Implementation and Evaluation of a Superscalar Processor Based on Dynamic Adaptive Redundant Architecture

Ryoji Watanabe, Jun Yao, Hajime Shimada
Graduate School of Information Science,
Nara Institute of Science and Technology
Email: {ryoji-w, yaojun, shimada}@is.naist.jp

Kazutoshi Kobayashi
Graduate School of Science and Technology,
Kyoto Institute of Technology
Email: kobayasi@kit.ac.jp

I. Introduction

Recently, technology trends have been leading to increasing transient and permanent faults. To address the problem, space redundancy such as dual modular redundancy (DMR) and triple modular redundancy (TMR) are commonly adopted for fault detection and correction. As introduced in paper [1], using a fine-grained space redundancy named Dynamic Adaptive Redundant Architecture (DARA), only multiple faults under a single cycle may have the possibility to cause a real failure.

However, there are also circumstances in which reliability is not firstly prioritized. As an example, major parts of the media processing algorithms can tolerate a small error rate in some frames while they usually lack of processing power to meet sufficient quality of service. For this concern, we propose a performance improving add-on to exploit parallelism by making full use of the existing duplicated resources in DARA.

II. Dynamic Adaptive Redundant Architecture.

We designed our performance enhancing scheme on an implementation of a space redundancy based processor called Dynamic Adaptive Redundant Architecture (DARA) [1]. Basically, DMR mode is the major working mode for soft error resilient execution in DARA, with error detection and proper re-execution schemes. TMR mode can thereby be used only for the diagnostics of system health under the awareness of possible permanent failures. After addressing the permanently defected module, the processor can fall back to the DMR mode, using valid pipeline modules.

As shown in Figure 1, DARA employs a stage-level recovery scheme by connecting pipeline modules together. The error detection logic is augmented in each pipeline module and it detects the error by comparing pipeline register outputs of local and neighbor pipelines. DARA utilizes a uni-direction link (as the dashed lines in Fig. 1) between pipelines to pass the comparison data, which may also be used for the performance enhancing purpose in this paper.

III. Superscalar Processor Based On DARA

In this section, we propose an incremental mode for DARA which can alternatively achieve better performance by using the existing duplicated hardware for fault toleration. Switching from DMR, the key idea is to make the pipeline modules cooperate with each other and thus construct a two-way issue superscalar processor. Core Fusion [2] and Composable Lightweight Processors [3] are similar to our approach.

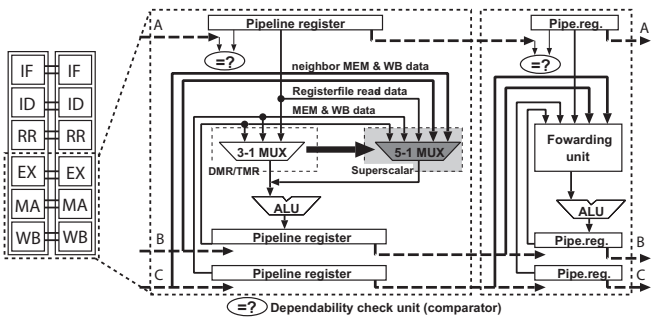

Fig. 1. Superscalar Processor base on DARA

TABLE I Area estimation.

Pipeline organization	Area estimation (in NAND2)				
	Logic	Latch	Total		
Base	50586	8581	59167	100.0%	-
DARA	53326	8605	61931	104.7%	100.0%
+Superscalar	64065	8670	72735	122.9%	117.4%

Fig. 1 gives an overview of the superscalar processor based on DARA. For simplicity, only the EX stage is shown in this figure. One advantage of this incremental two-way mode is that it does not require a large hardware extension. As shown in this figure, most resources like pipeline registers and ALUs can be used for superscalar execution. Moreover, the interconnections between pipelines that were originally used for transferring data to check reliability can now be employed to bypass forwarding data between pipelines. Comparators specially for error detection in DMR can be reused under superscalar mode as dependence check logic. Hardware augmentation specifically required in superscalar mode is mainly in the control logics like bypassing selection units. As an example, bold lines and gray-colored MUXs are newly employed for the forwarding data selection. In addition, read/write ports in the register file will be doubled to support superscalar executions.

IV. Evaluations

We evaluated the area increase to additionally support superscalar by an RTL implementation described in Verilog-HDL. The implementation is synthesized by Synopsys Design Compiler under a $0.18\mu m$ cell library. Table I lists the synthesized results of some pipeline module configurations. The first configuration is a single pipeline without any reliability and performance enhancement supports, as base line. The second configuration includes adaptive space redundancy mode which is for reliability. The third configuration represents the proposal in this paper which incrementally implements superscalar mode that can achieve a high performance.

The evaluation results with benchmarks “qsort” and “8queen” show that the superscalar mode improves performance by 8.1% and 28.9%, respectively. As shown in Table I, a 17.4% area increase is required to implement superscalar mode, as compared to DARA. This increase will be alleviated by including caches.

Acknowledgment

This work is supported by the VLSI Design and Education Center (VDEC), University of Tokyo with the collaboration of the Synopsys Corporation, and JST CREST.

References

[1] Jun Yao, et al., “A Scalable Pipeline Design for Modularizing High Dependable Framework via Spatial Redundancy,” DA symposium 2008, pp. 169-174, 2008.
[2] Engin Ipek, et al. “Core Fusion: Accommodating Software Diversity in Chip Multiprocessors,” ISCA-34, pp. 186-197, 2007.
[3] Changkyu Kim, et al. “Composable Lightweight Processors,” MICRO-40, pp. 381-393, 2007.