Exploration of Reliability-oriented Design Techniques for Multicore Systems

Felipe Rocha da Rosa¹, Luciano Ost², Ricardo Reis¹

¹ Instituto de Informática - PGMicro/PPGC, Universidade Federal do Rio Grande do Sul
Av. Bento Gonçalves 9500 Porto Alegre, RS, Brazil
{frdarosa, reis}@inf.ufrgs.br

² Department of Engineering - University of Leicester
University Road, Leicester, UK
luciano.ost@le.ac.uk

Abstract—The combination of constantly shrinking technology and ever-increasing on-chip power density and temperature operation calls for novel reliability-oriented techniques which can provide reliable system operation while meeting performance and energy constraints. This work reports techniques and tools, which have been developed during my first two PhD years.

Keywords — Reliability, multicore systems, fault injection,

I. INTRODUCTION AND MOTIVATION

Integrating multiple Off-The-Shelf (COTS) processors in the same system is now commonplace in both embedded system and high performance computing (HPC) domains [1]. Such systems aim to perform complex application workloads (i.e. billions of object code instructions), which will continue to grow in diverse fields (e.g. aerospace, automotive, etc.) [2]. To meet the system performance, processors of these systems may be required to operate in aggressive clock frequencies (i.e. gigahertz). The high frequency operation and the continuous technology shrink are making underlying systems more susceptible to soft errors, such as the ones caused by radiation effects [3], [4]. Soft errors or Single Event Effects (SEE) induced by neutron may cause critical failures on system behavior, which can lead to financial or human life losses [5]. In this regard, the occurrence of SEEs has been considered as a major concern in memory cells and processors working at ground level.

II. THESIS OBJECTIVE

The main goal of this Thesis is to investigate, develop and evaluate techniques and tools to improve the whole system reliability while providing appropriate guarantees on performance and energy efficiency. Underlying techniques should rely on system-level, i.e. techniques that can be applied and evaluated at application, operating system (OS) or architectural level.

III. THESIS CONTRIBUTIONS

This Thesis has so far two main contributions: (A) a methodology to evaluate SRAM cells vulnerability and (B) investigation of software error occurrence and impact on multicore system architectures.

A. Methodology to evaluate SRAM cells vulnerability

Proposal of a methodology that enables to determine the effects of temperature and voltage scaling in neutron-induced bit-flip in SRAM memory cells [6], [7]. This methodology allows determining the critical charge according to the dynamic behavior of the temperature as a function of the voltage scaling in FinFET and bulk SRAM cells. Experimental results show that both temperature and voltage scaling can increase in at least two times the susceptibility of SRAM cells to soft error rate (SER). In addition, a model for electrical simulation for soft error and different voltages was described to investigate the effects observed in the practical neutron irradiation experiments. Results can guide designers to predict soft error effects during the lifetime of SRAM-based devices considering different power supply modes.

B. Multicore soft error reliability analysis

This second contribution includes the following achievements:

i. Development of a fast and scalable fault injection framework based on a JIT-based simulator [8].

ii. Extending gem5 with fault injection capability, aiming at supporting soft error reliability analysis considering microarchitecture aspects.

iii. Comparison between both fault injection frameworks in terms of simulation speed/precision tradeoff.

Our first step was to implement a fast and flexible fault injector framework, called OVPSim-FIM, which supports parallel simulation to boost up the fault injection process. The fault injection campaign is displayed in Figure 1 and divided as follow:

1) Model simulation without faults (memory and CPU context collection);
2) Select the fault location (e.g. register) and injection time;
3) Model simulation with fault injections;
4) Faults are analyzed and classified;
5) Each application behavior is compared to the golden run and an error report is generated

Aiming at validating OVPSim-FIM, several fault injection campaigns were performed in ARM processors, considering a
market leading RTOS and benchmarks with up to 10 billion object code instructions. Results have shown that OVPSim-FIM enables to inject faults at speed of up to 10,000 million instructions per second (MIPS) (in a quad-core Intel i7), depending on the processor and the benchmark profile as shows in Figure 2, enabling to identify errors and exceptions according to different criteria and classifications.

OVPSim enables large scenarios explorations, capable of simulate hundreds of processors simultaneously. Nevertheless, this simulation speed comes with precision. OVPSim does not model cycle-accurate processors but rather instruction accurate processors, which limits the exploration of some techniques (e.g. cache reliability, internal architecture soft errors). To broaden the study of soft errors and compare the precision provide by the OVPSim-FIM, we develop a similar fault injection framework, running the same applications and applying the same speed-up techniques (e.g. checkpointing, parallel executions) using a well-known ARM cycle-accurate simulator, gem5 simulator [9]. In similar scenarios, the simulation speed reaches up to 2 MIPS and 0.4 MIPS in atomic and detailed mode respectively.

Results also show that the accuracy of OVPSim-FIM is below to 10% when compared to a gem5-based implementation. Figure 3 shows three benchmarks comparison using the gem5 atomic and detailed simulation model and the OVPSim-FIM. At same time that can reach up to 5000x times faster simulation speed.

REFERENCES