CPE-322 Homework #5

Kevin Barresi, Nishant Panchal, Giancarlo Rico, Bryan Bonnet

March 24, 2014

Abstract

The issue of digital circuit degradation is certainly an important one; as devices become more complex and expensive, the cost of circuit failure is very high. A single stuck-on fault can render an entire complex system inoperable. To date, existing methods focus on detecting rather than repairing these faults. This document proposes a method and framework of fault identification and repair in consumer-oriented markets using a hardware evolution approach. This system results in highly robust and reliable digital circuitry. The targeted platform for initial proof of concept (PoC) demonstrations is Field Programmable Gate Array (FPGA) devices.

1 Division of Work

Table 1 below shows the groups contribution to this project. Each member incorporated an even amount of effort to this homework.

Name	**Per. Contribution**
Kevin Barresi	25%
Nishant Panchal	25%
Giancarlo Rico	25%
Bryan Bonnet	25%

Table 1: Group contribution distribution

Kevin Barresi completed the new information and introduction section.
Nishant Panchal completed the constraints and ethics section.
Giancarlo Rico completed the abstract and final documentation.
Bryan Bonnet completed the objective attributes section.

2 Summary of New Information Found

2.1 Introduction

The group presents an idea of circuit repair that takes advantage of all the issues mentioned above. The idea the group proposes is one where the analysis of the circuitry is performed online meaning that the device is completely operable during the reconfiguration phase. The groups method will ensure that no interruption or degradation of service is detectable in any manner. Our method will utilize intrinsic hardware evolution, where physical tests are run on the hardware block being reconfigured. The proposed method operates well for Single Event Upset (SEU) errors through partial FPGA scrubbing, along with permanent fault.

The remainder of this section is organized as follows. In subsection 2.2, we discuss prior research obtained from a multitude of industry sources. In subsection 2.3, we discuss the system configuration as a whole. In subsection 2.4, we detail each of the components the make up the system design. In subsection 2.5, we discuss potential algorithms for achieving our goals.

2.2 Prior Research

There are currently various methods of achieving fault resistance. Of these, notable solutions are: (a) locating and masking faults by circuit redundancy, (b) chipwise synthesis, (c) and triple-modular redundancy (TMR).

Fault Masking via Circuit Redundancy

In essence, redundancy involved building many circuit blocks of the same functionality onto the same FPGA, with the intent of only having one such circuit block operating at a time. If a fault is detected in one of these blocks, then one of the redundant blocks effectively takes over operation, with the failing block being completely bypassed by the rest of the device. Usually, redundant parts are built onto the device in columns and rows, though some redundancy architectures exist in which redundant routing resources are evenly distributed in the FPGA. This method of fault tolerance is transparent to FPGA users, and synthesis is very easy and can be used for all chips using the same FPGA implementation.

However, while this is perhaps the most basic solution to circuit faults, it is not without its disadvantages. Because of the large degree of redundancy, there is a high physical area overhead for the FPGA. Also, there is extra latency within the circuit due to the need to bypass any failing parts.

Chipwise Synthesis

Chipwise synthesis has been applied to circuits with high fault rates, especially with regards to implementations using nano-technology. In this methodm, each fault is located, and then placement and routing is customized for each chip so that faults are effectively worked around.

This solution, however, does not have any appropriate degree of scalability; it would not be suitable for production en masse of a single FPGA application. Also, testing costs are exceedingly high when the number of faults is large.

Triple-Modular Redundancy

Triple-modular redundancy is a specific form of circuit redundancy, in which three systems perform a process in parallel and the result is processed by a majority-voting system. This system produces a single output based on the output of each individual system. If any one of the three systems fails, then the other two systems can correct the fault and mask it. This voting system is implemented as a full adder, with its carry output acting as the vote output.

There are still weaknesses to this method. Foremost is the fact that this implementation relies on no more than one of the three systems failing at once; if two or more of them happen to fail, then TMR will produce an incorrect result. In addition, the majority-voting system itself could (with a low probability) be at fault. This can be addressed by implementing a TMR system for the first TMR system, however, the problem does still arise should the secondary system also fail.

Evolutionary Approaches

Prior art investigating evolution-based self-healing circuitry has fallen short on several fronts. One attempt demonstrated that an evolutionary approach was feasible, but the methods used are impractical in field use. Their method was limited to the processing element (PE) level, rather than the Configurable Logic Block (CLB) level, which means entire operational circuit structures were ignored when evolving around circuit faults. The design was also limited to offline use. That is, during reconfiguration, the device was inoperable. Other work displayed promise, but was only demonstrated through simulation with no indication of a physical implementation in the works. No furthering of their work was made. Notwithstanding the forgoing known methods, improved and/or alternative methods and apparatus for designing and implementing self-repairing digital circuitry via biologically-inspired algorithms remains highly desirable.

2.3 Constraints and Ethics

Realistic Constraints

There are many realistic projects that apply to our design of creating an evolutionary repair mechanism for fault tolerant digital circuitry.

The economic constraints placed on our design would be tremendous. As components within embedded systems continue to grow smaller and more complex, they grow more sensitive to any damage they are exposed to. Newer FPGAs and embedded systems are composed of parts can usually be replaced by a moderate fee. For this reason, our design should not raise the cost of current circuitry a lot more than it is today.

Environmental constraints towards our design are minimal. The algorithm that the group plans to design will ensure longevity for sensitive circuitry, which will actually serve the environment. There could be power constraints associated with the algorithm, and it will be an important issue to make sure that the algorithm doesnt consume a lot of power.

There arent many health and safety constraints with our design, since our idea is software based. Certain safety constraints would be that the algorithm shouldnt consume too much power, or change the circuitry code in a way that it behaves completely different. Our product will not be manufactured, but it will face design constraints based on the circuit our code is embedded within. Since different circuits behave differently, our code would only work with the devices for which can find test vectors to use with our code. This means that our algorithm would only work for circuits for which we can find test vectors.

There are many sustainability constraints placed on our design. Firstly, while the algorithm is fixing the circuit, the user should still be able to use their system. The power consumption by the algorithm should be minimal in order to effectively work in many systems. These are the realistic constraints of our design.

Professional and Ethical Responsibilities

The ethical and professional responsibilities of our design are immense. The algorithm should actually be able to discover the best possible reconfiguration for any circuit while making sure that all other functions of the system are properly executed after reconfiguration.

The group will have to make sure that no licensed material is accessed while attempting to test the design on a circuitry, which is why the group will be using test vectors to test the design.

Professional responsibilities of our design would be to make sure that once the algorithm is implemented in the system, it constantly evolves and attempts to enhance system performance to the standards it had when there were no defects. An important ethical responsibility would be to not only showcase the cases where our design works flawlessly but also showcase the areas in which the design fails to reach its goals. When conducting experiments with test vectors, the group will not omit any findings regarding the design. The group has not discovered these methods on their own, and all the articles and ideas that the group uses to construct the algorithm and test vectors will be cited and given credit.

Lastly, the group will make sure, that the topic is thoroughly researched and the best possible engineering design for improving circuitry is presented as the final design.

2.4 Objectives

Object Attributes

1. **Self-Healing**: The system shall be self-healing in nature. It will not require an externally connect computer to achieve reconfiguration. All operations will occur on the chip, independently.

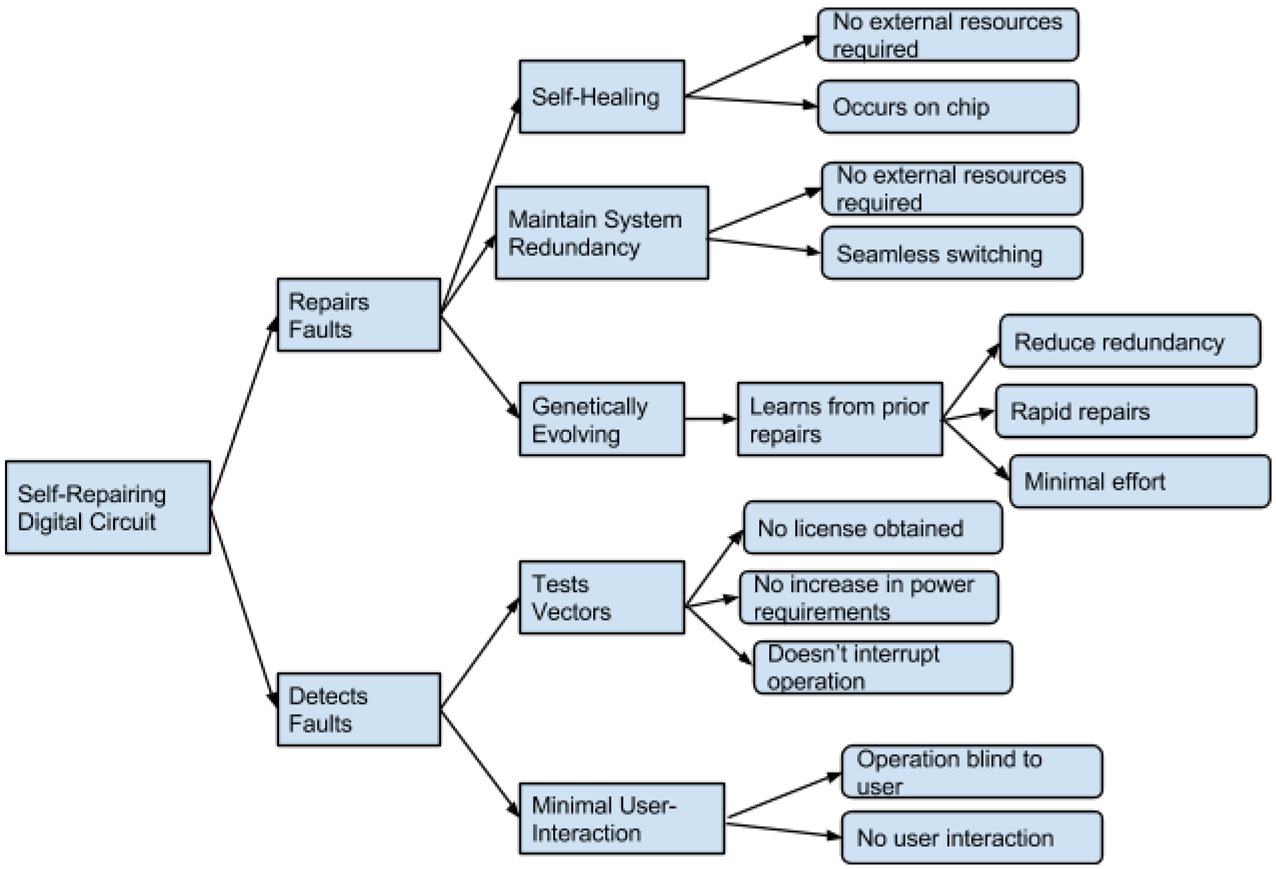
2. **Redundant**: During fault repair, there shall be no downtime to the chip. The chip will continue to operate while the fault is being repaired. This operational switch will be seamless.

3. **Genetically Evolving**: The chip shall learn from each prior repair. This will reduce redundancy causing minimal effort to repair the chip. Also, power and temperature requirements should not be affected by repair operations.

4. **Test Vector**: The test vector will be autonomously generated prior to operation. This will not require any user-license software to be accessed or obtained. Fault detection will occur regularly and not affect power requirements. Testing operations shall not affect normal operation of the chip.

5. **Minimal User Interaction**: Testing will be blind to the user and shall not require any manual user interaction.

Object Tree



References

[1] R. Salvador, A. Otero, J. Mora, E. de la Torre, L. Sekanina, and T. Riesgo. Fault tolerance analysis and self-healing strategy of autonomous, evolvable hardware systems. *In Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*, pages 164-169, 2011.

[2] J. Lohn, G. Larchev, and R. Demara. Evolutionary fault recovery in a virtex fpga using a representation that incorporates routing. *In Parallel and Distributed Processing Symposium*, 2003. Proceedings. International, pages 8 pp.-, 2003.

[3] J. Heiner, B. Sellers, M. Wirthlin, and J. Kalb. Fpga partial reconfiguration via configuration scrubbing. *In Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, pages 99-104, 2009.

[4] S. Harding, J.F. Miller, and W. Banzhaf. Self modifying cartesian genetic programming: Parity. *In Evolutionary Computation, 2009. CEC 09. IEEE Congress on*, pages 285-292, 2009.

[5] L. L. Goh. Novel fault localization approach for atpg / scan- fault failures in complex sub-nano fpga/ asic debugging. *In Physical and Failure Analysis of Integrated Circuits (IPFA), 2010 17th IEEE International Symposium on the*, pages 1-4, July 2010.

[6] Renovell, M.; Portal, J.M.; Figueras, J.; Zorian, Y., Testing the interconnect of RAM-based FPGAs, *Design and Test of Computers*, IEEE , vol.15, no.1, pp.45,50, Jan-Mar 1998.

[7] L. Zhao, D. M. H. Walker, and F. Lombardi, Bridging Fault Detection in FPGA interconnects using IDDQ, *International Symp. on Field Programmable Gate Arrays*, 1998, pp. 95-104.

[8] M. Renovell, J. M. Portal, J. Figueras, and Y. Zorian, SRAM-based FPGAs: Testing the LUT/RAM Modules, *Proc. International Test Conference*, 1998, pp. 1102-1111.

[9] M. Abramovici, C. E. Stroud, BIST-Based Test and Diagnosis of FPGA Logic Blocks, *IEEE Trans. on VLSI Systems*, 2001, pp. 159-172

[10] Miller, J.F.; Smith, S.L., Redundancy and computational efficiency in Cartesian genetic programming, *Evolutionary Computation, IEEE Transactions on*, vol.10, no.2, pp.167,174, April 2006