch based architecture to find how many incoming/outgoing instruction streams from/to other core are required for a core. Fig. 6 shows classification of stage-level fault-occurrence in a core as Fig. 1(b). Each row shows fault-occurrence case having the maximum number of incoming/outgoing instruction streams under the number of faults (1~4). Synthetically, the maximum incoming/outgoing instruction streams occur when the stage-level faults occur in the interleaved manner – number of faults is 2 and the maximum number of incoming/outgoing instruction streams is 3/3.

Therefore, instead of adopting the crossbar switch, it is possible to implement the architecture by hierarchical multiplexing as shown in Fig. 7. It shows interconnection structure between cores. A core is connected to all of the other cores through Level#1 multiplexers. Bit-width of an interconnection between two cores is three times the crossbar channel with of the original architecture because the maximum number of incoming/outgoing instruction streams is 3/3. The number of inputs on a L2 multiplexer is four – one passing instruction stream from the previous stage in the same core and the maximum three-incoming

No of Faults	Situations	*No' of Incoming	**No' of Outgoing
1		1	2
2		2	3
		3	2
3		2	1
4		0	0

Symbol	Meaning	Symbol	Meaning
⚡	Fault occurrence in the stage	⊗	Crossbar-switch routing instruction stream from/to other core
↘	Incoming instruction stream from other core	○	Crossbar-switch passing instruction stream in the same core or not used
↗	Outgoing instruction stream to other core		

*No' of Incoming: it means number of incoming instruction stream from other cores
**No' of Outgoing: it means number of outgoing instruction stream to other cores

Fig. 6. Classification of stage-level fault occurrence in a core.

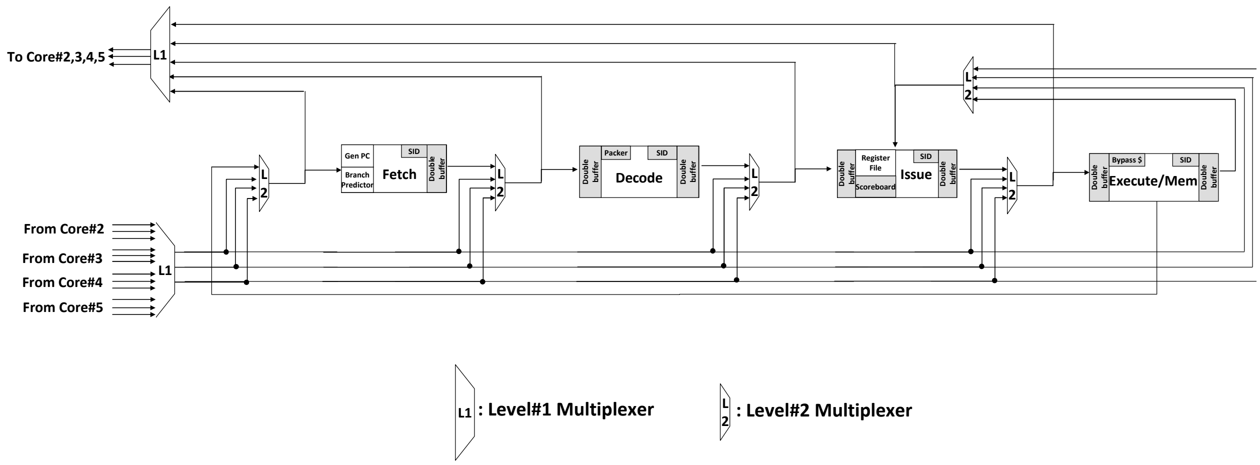


Fig. 7. Hierarchical multiplexing interconnection structure.

instruction streams from the other cores. Such a hierarchical multiplexing interconnection structure guarantees the same functionality and performance of the crossbar-switch based architecture with removing redundant functionality of the crossbar switch.

VI. Experiments

1. Experimental Setup

To demonstrate the quantitative effectiveness of the hierarchical multiplexing interconnection structure, I have implemented 4 cases of CMP architecture including the proposed interconnection structure with variation in the number of cores – 5, 10, 20, and 30 cores[3]. I have designed them at RT-level using VHDL and synthesized gate-level circuits from the VHDL descriptions to analyze hardware cost. The synthesis has been done using Design Compiler [20] with 90 nm technology and Design Compiler basically reports area, power and delay cost after gate-level synthesis.

2. Results

A. Area evaluation

Fig. 8 shows area cost evaluation varying the number of cores. 'Base' means CMP consisting of crossbar-switch interconnection network and 'Proposed' means CMP with the proposed interconnection structure. First of all, area cost of only the proposed interconnection structure is compared with the crossbar-switch interconnection network in Fig. 8(a). In the case of CMP including 5 cores, area reduction ratio is only 9%. However, the area cost of the proposed interconnection structure connecting 10~30 cores is reduced by 21%~48%. In the aspect of entire area, the proposed approach can reduce area by 1%~21% increasing the number of cores from 5 to 30 as shown in Fig. 8(b). Therefore, the proposed approach can reduce much area cost when the number of core increases.

B. Power evaluation

Fig. 9 shows power evaluation varying the number of cores in the similar manner as the area evaluation. Fig. 9 (a) shows the power comparison between the proposed interconnection structure and the crossbar-switch interconnection network. In the case of CMP including 5 cores, power reduction ratio is only 11%. However, the power consumption of the proposed interconnection structure connecting 10~30 cores is reduced by 26%~56%. In the aspect of overall power consumption, the proposed approach can reduce power by 3%~32% increasing the

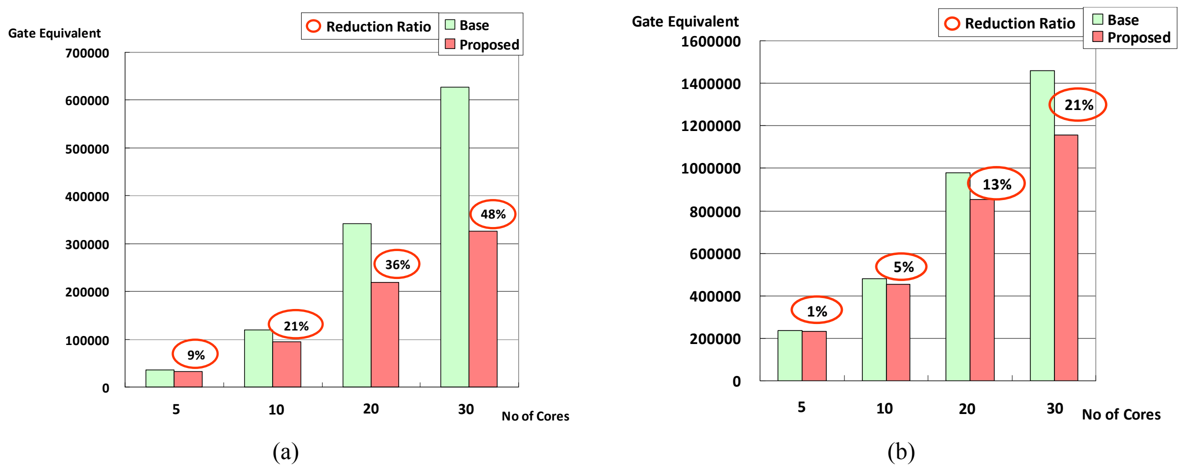
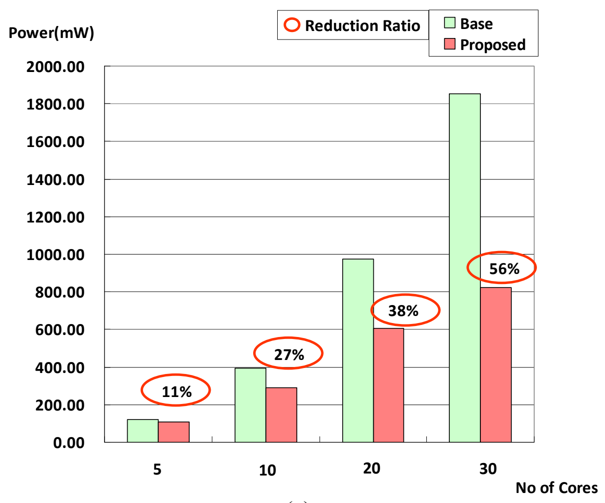
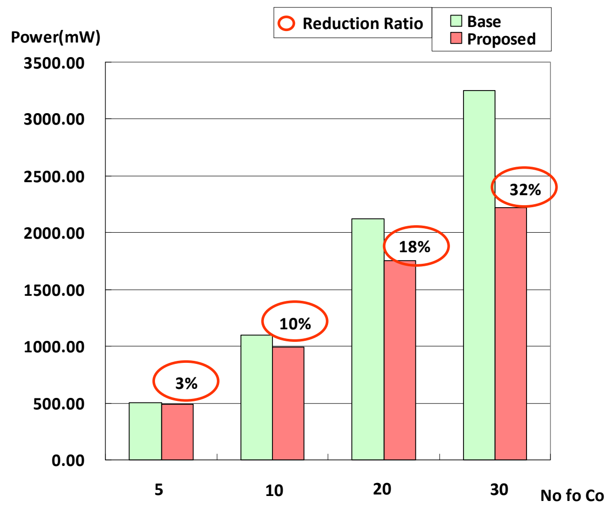


Fig. 8. Area comparison. (a) Interconnections among pipeline stages, (b) Entire CMP.

[3] The RTL architectures have been implemented without 'configuration manager' as shown in Fig. 2 because this module is not necessary for quantitative evaluation of the hierarchical multiplexing interconnection structure.



(a)



(b)

Fig. 9. Power comparison. (a) Interconnections among pipeline stages, (b) Entire CMP.

number of cores from 5 to 30 as shown in Fig. 9(b). Therefore, the proposed approach can reduce much power as well as area when the number of core increases.

C. Performance evaluation

The synthesis results show that the 4 base architectures and 4 proposed architectures have the same critical path delay, 6.37 ns. This is because the critical path is found in the 'EX/MEM' stage. It indicates the proposed approach does not cause performance degradation in terms of the critical path delay. In addition, execution cycle count of the proposed architecture can not vary from the base architecture because the functionality and flexibility of the proposed architecture is same as the base architecture. It also indicates the proposed approach does not come by performance degradation in terms of the execution cycle count.

3. Limitation of the Proposed Architecture

As shown in the experimental results, the proposed architecture is able to keep the reliability offered by the crossbar-switch while reducing the area and power overheads. Even though the proposed architecture does not support 'stage sharing' that is not mandatory technique for the SN architecture, the stage sharing can be essential to achieve high resource utilization for some cases. In general, if the number of alive *n* th stages is larger than that of *n*+1 th stages, the proposed architecture cannot fully utilize the alive *n*-th stages.

VII. Conclusions

Technology trends project high failure rates for future CMOS technology generations indicating a growing need for revolutionary new designs that can maintain functionality in the presence of multiple failed components. To satisfy such a need for fault tolerant design, stage-level Reconfigurable CMP architectures have been proposed for tolerating high failure rates. The advantage of being reconfigurable at a finer granularity than a processor core results in a longer lifetime for the system.

However, existing stage-level reconfigurable CMPs show trade-off between flexibility and area/power. To overcome such a limitation, I propose a novel interconnection structure for stage-level reconfigurable CMP. It has been shown the new interconnection structure is derived from the observation of the fault occurrence in the crossbar-switch based architecture. The proposed interconnection structure efficiently removes the redundancy of the crossbar-switch interconnection network by hierarchical multiplexing with reducing area and power. Experimental results show that the proposed approaches save area and power compared to conventional crossbar- switch based architecture without performance degradation. Implementation of the proposed structure demonstrates improving results. The area reduction up to 21% and the power savings up to 32% are evident when compared with the base architecture including crossbar-switch based interconnection network for 30 core CMP with a promise to reduce further overheads in large size CMPs.

ACKNOWLEDGMENTS

This Research was supported by the Sookmyung Women's University Research Grants 1-1103-0729.

REFERENCES

[1] N. Aggarwal, P. Ranganathan, N. P. Jouppi, and J. E. Smith, "Configurable isolation: building high availability systems with commodity multi-core processors," in *Proc. of the 34th Annual International Symposium on Computer Architecture*, pp.470-481, Jun., 2007.

[2] D. Sylvester, D. Blaauw, and E. Karl, "Elastic: An adaptive self-healing architecture for unpredictable silicon," *IEEE Journal of Design and Test*, Vol.23, No.6, pp.484-490, Dec., 2006.

[3] K. Bernstein, "Nano-meter scale cmos devices," in *Proc. of the International Symoposium on Quality Electronic Design*, p.7, Mar., 2004.

[4] S. Borkar, "Designing reliable systems from unreliable components: The challenges of transistor variability and degradation," *IEEE Micro*, Vol.25, No.6, pp.10-16, Nov., 2005.

[5] Shantanu Gupta, Shunguang Feng, Amin Ansari, Jason Blome, and Scott Mahlke, "The StageNet Fabric for Constructing Resilient Mulitcore Systems," in *Proc. of the 41st Annual International Symposium on Microarchitecture*, pp.141-151, Nov., 2008.

[6] B. F. Romanescu and D. J. Sorin, "Core cannibalization architecture: Improving lifetime chip performance for multicore processor in the presence of hard faults," in *Proc. of the 17th International Conference on Parallel Architectures and Compilation Techniques*, pp.43-51, Oct., 2008.

[7] F. A. Bower, P. G. Shealy, S. Ozev, and D. J. Sorin, "Tolerating hard faults in microprocessor array structures," in *Proc. of the 2004 International Conference on Dependable Systems and Networks*, p.51, Jun., 2004.

[8] K. Constantinides, S. Plaza, J. Blome, B. Zhang, V. Bertacco, S. Mahlke, T. Austin, and M. Orshansky, "Bulletproof: A defect-tolerant CMP switch architecture," in *Proc. of the 12th International Symposium on High-Performance Computer Architecture*, pp.3-14, Feb., 2006.

[9] R. Amerson, R. J. Carter, W. B. Culbertson, P. Kuekes, and G. Snider, "Teramac – configurable custom computing," in *Proc. of the 1995 International Symposium on FPGA's for Custom Computing Machines*, pp.32-38, Apr., 1995.

[10] J. C. Smolens, B. T. Gold, B. Falsafi, and J. C. Hoe, "Reunion: Complexity-effective multicore redundancy," in *Proc. of the 39th Annual International Symposium on Microarchitecture*, pp.223-234, Nov., 2006.

[11] W. Bartlett and L. Spainhower, "Commercial fault tolerance: A tale of two systems", *IEEE Transactions on Dependable and Secure Computing*, Vol.1, No.1, pp.87-96, Jan., 2004.

[12] D. Bernick, B. Bruckert, P. D. Vigna, D. Garcia, R. Jardine, J. Klecka, and J. Smullen, "Nonstop advanced architecture," in *Proc. of International Conference on Dependable Systems and Networks*, pp.12-21, Jun., 2005.

[13] W. Culbertson, R. Amerson, R. Carter, P. Kuekes, and G. Snider, "Defect tolerance on the teramac custom computer," in *Proc. of the 1997 International Symposium on FPGA's for Custom Computing Machines*, pp.116-123, Apr., 1997.

[14] L. Spainhower and T. Gregg, "IBM S/390 Parallel Enterprise Server G5 Fault Tolerance: A Historical Perspective," *IBM Journal of Research and Development*, Vol.43, No.5, pp.863-873, Sep., 1999.

[15] K. Constantinides, S. Plaza, J. Blome, B. Zhang, V. Bertacco, S. Mahlke, T. Austin, and M. Orshansky. "Bulletproof: A defect-tolerant CMP switch architecture," in *Proc. of the 12th International Symposium on High-Performance Computer Architecture*, pp.3-14, Feb., 2006.

[16] P. Shivakumar, S. Keckler, C. Moore, and D. Burger, "Exploiting microarchitectural redundancy for defect tolerance," in *Proc. of the 2003 International Conference on Computer Design*, p.481, Oct., 2003.

[17] Terry Tao Ye, Luca Benini, and Giovanni De Micheli, "Analysis of Power Consumption on Switch Fabrics in Network Routers," in *Proc. of Design Automation Conference*, pp.524-529, Jun., 2002.

[18] N. P. Carter and A. Hussain, "Modeling wire delay, area, power, and performance in a simulation infrastructure," *IBM Journal of Research Development*, Vol.50, No.3, pp.311-319, Mar., 2006.

[19] Gaisler Research; http://www.gaisler.com/cms

[20] Synopsys Corp. : http://www.synopsys.com

[21] Di Wu, Imyong Lee, Junwhan Ahan, and Kiyoung Choi, "Fast Generation of Multiple Custom Instructions under Area Constraints," *Journal of Semiconductor Technology and Science*, Vol.11, No.1, pp.51-58, Mar., 2011.

Yoonjin Kim received the B.S. degree in information and communication engineering from Sungkyunkwan University, Seoul, South Korea, in 2003, the M.S. degree in electrical engineering and computer science from Seoul National University, Seoul, South Korea, in 2005, and the Ph.D. degree in computer engineering from Texas A&M University, College Station, in 2009. From 2009 to 2010, he was a Senior R&D Staff Member with the Samsung Advanced Institute of Technology (SAIT), Gyeonggi, South Korea. Since 2010, he has been an Assistant Professor with the Department of Computer Science at Sookmyung Women's University in Seoul, South Korea. His research interests include embedded systems, computer architecture, VLSI/system-on-chip design, and hardware/software co-design.