

A Study of the Experimental Validation of Fault-Tolerant Systems using different VHDL-Based Fault Injection Techniques*

J. Gracia, J.C. Baraza, D. Gil, P.J. Gil

Grupo de Sistemas Tolerantes a Fallos (GSTF - DISCA)

Universidad Politécnica de Valencia. Spain

e-mail: {jgracia, jcbaraza, dgil, pgil}@disca.upv.es

Abstract

Three different VHDL-based fault injection techniques have been compared to validate a fault tolerant micro-computer system. We have studied the error pathology, their detection and recovery coverages and their latencies.

1 Introduction

In the simulated fault injection technique, the system under test is simulated in another computer system. Faults are induced altering the logical values of the model elements during the simulation. The work presented here is framed in the simulation of models based on the VHDL hardware description language [1].

2 VHDL-based fault injection techniques

We have implemented three main classic techniques [1]: simulator commands, *saboteurs* and *mutants*. Some improvements have been added. In *saboteurs* technique, the classification of [1] has been extended by introducing bidirectional serial simple/complex *saboteurs* and adapting their designs to buses. In *mutants* technique, we have implemented transient mutants by dynamic modification of the behavioral architectures using guarded assignments in blocks.

3 Fault models

We have aimed to use a wide set of fault models that would represent real physical faults that occur in ICs. The models for simulator commands and *saboteurs* techniques have been deduced from the physical causes and mechanisms implied in the occurrence of faults [2]. Some examples are: stuck-at (0,1), bit-flip, indetermination, open-line, stuck-open, bridging and delay. Fault models for mutants have been generated by syntactical changes on VHDL code [3].

4 Fault injection experiments. Results.

We have validated an academic microcomputer system model, duplex with cold stand-by sparing, parity detection and watchdog timer [2]. Permanent and transient faults have been injected, with random injection place and instant. All the experiments have been made with the help of the injection tool developed by the GSTF [4].

Some significant results (Figs.1-2) show that coverages in transient faults can be obtained quite accurately using

any of the techniques. In permanent faults, *mutants* technique shows some discrepancy respect to the other techniques in the values of detection and recovery coverages. Our conclusion is that it would be necessary to improve the fault models for permanent *mutants*. In any case, the combined use of the three techniques seems to provide a powerful method to validate models with different abstraction levels.

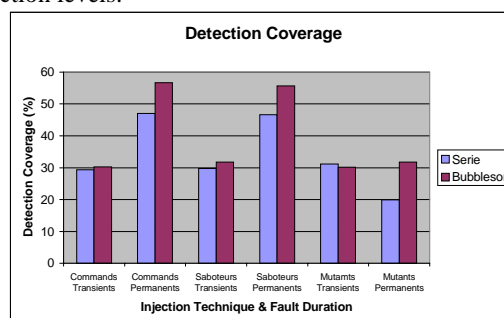


Figure 1: Detection coverage related to the injection technique and fault duration.

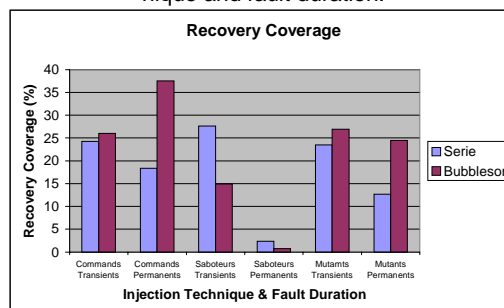


Figure 2: Recovery coverage related to the injection technique and fault duration.

References

- [1] E. Jenn et al. "Fault Injection into VHDL Models: The MEFISTO Tool", *FTCS-24*, Austin, Texas (USA), June 1994.
- [2] D. Gil, "Validación de Sistemas Tolerantes a Fallos mediante inyección de fallos en modelos VHDL". Tesis Doctoral, DISCA-UPV, (Spain). 1999.
- [3] J.R. Armstrong et al. "Test generation and Fault Simulation for Behavioural Models". *Performance and Fault Modelling with VHDL* (J.M. Schoen ed.), Englewood Cliffs, Prentice Hall, 1992.
- [4] J.C. Baraza et al. "A Prototype of a VHDL-Based Fault Injection Tool", *DFT2000*, Yamanashi (Japan), October 2000.

* This work has been partially funded by the Spanish research project CICYT TAP96-1090-C04-01