

Self-Stabilization Testing of LUT-Based FPGA Designs by Fault Injection

Michael Böhnel
Institute for Technical Informatics
Graz University of Technology, Austria
Inffeldgasse 16
A-8010 Graz, Austria
boehnel@iti.tu-graz.ac.at

Reinhold Weiss
Institute for Technical Informatics
Graz University of Technology, Austria
Inffeldgasse 16
A-8010 Graz, Austria
weiss@iti.tu-graz.ac.at

Abstract

New testing methods are required as the complexity of Field Programmable Gate Array (FPGA) designs grow rapidly and time-to-market demands shorten. In this paper we propose a new, physical fault injection method for the test of a system's self-stabilizing property, that is its intrinsic ability to recover from transient faults. Therefore we inject transient faults in Look-Up Table (LUT)-based FPGA designs by dynamical, partial reconfiguration.

1. Introduction

Little work concerning fault injection and FPGAs has been published. Either FPGAs are used as a flexible target specific interface for a physical fault injection tool or as a fault emulator for ASIC designs [1]. To our present knowledge no publication deals with fault injection in FPGA designs. Self-stabilizing systems suit perfectly to the concept of fault tolerance [2]. These systems can be started in an arbitrary state and will converge to its desired behavior. Since the occurrence of a transient fault stands for an illegal deviation in a system's state, self-stabilizing systems are inherently tolerant to transient faults.

2. New Approach and Preliminary Results

The fault injector is programmed in Java, which allows the use with different operating systems and in distributed systems. The interface is designed to adapt to different FPGA families and types. Actual FPGA parameters such as number of rows and columns or size of LUT are managed by the Java class. We assume a single temporary stuck-at in a flip-flop due to design errors, metastability or timing violations of peripheral components. Fault injection is performed by run-time manipulation of the LUTs. We inject

stuck-at faults by setting the LUT value to all-zeros (s-a-0) and all-ones (s-a-1), respectively. After leaving the fault active for some time, the original LUT value is restored. Duration for stabilization can be measured by an observer from this point of time. Injection place, instant and duration can be chosen randomly for self-stabilization testing. In this case, no knowledge of the internal design including reusable components is necessary. The usage of the slow serial JTAG interface has one major drawback: there are no exact timing synchronizations possible. Hence the reproducibility is limited, but can be neglected if only the design's stability is tested by random time fault injection. With our method we have analyzed a one-hot and binary encoded state-machine. The fault injection unit is still under development. Therefore, the actual step of modifying LUTs was performed theoretically. In our experiment binary encoding was self-stabilizing with the succeeding clock cycle after fault removal, while one-hot encoding was not.

3. Conclusions and Future Work

Our approach has many advantages: (i) No extra hardware or FPGA resources are used. (ii) No design source code/design details are needed. (iii) Fault injection by standardized JTAG interface. (iv) Applicable in distributed systems. (v) It suits to today's need for employment of reusable components. Future work will include the manipulation of extra memory blocks, provided by some FPGA families. In the longer run we will use our approach to evaluate a COTS-based, reliable voice communication switch.

References

- [1] K.-T. Cheng, S.-Y. Huang, and W.-J. Dai. Fault emulation: A new methodology for fault grading. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 18(10):1487–1495, October 1999.
- [2] S. Dolev. *Self-Stabilization*. MIT Press, 2000.