ISSN 2278 - 3091

International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 10 – 13 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp03.pdf

A Survey on High speed fault injection module for fault detection processor on FPGA

Ms. Shwetha N
M.Tech (VLSI & Embedded Systems)
Sri Siddhartha Institute of Technology
Tumkur, Karnataka, India

Mr. Sunil T D
Assistant professor, Department of ECE
Sri Siddhartha Institute of Technology
Tumkur, Karnataka, India

Dr. M Z Kurian
Professor and Head, Department of ECE
Sri Siddhartha Institute of Technology
Tumkur, Karnataka, India

ABSTRACT

Fault injection is mainly used to test and evaluate the fault-tolerance based designs. In current VLSI technology fault injection has become a popular technique for experimentally verifying the fault tolerant based designs. There are fundamentally two types of fault injection methods; they are hardware-based fault injection and software-based fault injection. Both have their own limitations and advantages. The FPGA synthesizable fault injection model can give reasonable solution with high speed testing platform and also allows good controllability and observability. Even though a considerable progress has been made in research part of the fault injection algorithms, there is a little progress done in developing a tool for FPGA based fault emulation.

An FPGA-based fault injection tool (FITO) that supports several synthesizable fault models of digital systems are implemented using Verilog. Aim is to build real time fault injection mechanism with good controllability and observability. Fault injection will be done by applying some extra gates and wires to the original design description and modifying the target Verilog model of the target system. Analysis will be carried out by studying the controllability and observability of the proposed scheme. Comparison will be carried out to estimate the speed wise improvement with respect to software simulation based fault injection method.

Keywords -VLSI, FITO, VHDL, fault modeling, modelsim, Xilinx, GUI, spartan 3E FPGA kit.

INTRODUCTION

A system may not always perform the function it is intended for. The cause and consequence of deviations from the expected function of a processor/system are called the factors to dependability. Fault is a physical defect, imperfection, or flaw that occurs within some hardware or software component. Due to the evolution of the technologies, the probability of fault occurring in integrated circuits is noticeably increasing.

Fault injection is defined as the validation technique of the dependability of fault tolerant systems which consists of controlled experiments where the observation of system’s behavior in presence of faults is induced explicitly by writing injection of faults in the system. Fault injection is mainly used to evaluate fault tolerant mechanisms. In the last decade, fault injection has become a popular technique for experimentally determining dependability parameters of a system, such as fault latency, fault propagation and fault coverage. Within the numerous fault injection approaches that have been proposed, there are two classifications of fault injection methods : 1) hardware-based fault injection and 2) software-based fault injection. Software-based fault injection methods are divided into software-implemented fault injections (SWIFI) and simulation-based fault injections.

In the simulation-based fault injection, faults are injected into the simulation model of the circuits using VHDL or Verilog languages. The main advantage of simulation-based fault injection as compared with other fault injection methods is the high observability and controllability. However, simulation-based fault injection methods are too time-consuming. One way to provide good controllability and observability as well as high speed in the fault injection experiments is to use FPGA-based fault injection. An effective FPGA-based fault injection technique should support several properties as below:

1) High controllability and observability,
2) High speed fault injection experiments with the target system running at full speed,
3) Capability of injecting permanent and transient faults,
4) Minimum area and time overhead into a target system.

This paper describes the FPGA-based fault injection tool, called, FITO which support all of the fourth properties as mentioned above and is based on Verilog description of the systems. FITO supports several fault models into RTL and Gate-level abstraction levels of the target system which has been described by the Verilog. For supporting high speed fault injection experiments, the fault injector part of FITO with low area overhead is implemented with synthesized microprocessor core inside the FPGA.

LITERATURE SURVEY

Shi-Y Huang and Wei-Jin Dai[1] Presented a method that uses the field programmable gate array (FPGA)-based emulation system for fault grading. The real-time simulation capability of a hardware emulator could significantly improve the performance of fault grading, which is one of the most time-consuming tasks in the circuit design and test process. Employ a serial fault emulation algorithm enhanced by two speed-up techniques. First, a set of independent faults can be injected and emulated at the same time. Second, multiple dependent faults can be simultaneously injected within a single FPGA-configuration by adding extra circuitry. Because the reconfiguration time of mapping the numerous faulty circuits into the FPGA's is pure overhead and could be the bottleneck of the entire process, using extra circuitry for injecting a large number of faults can reduce the number of FPGA-reconfigurations and, thus, improving the performance significantly.

Fault simulation for large sequential circuits remains a very time-consuming task. With the increasing performance of Field-programmable gate-array and logic emulation technology, a hardware fault emulation system has become not only feasible but also very efficient as compared with the existing software-based or hardware-accelerator-based approaches. This paper addresses the issues of realizing a fault emulator based on an existing logic emulation system. In addition to a primitive scenario of serial fault emulation, two techniques are proposed to further speed up the process. The concept of the independent fault set is exploited to allow parallel fault emulation, while the dynamic fault injection using extra circuitry attempts to break the performance bottleneck, i.e., the reconfiguration time. The experimental results show that the overhead of our conservative policy of four-input CLB realization for dynamic fault injection is modest. Meanwhile, the issue of incorporating the unknown logic value in the emulator is addressed. A dual-railed logic is used to augment our emulator with the ability to simulate the unknown logic value and, thus, to differentiate the hard detected faults from those potentially detected faults. It is worth mentioning that the use of the dual-railed logic is not always necessary: 1) For circuits with a reset state, there will be no potentially detected faults assuming that the reset hardware is fault-free. 2) For synchronous circuits, the percentage of the potentially detected faults is usually very low.

Peter Folkesson et al.[2] compares two fault injection techniques: scan chain implemented fault injection (SCIFI), i.e. fault injection in a physical system using built in test logic, and fault injection in a VHDL software simulation model of a system. The fault injections were used to evaluate the error detection mechanisms included in the Thor RISC microprocessor, developed by Saab Ericsson Space AB. The Thor microprocessor uses several advanced error detection mechanisms including control flow checking, stack range checking and variable constraint checking. A newly developed tool called FIMBUL (Fault Injection and Monitoring using Built in Logic), which uses the Test Access Port (TAP) of the Thor CPU to do fault injection, is presented. The simulations were carried out using the MEFISTO-C tool and a highly detailed VHDL model of the Thor processor. The results show that the larger fault set available in the simulations caused only minor differences in the error detection distribution compared to SCIFI and that the overall error coverage was lower using SCIFI (90-94% vs. 94-96% using simulation based fault injection).

P. Civera, L. Macchiarulo et al.[3] explained the widespread adoption of VLSI devices for safety-critical applications asks for effective tools for the evaluation and validation of their reliability. Fault Injection is commonly adopted for this task, and the effectiveness of the adopted techniques is therefore cc key factor for the reliability of the final products. In this paper we present new techniques for exploiting FPGAs to speed-up Fault Injection in VLSI circuits. Thanks to the suitable circuitry added to the original circuit, transient faults affecting memory elements in the circuit can be considered. The proposed approach allows performing Fault Injection campaigns that are comparable to those perjured with hardware-based techniques in terms of speed, but shows a much higher flexibility in terms of supported fault models.

The continuous increase in the integration level of electronic systems is making increasingly difficult to guarantee an acceptable degree of reliability, due to the occurrence of transient faults (often modeled as soft errors) that can dramatically affect the behavior of the system.

ISSN 2278 - 3091

International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 10 – 13 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp03.pdf

To face the above issue, mechanisms are required to increase the robustness of electronic devices and systems with respect to possible errors occurring during their normal function. At the same time, designers strongly need effective techniques and methods to debug and verify their correct design and implementation.

Babak Rahbaran et al.[4] described an Experimental fault-injection plays a key role in the process of fault tolerance validation. This paper discuss the limitations of conventional experimental setups and investigate how highly complex FPGA scan aid in overcoming these .Based on a thorough analysis of the potential aims of fault-injection experiments derive a set of conditions for the design of an FPGA-based fault-injection toolset. Present the fault-injection tool FIDYCO as an example implementation of this concept. Our FPGA-based toolset has three main advantages: First, the availability of a physical target system allows to perform experiments in real time. Second, the programmable nature of the FPGA target platform facilitates controllability and observability comparable to that of simulation-based approaches. Third, the tight integration of fault injector and device under test on the same hardware platform allows for a higher precision of fault injection and diagnostic resolution.

Dependability is the property of a system that justifies to have confidence in it. Fault Tolerance is the ability of a system to operate correctly in the presence of faults. Obviously fault tolerance can only be attained with respect to a given number and class of faults. A failure is the deviation of the delivered service from the system specification or equivalently the transition from proper service delivery to improper service delivery .An error is the manifestation of a fault in a system. An error is liable to lead to a subsequent failure. Whether or not an error will actually lead to a failure depends on factors like (a) available redundancy, (b) system activity or (c) the definition of what is viewed as a failure. A fault is the hypothesized cause of an error. Depending on what is viewed a"system", a fault can in turn be the consequence of a failure of another (sub-)system.

Bruno Pacheco Sanches et al.[5] explained Software faults are known as a major cause of computational systems' defects. Even when these systems are tested exhaustively they can present some failures due to the activation of residual software faults in the source code. Software fault injection tools are useful to emulate the presence of software faults and to monitor the system allowing one to observe if the system continues to operate as expected. A tool helps to evaluate the possible failures in order to define countermeasures to avoid them or to reduce their severity, increasing the levels of dependability of the application under test. This work presents the J-SWFIT tool, which emulates Java software faults directly in compiled code. The architecture of the tool was proposed in an abstract level that can be easily understood and extended. J-SWFIT works based on a set of predefined Java operators and consists of analyzing the byte codes of compiled Java files, finding locations where specific faults can exist and can be injected each one independently. J-SWFIT allows comparing the systems' behavior in the presence and absence of each fault.

Volkmar Sieh et al.[6] presented a new technique for reliability evaluation of digital systems will be presented by demonstrating the functionality and usage of the simulation based fault injector VERIFY (VHDL-based Evaluation of Reliability by Injecting Faults efficiently). This software tool introduces a new way for describing the behavior of hardware components in case of faults by extending the VHDL language with fault injection signals together with their rate of occurrence. The accuracy of the results is obtained by using the same VHDL-models which have been developed during conventional phases of hardware design. For demonstrating the capabilities of VERIFY, a VHDL-model of a simple 32-bit processor (DP32) will be used as an example to illustrate the several steps of reliability evaluation.

VERIFY determines the recovery time of each injected fault by the following algorithm. Before injecting the fault (during phase) all corresponding signals will have the same value. During (small black rectangle) and after fault injection values of signals of the golden run and the faulty run will differ (phase). If fault recovery is successful all signals will return to legal values (phase). The overhead introduced by the error recovery, it is the recovery time, is the difference of the length in time of phase of the faulty run and phase of the golden run.

DESIGN OF FITO

FITO environment consists of three parts

1) Source code modifier and fault list generator.
2) Fault injection manager
3) Results capturing with FPGA emulation.

ISSN 2278 - 3091

International Journal on Advanced Trends in Computer Science and Engineering (IJATCSE), Vol. 4 No.1, Pages : 10 – 13 (2015)
Special Issue of ICACE 2015 - Held on March 02, 2015,Mysore, India
http://warse.org/pdfs/2015/icace2015sp03.pdf

1) Source code modifier and fault list generator Source code modifier and fault list generator are the software parts of the FITO.

2) Fault injection manager

Fault injection manger is responsible for performing the real time fault injection. The fault injection manager is implemented in Verilog. The fault injection manager

- Fault scheduler
- Fault insertion components

The fault scheduler runs multiple counters to schedule each fault with required fault activation time and fault propagation time. The fault scheduler produces output fault number which is currently being active. This module generates the parallel fault injection signals for every fault. These signals are routed to all fault sites. Fault insertion components are gates with FIS (fault injection signal) control to inject the faults when the FIS is active high. These components instances are automatically made when ever faults are injected.

3) Results capturing with FPGA emulation

This hardware part is implemented on the FPGA board. Result analysis will be carried out with FPGA emulation results and fault list generated. The analysis shall summarize the fault responses for each injected fault. The following Fig 1 shows the fault injection process with FITO.

The golden trace data is ideal trace data (results obtained) without any faults

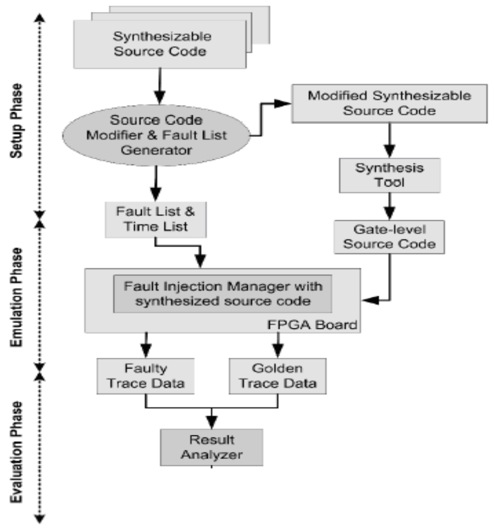

Fig 1. FITO injection process

CONCLUSION

The project implements the FPGA based fault injection called FITO for evaluating the digital systems modeled in Verilog. Fault injection with FITO is done by applying some extra gates and wires to the original design description and modifying the target Verilog model of the target system.

REFERENCES

[1] K.-T. Cheng, S.-Y. Huang, and W.-J. Dai, “Fault emulation:A new methodology for fault grading,” Trans. on the IEEE Computer-Aided Design of Integrated Circuits and Systems, Oct. 1999, pp. 1487-1495.

[2] P. Folkesson, S. Sevensson, and J. Karlsson, “A compa-rsion of simulation based and scan chain implemented fault injection,” Proc. of the Annual International Symposium on Fault-Tolerant Computing,Jun. 1998, pp. 284-293.

[3] P. Civera, L. Macchiarulo, M. Rebadengo, M. S. Reorda, and M. A. Violante, “Exploiting FPGA for accelerating fault injection experiments,” Proc. of the International On-Line Testing Workshop, 2001, pp. 9-13.

[4] B. Rahbaran, A. Steininger, and T. Handl, “Built-in fault injection hardware – The FIDYCO example,” Proc. of the IEEE International Workshop on Electronic Design, Test and applications.

[5] Bruno Pacheco sanches and Regina Moraes, “J-SWFIF: A Java Software Fault Injection Tool”2011

[6] V. Sieh, O. Tschache, F. Balbach, “VERIFY: evaluation of reliability using VHDL-models with embedded fault descriptions”, Proc. 27th Int. Symp. on Fault-Tolerant Computing, (FTCS-27), Digest of Papers, June 1997.