DEPENDABLE ADAPTIVE COMPUTING SYSTEMS
THE STANFORD CRC ROAR PROJECT

Subhasish Mitra, Wei-Je Huang, Nirmal R. Saxena, Shu-Yi Yu and Edward J. McCluskey
Center For Reliable Computing
Stanford University (http://crc.stanford.edu)

Abstract

We describe architectures and concurrent error detection, fault-location and recovery techniques for designing reconfigurable systems with high availability, data integrity, and protection from temporary, permanent and common-mode failures. These systems can also be used for unmanned remote applications.

1. Introduction

This paper describes dependable architectures for adaptive computing systems (ACS) comprising microprocessor, memory and reconfigurable logic (e.g., Field Programmable Gate Arrays or FPGAs) that have been developed in the DARPA sponsored ROAR project at Stanford Center for Reliable Computing (CRC). ROAR is an acronym for Reliability Obtained by Adaptive Reconfiguration. The main idea is to develop: (1) Concurrent Error Detection (CED) techniques to detect errors while the ACS is in operation; (2) fault-location techniques to identify the defective part (e.g., the defective logic block or routing resource); and (3) recovery techniques to reconfigure the system to operate without using the faulty part. Unlike conventional fault-tolerant systems where the Field Replaceable Unit (FRU) is a chip or a board, the FRU for an ACS is a logic block or a routing resource. Our work demonstrates that it is feasible to design low-cost but very effective dependable ACS. While the major thrust of this work is on ACS based on FPGAs, our techniques are applicable for other commercial reconfigurable hardware.

2. Concurrent Error Detection, Fault Location and Recovery Techniques

In this section, we describe concurrent error detection, fault-location and recovery techniques developed by us for applications mapped on the reconfigurable hardware (e.g., FPGAs).

2.1. Concurrent Error Detection (CED)

The CED techniques studied in our project are diverse duplication [Mitra 99, Mitra 00b, Mitra 00c, Mitra 00d], parity prediction [Touba 97, Zeng 99], multi-threading [Yu 00] and inverse comparison [Huang 00a].

Hardware duplication is a simple CED technique where two implementations of the same logic function are used and their outputs are compared — an error is reported when a mismatch occurs. Duplication guarantees data integrity in the presence of a single failure. However, data integrity is not guaranteed in the presence of multiple failures and *Common-Mode Failures* (CMFs). CMFs result from failures affecting both implementations of a duplex at the same time, due to a common cause [Mitra 00a]. Design diversity was proposed in [Avizienis 84] to protect redundant computing systems against common-mode failures. The idea of design diversity is to use two "different" implementations of the same logic function during duplication so that, common failure modes create different effects in the two implementations. However, the concept of design diversity as described in [Avizienis 84] was qualitative. We have developed a metric to quantify [Mitra 99] that quantifies diversity. Using this metric, we developed techniques for designing systems with CED based on diverse duplication [Mitra 00b, Mitra 00c]. The superiority of diverse duplication over other CED techniques like parity prediction was demonstrated in [Mitra 00c].

2.2. Fault Location

Several techniques can be used for locating the faulty logic block or interconnect in an FPGA [Stroud 97, Mitra 98, Das 99]. Specific features, the column readout feature of Virtex FPGAs for example, can also be utilized for fault-location [Huang 00b].

2.3. Recovery from Temporary Failures

Temporary failures can be caused due to intermittent or transient failures. For temporary failures in the configuration bits of a reconfigurable system, a reload of the contents from a safe storage is sufficient. The online partial reconfiguration feature of FPGAs can be used to read back and modify the data in configuration memory cells of FPGAs without stopping the system operation. However, some of the data in configuration frames of FPGAs are memory contents in user applications and a simple reload can cause memory coherence problems. A solution to this problem using "dirty bits" has been developed in [Huang 00b]. For recovery from temporary faults that do not cause errors in FPGA configuration, rollback techniques [Huang 00a] can be used.

2.4. Recovery from Permanent Failures

To repair a reconfigurable system with permanent faults, our approach is to use column-based pre-compiled configurations [Huang 00c]. The basic idea is to generate different FPGA configuration versions during the design phase. In each configuration version, a column of the FPGA is intentionally unused so that the corresponding configuration can tolerate permanent faults in the unused column. Since the different configuration versions are generated during the design phase, the system can switch between different configurations rapidly once the faulty FPGA column is located. For

minimizing the storage overhead of the different versions, data compression techniques can be used.

3. Dependable ACS Architectures

3.1. Multi-threaded ACS (MT-ACS) Architecture

The first ACS architecture is called the MT-ACS architecture (Fig. 3.1). The MT-ACS architecture is an extension of the conventional ACS architecture and contains a multi-threaded processor, memory, I/O devices and a reconfigurable coprocessor connected to the bus. For a given application, a part of the application (determined by the compiler) will be executed on the processor and the remaining part will be implemented on the reconfigurable coprocessor. For the portions of the applications mapped on the reconfigurable hardware, CED, fault-location and recovery techniques described in Sec. 2 can be used. Multi-threading can be used to implement fault-tolerance in the processor. Fault-tolerance is accomplished by using multiple threads of computations and algorithms [Saxena 00]. With the MT-ACS architecture, recovery from permanent faults in the processor requires board swapping or replacement of the microprocessor chip (unless special recovery structures using standby spares are used). Thus, the MT-ACS architecture needs human intervention for recovery and may not be suitable for unmanned applications (e.g., remote space exploration). The Dual FPGA architecture described in Sec. 3.2 addresses this problem.

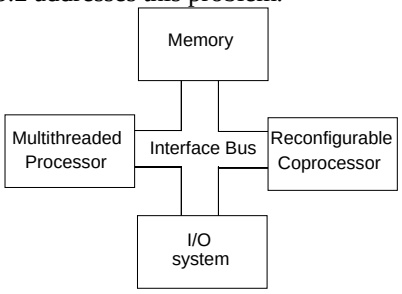

Figure 3.1. MT-ACS architecture

3.2. Dual FPGA ACS Architecture

The Dual FPGA ACS architecture is shown in Fig. 3.2. Each FPGA in Fig. 3.2 is configured to run certain user applications with some CED schemes. For example, a microprocessor with various CED features can be implemented on one of the FPGAs. We are currently designing such a "self-healing soft microprocessor" with built-in CED and autonomous recovery techniques. In addition, the controller (e.g., an 8051 micro-controller) on each FPGA monitors the error signal from the CED schemes implemented on the other FPGA and performs the configuration data recovery when necessary. Note that, the configuration frames of both FPGAs are stored in a safe memory space that is either replicated or protected by Error Correcting Codes (ECCs) to guarantee the correctness. All the CED, fault-location and recovery techniques described in Sec. 2 can be used for the Dual FPGA architecture.

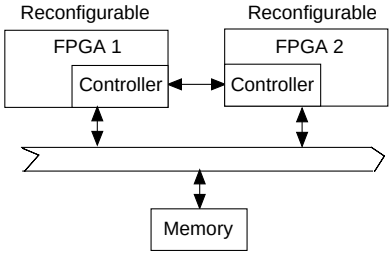

Figure 3.2. Dual FPGA ACS architecture

4. References

[Avizienis 84] Avizienis, A. and J. P. J. Kelly, "Fault Tolerance by Design Diversity: Concepts and Experiments," *IEEE Computer*, pp. 67-80, August 1984.

[Das 99] Das, D., and N.A. Touba, "A Low-Cost Approach for Detecting, Locating and Avoiding Interconnect Faults in FPGA-based Reconfigurable Systems," *Intl Conf. VLSI Design*, pp. 266-269, 1999.

[Huang 00a] Huang, W.-J., and E. J. McCluskey, "Analysis of Transient Error Effects in LZ Compression Algorithm and Rollback Error Recovery Schemes," *Proc. Pacific Rim Intl. Symp. Dependable Computing*, 2000.

[Huang 00b] Huang, W.-J., and E. J. McCluskey, "A Memory Coherence Technique for Online Transient Error Recovery of FPGA Configuration," *International Symp. FPGAs*, 2001.

[Huang 00c] Huang, W. J., "FPGA Fault Recovery by Reconfiguration," Reliability and Testability Seminar, 2000. (http://crc.stanford.edu/projects/roar/roarPres.html)

[Mitra 98] Mitra, S., P. P. Shirvani, E.J. McCluskey, "Fault Location in FPGA-Based Reconfigurable Systems," *High Level Design Validation and Test Workshop*, 1998.

[Mitra 99] Mitra, S., N. R. Saxena and E. J. McCluskey, "A Design Diversity Metric and Reliability Analysis for Redundant Systems," *Intl Test Conf.*, pp. 662-671, 1999.

[Mitra 00a] Mitra, S., N. R. Saxena, E. J. McCluskey, "Common-Mode Failures in Redundant VLSI Systems: A Survey," *IEEE Trans. Reliability*, 2000.

[Mitra 00b] Mitra, S. and E. J. McCluskey, "Combinational Logic Synthesis for Diversity in Duplex Systems," *Intl Test Conf.*, pp. 179-188, 2000.

[Mitra 00c] Mitra, S. and E. J. McCluskey, "Which Concurrent Error Detection Scheme To Choose?," *Proc. International Test Conf.*, pp. 985-994, 2000.

[Mitra 00d] Mitra, S., N. Saxena, E. J. McCluskey, "Fault Escapes in Duplex Systems," *VLSI Test Symp.*, pp. 453-458, 2000.

[Saxena 00] Saxena, N.R., *et al*., "Dependable Computing and On-line Testing in Adaptive Computing Systems," *Design and Test of Computers*, Vol. 17, No. 1, pp. 29-41, 2000.

[Stroud 97] Stroud, C., E. Lee and M. Abramovici, "BIST-Based Diagnostics of FPGA Logic Blocks," *Intl. Test Conf.*, pp. 539-547, 1997.

[Touba 97] Touba, N. A. and E. J. McCluskey, "Logic Synthesis of Multilevel Circuits with Concurrent Error Detection," *IEEE Trans. CAD*, pp. 783-789, July 1997.

[Yu 00] Yu, S-Y., N. Saxena, E. J. McCluskey, "ACS Implementation of a Robotic Controller Algorithm with Fault-Tolerant Capabilities," *FCCM*, 2000.

[Zeng 99] Zeng, C., N.R. Saxena, E.J. McCluskey, "Finite State Machine Synthesis with Concurrent Error Detection," *Intl Test Conf.*, pp. 672-679, 1999.