

# INERTE: Integrated NEXus-based Real-Time fault injection tool for Embedded systems<sup>1</sup>

P.Yuste, D. de Andrés, L.Lemus, J.J.Serrano, P.Gil  
Technical University of Valencia, Spain  
{pyuste,ddandres,lemus,jserrano,pgil}@disca.upv.es

## 1. INTRODUCTION

Software implemented fault injection techniques (SWIFI) enable emulation of hardware and software faults. This emulation can be based on debugging mechanisms of general purpose processors [1] or in special debugging ports of embedded processors [2]. A well-known drawback of existing SWIFI tools rely on the temporal overhead introduced in the target system. This overhead is a problem when validating real-time systems.

This paper presents a new SWIFI tool (INERTE) that solves this problem by using a standard debug interface called Nexus [3]. Using Nexus, system memory can be accessed at runtime without any intrusion in the target system. Thus, INERTE is able to inject transient faults without any temporal overhead.

## 2. INERTE: BRIEF DESCRIPTION

Fig. 1. shows the three functional parts of INERTE: the Experiment Generator Module, the Fault Injector and the Analysis Tool.

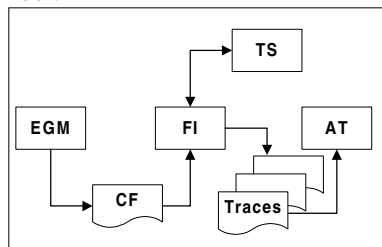


Fig.1. Block diagram of INERTE

### 2.1. Experiment Generator Module (EGM)

EGM inputs are the memory address range where faults can be injected, the number of injections to be performed and the random distribution (uniform, normal and Erlang) associated to these injections. Using this information, the EGM generates configuration files (CF) that are used during fault-injection campaigns. These files contain the memory addresses to modify and the mask bytes to be used during experiments by the fault-injector.

<sup>1</sup>This work is sponsored by the DBench Dependability Benchmarking IST-2000-25425 project, funded by the European Community under the "Information Society Technology" Programme (1998-2002)

### 2.2. Fault Injector (FI)

The FI is a script written in Practice language. This script uses the CFs generated by the EGM in order to run a sequence of fault injection experiments. Before each experiment, the system is reset and the target system (TS) is restored to have a common experiment starting point. Then, the target system runs and the FI injects a fault and waits for its activation. Thanks to Nexus, fault injections can be performed without stopping the normal execution of the system. Furthermore, the activity of the target system can be also traced without any temporal overhead. The end of each experiment occurs after a pre-determined time from the injection of the fault. The trace of the system activity is then stored in a database and the FI starts the next experiment.

### 2.3. Analysis Tool (AT)

The AT is a parser that compares the traces obtained by the FI from each fault injection experiment and those associated with fault-free runs of the system (called *golden-runs*). The AT extracts timing information and searches for differences between them. It also deduces whether or not the injected fault has been activated and the error detection mechanisms have been properly triggered. Times between these events are also obtained. That is how INERTE computes error detection coverage and latencies.

## 3. CONCLUSION

These few lines summarize the description of INERTE, a SWIFI tool that uses Nexus for eliminating the temporal overhead of fault-injection in real-time systems. INERTE is being used in the DBench project to develop a benchmark for an embedded diesel engine control application.

- [1] Carreira, J. et al. "Xception A Technique for the Experimental Evaluation of Dependability in Modern Computers". IEEE Transactions on Software Engineering, vol. 24, 1998. pp. 125-136
- [2] Rebaudengo, M., Sonza Reorda, M. "Evaluating the Fault Tolerance Capabilities of Embedded Systems via BDM". Proc. VTS99: 17th IEEE VLSI Test Symposium, 1999, pp. 452-457
- [3] The Nexus 5001 Forum™ Standard for a Global Embedded Processor Debug Interface. IEEE-ISTO 5001-1999. <http://www.ieee-isto.org/Nexus5001>. 1999