Error Compensation and Self-Repair for FPGA-based Processors

Farnoosh Hosseinzadeh, Heinrich Theodor Vierhaus
Computer Science Technical University of Brandenburg,
Cottbus, Germany
(hossefar, Heinrich.Vierhaus)@b-tu.de

***Abstract*—The successful usage of FPGAs in critical applications depends on reliability and hence mitigation methodologies are needed in order to handle and correct any kind of faults caused at the minimum cost in hardware and power. This paper presents an approach based on the combination of a diagnostic self-test in a processor for the detection and the mitigation of faults by means of two proposed designs: applying spare parts instead of corrupted parts in the design and partial reconfiguration of the corrupted portion of the circuit implemented in the FPGA. The later proposed design is able to reduce the utilization and hardware overhead.**

Keywords—FPGA; Fault tolerance design; Soft processor; spare functional blocks; Partial reconfiguration.

I. INTRODUCTION

With the reduction in feature size comes the added the occurring of burden of an increased probability of failure due to permanent and intermittent faults. Today's technologies get closer and closer to the physical limits and the nature of physics. It also is one of the main reasons why the new devices are sensitive to negative effects of various changes of the internal nanostructures and parameters. Every device has different error responses. In order to cope with the error, it is needed to understand the differences and design appropriately. There are two basic types of fault in digital circuits may occur:

1- Hard-type faults: These are functional faults due to physical defects, which caused by radiation (scarcely), or by aging and parameter deterioration as well as by over-stressing.
2- Soft-type faults: These are faults induced by radiation or by electromagnetic coupling effects. They are non-permanent.

Many researches have been done in fault tolerance of FPGA in against different faults. Several fault tolerance methods have been proposed in the past in order to mitigate the effects of SEUs in the configuration memory of FPGAs. These methods could be divided in two main categories.

The simplest approach, which is Reconfiguration-Based Technique, used to correct SEUs in the FPGAs' configuration memory, is known as Scrubbing, which consists of periodically rewriting the configuration memory [1]. Another technique is called Redundancy-based, which aims at masking the propagation of SEU's effects to circuit's outputs [2], [3], [4]. Fault masking is usually achieved through Triple Module Redundancy (TMR), where three identical replicas of the same circuit work in parallel and the outputs are compared and voted through a majority voter. These mitigation strategies offer tremendous improvements in reliability, but are often expensive in terms of FPGA resource utilization, power consumption, etc. [5], [6].

FPGAs in advanced technologies that have a minimum feature size of 20 nm and below are becoming available [7] and through continued innovation in circuit design and layout techniques, the intrinsic SEU FIT of the silicon with each new generation has been lowered, hence most application deployment without any additional SEU mitigation [8]. However, most of them suffer still from permanent faults. Some authors have performed investigations on performance degradation and aging effects on FPGAs, indicating that long-term dependable systems will need a functionality of parameter supervision [9-11]. Hence, there is a strong need to design fault-tolerant and long-term dependable systems, which can handle permanent faults at the minimum cost in hardware and power specifically on the technology platform provided by FPGAs.

II. OVERALL ARCHITECTURE OF THE SELF-REPAIRING PROCESSOR

The proposed work is targeted at managing the fault behavior of a soft core, making use of other external hard-or soft wired processors where needed. The basic architecture of a processor is shown in Fig.1. In order to handling permanents faults in a soft processor by FPGA, we provide two proposed designs: 1) Applying spare parts in the design 2) using partial re-configuration in the system.

A. Using spare functional blocks

The repair mechanism is accomplished by using an extra reconfiguration controller, implemented either in hardware or in software, running on an external processor (hard or soft) of the same FPGA. For a single component of the processor, specific configurations will be computed during design time that can be mapped in the same region of the FPGA.

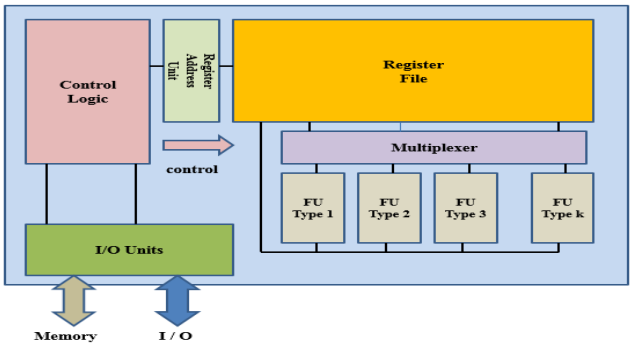

Fig.1.The basic architecture for a processor

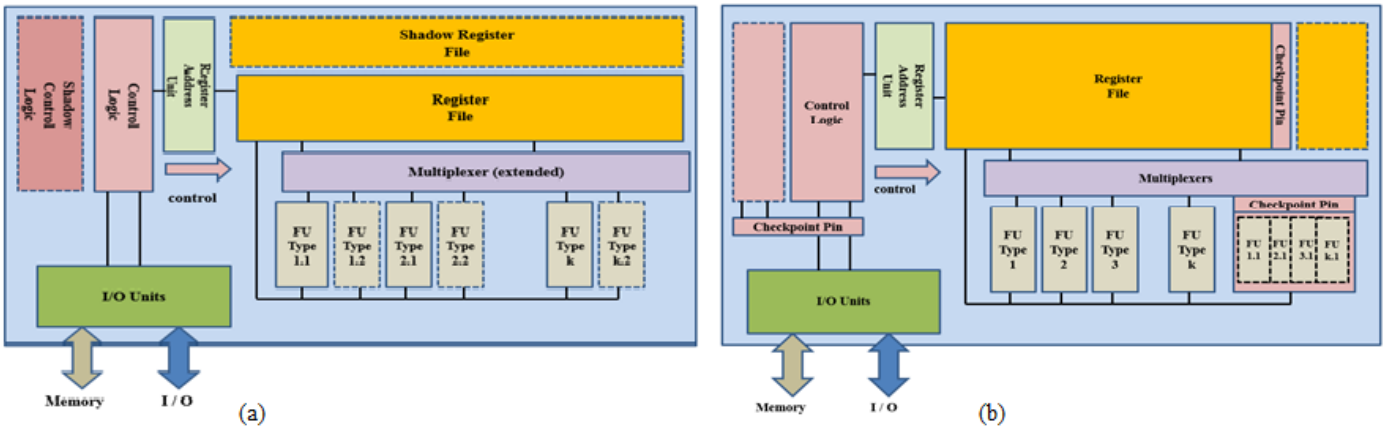

Fig.2. (a) The self-repairing processor by spare functional block. (b) The partial reconfiguration in the processor

Typically these alternative implementations will include "spare parts" (such as extra registers, ALUs, Shifters), which can be activated upon demand, once such a functional has been identified as "faulty" by a diagnostic self-test, which is done by ring oscillators [12].

In the first approach, "spare functional blocks" are synthesized from the beginning. They are used "upon demand" in case of permanent faults, using only extra switching functions for system re-organization, which is shown in Fig.2(a). This approach avoids in-system re-synthesis operations, but it will suffer from many resources that are allocated for back up, but may never be needed.

B. Applying partial reconfiguration technique

In another approach, state-of-the-art FPGAs allow for partial re-configuration "in system", which, in case of permanent faults, would allow the re-synthesis of just needed functional blocks from a general reserve of programmable devices (Fig.2 (b)). The Partial Reconfiguration allows for the dynamic change of modules within an active design [13]. The resynthesize process must be based on precomputed configurations that generate the needed functional blocks without requiring the non-available synthesis software. In the proposed design, a partial reconfiguration controller is designed. The dynamic partial reconfigurable modules are updated using not only the hardware triggers, but also user Command.

III. Conclution

Along with cost and performance, dependability is the third critical criterion based on which system-related decisions are made. Dependability evaluation is important because it helps identifying which aspect of the system behaviors, e.g. component reliability, fault coverage or maintenance strategy plays a critical role in determining overall system dependability. Moreover, the reliability of semiconductor devices and dependability of electronics equipment is one of the most discussed topics today. To achieve the targeted reliability, the errors should be recovered. The permanent error recoveries cause huge area and energy overhead. This paper expressed a significant contribution to the area of long-term reliable operation of logic circuits, especially special purpose processors, which are based on Field-Programmable Gate Arrays (FPGAs). Those proposed designs designed in VHDL and synthesized, and implemented using Xilinx Vivado. The results show that the design by partial reconfiguration reduces hardware overhead compared to the design by using spare parts.

Acknowledgment

This work is supported in part by German Academic Exchange Service (DAAD).

References

[1] C. Carmichael, "Triple Module Redundancy Design Techniques for Virtex FPGAs," Xilinx Application Notes, XAPP197, 2001.

[2] F. Lima Kanstensmidt, G. Neuberger, R. Hentschke, L. Carro, and R. Reis,"Designing Fault-Tolerant Techniques for SRAM-Based FPGAs", IEEE Design and Test of Computers, pp. 552-562, 2004.

[3] F. Lima, L. Carro, and R. Reis, "Designing Fault Tolerant System into SRAM-Based FPGAs," Proc. IEEE/ACM Design Automation Conf., pp. 650-655, June 2003.

[4] S. Habinc Gaisler Research, "Functional Triple Modular Redundancy (FTMR) VHDL Design Methodology for Redundancy in Combinational and Sequential Logic," www.gaisler.com, 2002.

[5] N. Rollins, M. Wirthlin, M. Caffrey, and P. Graham, "Evaluating TMR techniques in the presence of single event upsets", in Proceedings of the MAPLD Conference, September 2003.

[6] N. Rollins, M. Wirthlin, and P. Graham, "Evaluation of power costs in triplicated FPGA designs", in Proceedings of the MAPLD Conference, September 2004.

[7] "Introducing16nmUltraScale+FPGAs", 3D ICs, and MPSoCs, Xilinx Corporation, 2015.

[8] http://www.xilinx.com/support/quality/single-event-upsets.html

[9] Amouri, A.; Tahoori, M., "A Low-Cost Sensor for Aging and Late Transitions Detection in Modern FPGAs," Field Programmable Logic and Applications (FPL), 2011 pp.329,335, 5-7 Sept. 2011

[10] Kiamehr, S.; Amouri, A.; Tahoori, M.B., "Investigation of NBTI and PBTI induced aging in different LUT implementations, Field-Programmable Technology (FPT), 2011 International Conference on , vol., no., pp.1, 8, 12-14 Dec. 2011

[11] Pfeifer, P.; Pliva, Z., "On measurement of parameters of programmable microelectronic nanostructures under accelerating extreme conditions (Xilinx 28nm XC7Z020 Zynq FPGA), Field Programmable Logic and Applications (FPL), 2013 no., pp.1, 4, 2-4 Sept. 2013.

[12] Petr Pfeifer, Reliability Assessment and Advanced Measurements in Modern Nanoscale FPGAs, A summary of the Doctoral Dissertation thesis, 2014.

[13] Vivado Design Suite User Guide Partial Reconfiguration, UG909 (v2015.1) April 1, 2015.