Early Component-Based System Reliability Analysis for Approximate Computing Systems

Early Component-Based System Reliability Analysis for Approximate Computing Systems

A.Vallero, A.Savino, G.Politano, S.Di Carlo
Politecnico di Torino
Torino, Italy

A.Chatzidimitriou, S.Tselonis, M.Kaliorakis, D.Gizopoulos,
University of Athens
Athens, Greece

M.Riera, R.Canal, A.Gonzalez
Universitat Politècnica de Catalunya
Barcelona, Spain

M.Kooli, A.Bosio, G.Di Natale
LIRMM, Montpellier
Montpellier, France

***Abstract*— A key enabler of real applications on approximate computing systems is the availability of instruments to analyze system reliability, early in the design cycle. Accurately measuring the impact on system reliability of any change in the technology, circuits, microarchitecture and software is most of the time a multi-team multi-objective problem and reliability must be traded off against other crucial design attributes (or objectives) such as power, performance and cost. Unfortunately, tools and models for cross-layer reliability analysis are still at their early stages compared to other very mature design tools and this represents a major issue for mainstream applications. This paper presents preliminary information on a cross-layer framework built on top of a Bayesian model designed to perform component-based reliability analysis of complex systems.**

Keywords-component; reliability analys; cross-layer

I. INTRODUCTION

It is well known today that aggressive scaling of hardware feature sizes followed to improve performance and computing capability produces systems that are increasingly susceptible to soft errors [1][2]. While this has been considered for several years as a severe threat to be mitigated by different techniques able to detect and mask soft errors [3][4][5][6][7] (which typically incur time or energy overhead), some researchers now see this as an opportunity to develop tools and techniques that enable applications that are inherently tolerant to soft errors to execute on unreliable hardware with increased performance and reduced power consumption [3][8][9][10][11].

Regardless of the approach used to enable applications to be executed on unreliable hardware, one of the key elements to enable real applications execution on approximate computing systems is the availability of instruments to analyze the reliability of the system, early in the design time. Such tools are important to clearly understand if the loss of quality of the generated results due to the unreliable hardware is still compatible with the reliability requirements of the applications.

Accurately measuring the impact on system reliability of any change in the technology, circuits, microarchitecture and software is most of the time a multi-team multi-objective problem and reliability must be traded off against other crucial design attributes (or objectives) such as power, performance and cost. Unfortunately, tools and models for cross-layer reliability analysis are still at their early stages compared to other very mature design tools and this represents a major issue for mainstream applications [13][14].

This paper summarizes a cross-layer framework built on top of a Bayesian model designed to perform component-based reliability analysis of complex systems. The framework is developed under the EU FP7 project CLERECO (http://www.clereco.eu) whose goal is to enable early reliability evaluation of complex systems.

In component-based system modeling it becames quickly evident that the whole system is more than the sum of its parts. Each component of the system can affect global perceivable properties of the entire system. By carefully integrating parameters obtained by the characterization of technologies, circuits, microarchitectural components and software (including the OS) into a Bayesian model of the prospective system, we are capable of evaluating very accurately the reliability of the full system.

It is important to remark that a statistical model itself would be useless in real applications without providing the instruments to compute its parameters for a specific application and a specific computing system. Together with the proposed model we have developed a complete framework comprising a tool chain able to analyze the most important parts of a complex system providing all necessary information to realize and validate the proposed reliability assessment model.

II. SYSTEM LEVEL RELIABILITY MODEL

In this paper we focus on errors caused by defects at the low-level technology layer of a system due to effects such as process variations, radiation, aging, etc. [15]. Errors resulting from low-level faults may manifest, be masked or be propagated through all Hardware (HW) and Software (SW) layers of the system stack eventually resulting in partial or total failure of the system activities (Figure 1.).

To support reliability analysis of complex systems we use a *component-based* (CB) reliability model [16]. In CB reliability, the reliability of a system is estimated using reliability information and other properties (e.g., size, complexity, etc.) of its individual components and their interconnections (the system architecture). Reliability estimation is performed early in the design cycle before the full system design takes place. This in turn supports architectural decisions about component characteristics and

gives indications about those components that are critical and deserve customized development efforts.

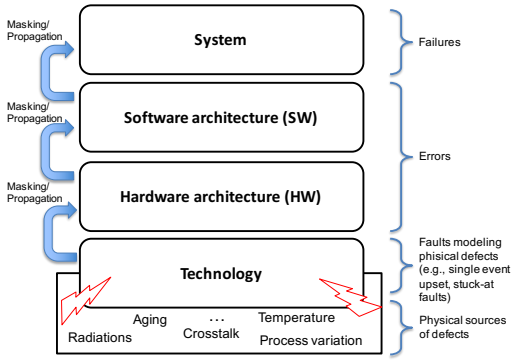

Figure 1. System stack – faults originate at the lower level of the system stack and are either masked or propagated to the upper layer eventually resulting in a failure at system level.

Figure 3. shows the architecture of the proposed reliability analysis framework. Our model exploits Bayesian Networks (BNs) as a statistical foundation for full system reliability estimation. Some of the reasons that lead to the use of BNs for system reliability modeling are:

- their efficient calculation scheme,
- intuitive representation of all system components,
- capability of fitting on field data,
- compact representation and decision support.

Our component-based BN model of a system includes a *qualitative model* represented by the network itself that models the architecture of the system and a *quantitative model,* representing the parameters of each component and their relations (modeled as a set of Conditional Probability Tables – CPT).

The nodes of the network correspond to the components of the system whose reliability is associated to a set of random variables, while arcs define temporal or physical relations among components, e.g., a failure state of a component may influence the state of other components.

Nodes are grouped in different domains:

- *Technology domain:* it models the physical technological layer of the system.
- *Hardware domain*: it models the hardware architecture. It comprises hardware blocks such as CPUs, GPUs, memories, accelerators, custom IP cores, etc. Granularity at which hardware blocks are modeled in this domain depends on the level of details the designer needs to obtain from the reliability analysis, and on the degree of freedom the designer has with the design of selected components.
- *Software domain*: This domain models the software architecture. To decouple the analysis of the software domain from the hardware domain special attention is required to define the interface between the two layers. A set of special nodes denoted as Software Fault Models (SFMs) translate hardware failures into the software domain and therefore decouple the hardware domain from the software domain so that the corresponding supporting tools can operate in parallel. SFMs are mainly based on alterations (due to faults) that have an impact on the Instruction Set Architecture (ISA) of the microprocessor.
- *System domain*. It is the higher-level domain of the system. Nodes in this domain are basically output nodes of the system, i.e., nodes in which a fault can be observed as a failure of the system.

Every domain has a dedicated set of tools for the characterization of its components.

In the technology domain we have developed a framework built on top of the HSPICE simulator able to characterize the main building blocks composing any logic circuit and to compute their marginal fault probability with respect to a given failure mode. The considered blocks include sequential blocks (i.e., memory cells, flip-flops and latches), as well as combinatorial blocks (i.e., logic gates). Each block is analyzed for different run-time parameters (i.e., combinations of voltage, temperature and geographical location). So far our technology library comprises: Bulk Planar CMOS 22nm and 16nm, Bulk Fin FET 20nm and 14nm, SOI Planar 22nm [17].

At the hardware domain we focused our effort on the development of a tool-chain able to characterize the different microarchitectural blocks of any microprocessor. Microprocessors are our main target since they represent one of the most complex and important blocks of a system. In order to estimate accurately the reliability of the hardware layer we resort to a microarchitecture-level fault injection, which delivers very accurate results for array-based structures. Two tools built on top of the well-known Gem5 and Marssx86 simulators have been developed to enable characterization of both x86 and ARM architectures [18].

At the software layer, to enable software characterization in isolation from the target platform, we resort to LLVM (Low Level Virtual Machine), a compiler framework that uses virtualization with virtual instruction sets to perform complex analysis of the software applications. We built a high-level software fault injection framework able to inject software fault models within an application and observe its effect on the software outcome.

At the system level contributions from all tools are combined into a BN model of the system that is used to reason about the overall reliability properties of the system. Both Diagnostic reasoning from symptoms to cause, and predictive reasoning starting from the information about causes (i.e., raw technology failure rates) to new beliefs about effects is supported.

By resorting to this high-level statistical reasoning, system designers are provided with a tool enabling to perform early estimation of the overall system reliability and to analyze the effect hardware unreliability has on the quality

of the system's results. Moreover, using diagnostic reasoning, weak components can be probabilistically identified. This provides means to drive the reliability design effort toward the most critical components of the system thus optimizing the overall system.

III. Experimental Results

In order to give a preliminary insight of the capability of the proposed framework we propose here its application to the analysis of an x86 based system running a single application.

The hardware architecture of the system is summarized in TABLE I. The system is based on an x86 out-of-order microprocessor built with a 22nm Bulk Planar technology (ASU PTM Models). We assume the RAM is protected by ECC and the analysis focuses on faults in the microprocessor structures.

TABLE I. Hardfware Architecture

x86 out-of-order-model		
Component	***Size***	***Technology***
Register file (256 regs each 64-bits)	2KB	22nm Bulk Planar
L1 Instruction Cache	32KB	22nm Bulk Planar (8T SRAM)
L1 Data Cache	32KB	22nm Bulk Planar (8T SRAM)
L2 Cache	1 MB	22nm Bulk Planar (6T SRAM)
Load/Store Queue	128 bytes	22nm Bulk Planar
Main memory		
DRAM protected with ECC (i.e., no fault injected)		

The system executes the string search application from MiBench[1].

We analyzed the system first through a complete fault injection campaign resorting to the GeFIN fault injector [18]. We resorted to statistical fault sampling as described in [19] to compute the number of faults (single bit flips) to inject. For all the hardware structures of our study, the number of fault injection runs was set to 2000, which corresponds to 2.88% error margin and 99% confidence level. For each injection the full application has been simulated and results classified into one of the following classes: benign, silent data corruption, unresponsive.

The same analysis has been performed resorting to our reliability analysis framework.

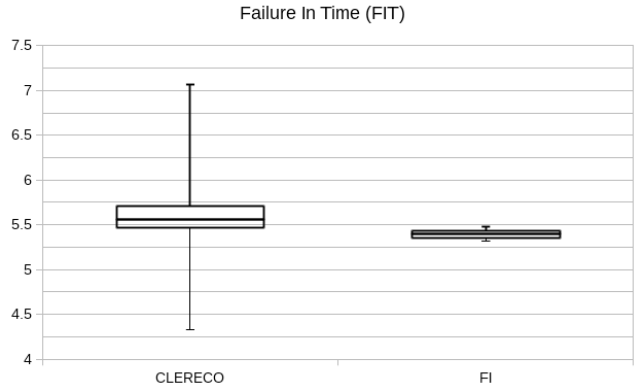

Figure 2. Failure in time (FIT) for the considered system computed using the CLERECO reliability framework and full fault injection campaign.

Figure 2. compares the result in terms of accuracy of the computed reliability. Results obtained with the fault injection have been de-rated using the error probability for the target technology available in our technology library. The figure clearly show the accuracy of the framework compared to a full fault injection campaign. Interestingly, computation time is reduced of about one order of magnitude, with significant benefits for the system designers.

IV. Conclusions

This paper presents preliminary results on the development of a cross-layer framework built on top of a Bayesian model designed to perform component-based reliability analysis of complex systems.

The framework enables very accurate analysis of the reliability of a system with significant reduction on the simulation time. It therefore represents an interesting instrument for system designer that need to trade off reliability of their systems with other parameters such as performance and power.

Acknowledgment

This paper has been fully supported by the 7th Framework Program of the European Union through the CLERECO Project, under Grant Agreement 611404.

References

[1] F. Cappello, A. Geist, B. Gropp, L. Kale, B. Kramer, and M. Snir. Toward exascale resilience. International Journal of High Performance

[2] P. Shivakumar, M. Kistler, S.W. Keckler, D. Burger, and L. Alvisi. Modeling the effect of technology trends on the soft error rate of combinational logic. DSN, 2002.

[3] M. de Kruijf, S. Nomura, and K. Sankaralingam. Relax: an architectural framework for software recovery of hardware faults. ISCA '10.

[4] D. Ernst, N. S. Kim, S. Das, S. Pant, R. Rao, T. Pham, C. Ziesler, D. Blaauw, T. Austin, K. Flautner, and T. Mudge. Razor: A low-power pipeline based on circuit-level timing speculation. MICRO, 2003.

[1] http://wwweb.eecs.umich.edu/mibench/)

[5] F. Perry, L. Mackey, G.A. Reis, J. Ligatti, D.I. August, and D. Walker. Fault-tolerant typed assembly language. PLDI, 2007.

[6] J. Ray, J. Hoe, and B. Falsafi. Dual use of superscalar datapath for transient-fault detection and recovery. MICRO, 2001.

[7] G. Reis, J. Chang, N. Vachharajani, R. Rangan, and D. August. Swift: Software implemented fault tolerance. CGO 05, 2005.

[8] M. Carbin and M. Rinard. Automatically identifying critical input regions and code in applications. ISSTA, 2010.

[9] S. Misailovic, S. Sidiroglou, H. Hoffmann, and M. Rinard. Quality of service profiling. ICSE, 2010.

[10] M. Rinard. Probabilistic accuracy bounds for fault-tolerant computations that discard tasks. ICS, 2006.

[11] A. Sampson, W. Dietl, E. Fortuna, D. Gnanapragasam, L. Ceze, and D. Grossman. Enerj: Approximate data types for safe and general low-power computation. PLDI, 2011.

[12] D.W. Coit, J. Tongdan, N. Wattanapongsakorn, "System optimization with component reliability estimation uncertainty: a multi-criteria approach," in Reliability, IEEE Transactions on , vol.53, no.3, pp.369-380, Sept. 2004 doi: 10.1109/TR.2004.833312

[13] A.Evans, M.Nicolaidis, S.-J.Wen, D.Alexandrescu, and E.Costenaro, "RIIF - reliability information interchange format," in On-Line Testing Symposium (IOLTS), 2012 IEEE 18th International, June 2012, pp. 103– 108.

[14] U.Schlichtmann, V.Kleeberger, J.Abraham, A.Evans, C.Gimmler- Dumon, M.Glas, A.Herkersdorf, S.Nassif, and N.Wehn, "Connecting different worlds - technology abstraction for reliability-aware design and test," in Design, Automation and Test in Europe Conference and Exhibition (DATE), 2014, March 2014, pp. 1–8.

[15] T.Austin, V.Bertacco, S.Mahlke, and Y.Cao. "Reliable Systems on Unreliable Fabrics" IEEE Design & Test of Computers, 25(4): 322-333, July 2008.

[16] A. Filieri, C. Ghezzi, V. Grassi, R. Mirandola, Reliability analysis of component-based systems with multiple failure modes. In Component-Based Software Engineering (Springer), pp. 11-20, 2010.

[17] S. Ozdemir, N. Aymerich, M. Riera, R. Canal, A. González, M. Kaliorakis, S. Tselonis, N. Foutris, D. Gizopoulos, A. Benso, S. Di Carlo, "Characterization of failure mechanisms for fu- ture systems (preliminary)" [on-line] http://www.clereco.eu/images/deliverables/D2.2.1-characterizaion-of-failure-mechanisms_v1.4_final.pdf

[18] M.Kaliorakis, S.Tselonis, A.Chatzidimitriou, and D.Gizopoulos, "Differential Faut Injection on Microarchitectural Simulators", in IEEE International Symposium on Workload Characterization (IISWC), Oct. 2015

[19] R.Leveugle, A.Calvez, P.Maistri, P.Vanhauwaert, "Statistical fault injection: Quantified error and confidence", DATE 2009.

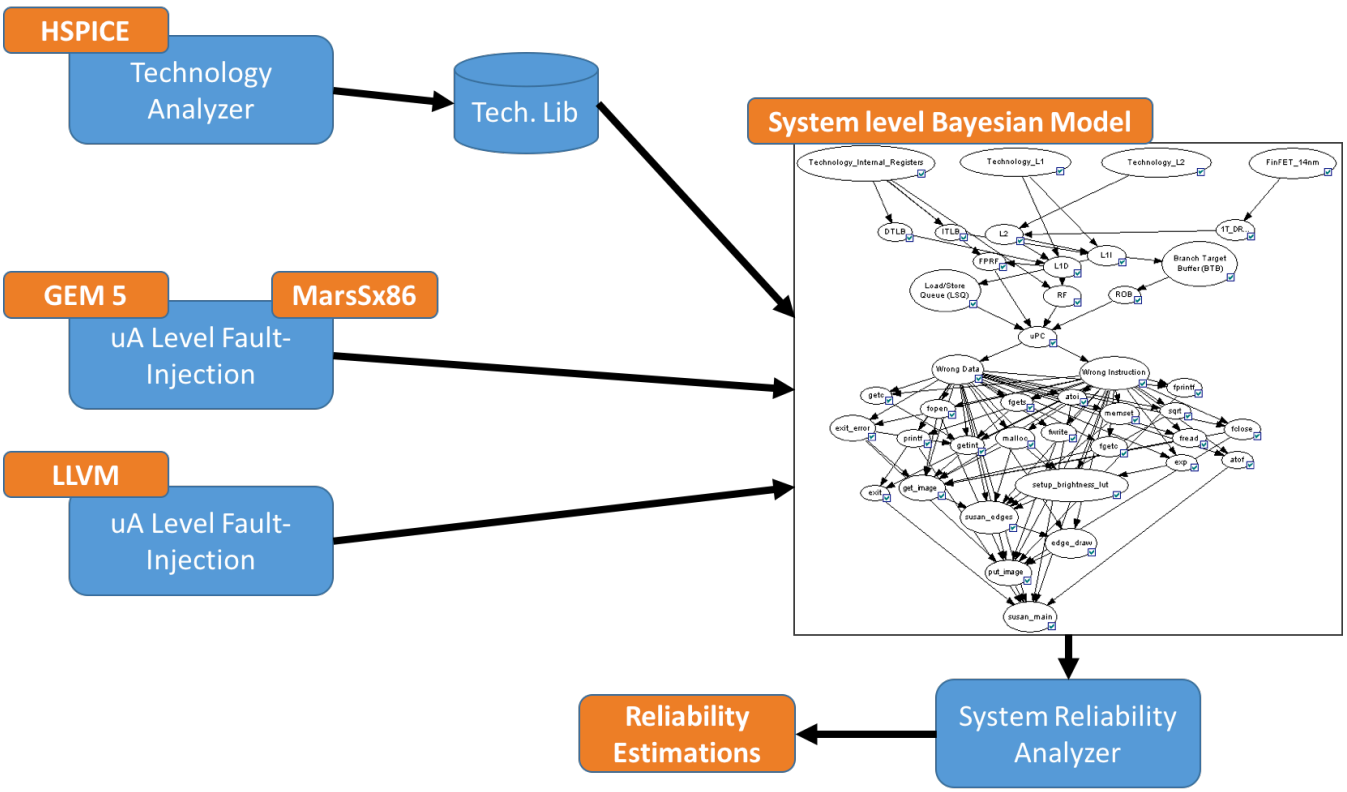

Figure 3. The CLERECO Reliability Analizer Framework.