Cross-Layer Early Reliability Evaluation for the Computing cOntinuum

Cross-Layer Early Reliability Evaluation for the Computing cOntinuum

Stefano Di Carlo[1], Alessandro Vallero[1], Dimitris Gizopoulos[2], Giorgio Di Natale[3], Arnaud Grasset[4], Riccardo Mariani[5], Frank Reichenbach[6]

[1]Politecnico di Torino, Italy. Control and Computer Engineering Departments Email: {name.familyname}@polito.it

[2]University of Athens, Greece. Department of Informatics and Telecommunications Email: dgizop@di.uoa.gr

[3]LIRMM (Universit Montpellier II / CNRS UMR 5506), France. Email: giorgio.dinatale@lirmm.fr

[4]Thales Research & Technology, France. Email: arnaud.grasset@thalesgroup.com

[5]Yogitech s.p.a, Italy. Email: riccardo.mariani@yogitech.com

[6]ABB Corporate Research, Norway. Email: frank.reichenbach@no.abb.com

***Abstract*—Advanced multifunctional computing systems realized in forthcoming technologies hold the promise of a significant increase of the computational capability that will offer end-users ever improving services and functionalities (e.g., next generation mobile devices, cloud services, etc.). However, the same path that is leading technologies toward these remarkable achievements is also making electronic devices increasingly unreliable, posing a threat to our society that is depending on the ICT in every aspect of human activities. Reliability of electronic systems is therefore a key challenge for the whole ICT technology and must be guaranteed without penalizing or slowing down the characteristics of the final products. CLERECO EU FP7 (GA No. 611404) research project addresses early accurate reliability evaluation and efficient exploitation of reliability at different design phases, since these aspects are two of the most important and challenging tasks toward this goal.**

***Index Terms*—reliability, embedded systems, high performance computing**

I. Introduction

Information technology is significantly changing and influencing our society, with the introduction of sophisticated information processing systems. Today's computing is a true continuum that ranges from smartphones to mission-critical datacenter machines, and from desktops to automobiles (Figure 1) [1]. On aggregate, these computing devices represent a total addressable market approaching a billion processors a year, which is expected to explode to more than two billion per year before 2020. Within this computing continuum, the same key technologies and industrial players will act across all computing segments. Therefore, in the near future, we will likely see embedded systems (ES) with High Performance Computing (HPC) characteristics and functionalities, HPC systems used in time- and safety-critical applications, cloud resources used with very different business models, etc.

For more than three decades, electronic industry has evolved by roughly doubling the device density (and corresponding performance) every two years following Moore's law. However, future device integration technology is expected to dramatically reduce the device quality, and therefore the operational reliability of circuits due to higher device variability, manufacturing defects, aging, and higher susceptibility to transient and permanent faults (soft-errors, wear-out) [2]. The great challenge for future technologies is building "dependable" systems on top of unreliable components, which will degrade and even fail during normal lifetime of the chip.

Figure 1: The computing continuum

Conventional design techniques expend significant amount of energy to tolerate the device unpredictability by adding fault tolerance mechanisms at different levels (e.g., technology, architecture, software). However, the rising energy costs needed to compensate for increasing unpredictability are rapidly becoming unacceptable in today's environment where power consumption is often the limiting factor on integrated circuit performance, and energy efficiency is a top concern. In this era, where low reliability threatens to end the benefits of feature size reduction, a holistic approach is required. Such an approach aims at ranging across different computing disciplines, across computing system layers and across computing market segments to have a unique reliability assessment methodology.

On one hand, we need accurate methodologies that reduce the performance and energy tax paid to guarantee correct operation of systems. On the other hand, we need an early system reliability evaluation. Nowadays, system time-to-market (TTM) is already pivotal for the market success of both HPC and ES designs. An increasing number of projects whose reliability was assessed at very late stages of the design cycle, miss their announced market entry dates due to major design changes that in many cases are not affordable. Therefore, early budgeting for reliability has the potential to save significant design effort and resources and has a profound impact on the TTM of a product.

In this scenario, the FP7 Collaboration Project CLERECO [3] addresses early reliability evaluation with a cross-layer approach across different computing disciplines, across computing system layers and across computing market segments. The fundamental objective is to investigate in depth a methodology to accurately assess the reliability through all stages of the design cycle for the future systems of the emerging computing continuum. CLERECO methodology will consider low-level information such as raw failure rates as well as the entire set of hardware and software components of the system that eventually determine the reliability delivered to the end users.

This paper presents an overview of the CLERECO project. The paper is organized as follows: Section II introduces the main CLERECO concepts and ideas while Section III overviews the characteristics of the upcoming CLERECO design framework. In order to provide a better understanding of how CLERECO complements other research efforts in the reliability field, Section IV briefly reviews related works in this field. Finally Section V summarizes the main contributions and concludes the paper.

II. CLERECO RATIONALE

Traditionally, reliability estimation performed at different stages of the design cycle can lead to worst-case decisions and over-designed systems. While the required system reliability can be guaranteed, the cost of the employed reliability mechanisms (in terms or area, energy/power, and performance) and the design time required for their integration and evaluation are both excessive. Moreover, standard reliability evaluation approaches strongly rely on massive and time-consuming simulations and/or fault injection campaigns, which are becoming a bottleneck due the increasing complexity of computing systems.

CLERECO addresses the problem of having an early, fast, and accurate evaluation of computing systems reliability to support design decisions for hardware and software reliability enhancing mechanisms in the system. Such a framework will be a key enabler for the continuation of technology scaling benefits harnessing for several decades. Moreover, it will also enable the implementation of the computing continuum that societal services demand [1]. The benefits of an early and accurate reliability estimation methodology for the computing continuum are many. In particular, design time, energy efficiency and system performance would be improved thanks to CLERECO as it is highlighted in Figure 2.

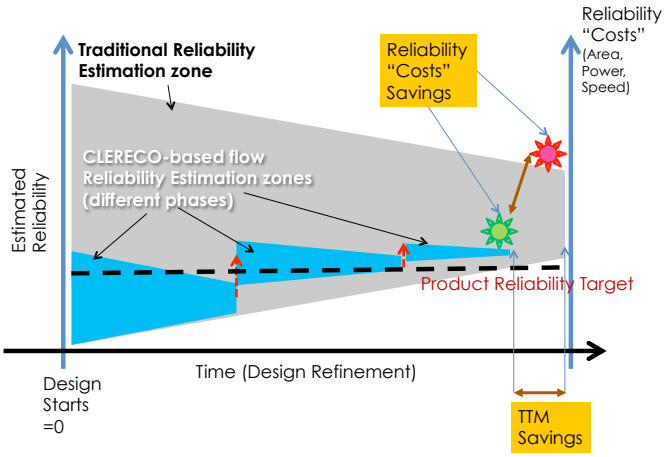

Figure 2: The difference between current methodologies employed to estimate reliability and the new CLERECO approach. It shows the benefits in terms of costs (performance and energy tax) and TTM.

A. Design time

CLERECO framework supports the system design process and avoids under-estimations or over-estimations of the final system's reliability that can adversely affect the system development cost and/or its TTM constraints. Early and accurate identification of reliability weaknesses of the computing system leads to suitable decisions for the employment of hardware and software mechanisms against them. When the reliability weaknesses are identified late in the system design, major redesign costs may be imposed and TTM can be significantly affected. Moreover systems are often over-designed to ensure the fulfillment of the reliability requirements. This is due to the lack of accurate and early estimation methods. Instead the system should be developed to fulfill its purpose at minimal design costs.

B. Energy-efficient reliability

Energy and power consumption is already the primary limitation in the design of computing systems in all domains. Typical reliability solutions based on worst-case scenarios and massive guard-banding at lower layers of abstraction add excessive energy overheads and performance delimiters in scaled technologies. For instance, fault tolerance solutions such as Double Modular Redundancy (DMR) or Triple Modular Redundancy (TMR) are typically used in high-reliability systems. This means 100%-200% more area and energy overhead compared to the unprotected system. CLERECO will enable the employment of more fine grain reliability solutions (e.g., use DMR only in the very vulnerable parts of the system) with a positive impact on the power budget of the system.

C. Reducing the performance impact of reliability

Throughput of systems will also benefit from CLERECO. Reliability mechanisms based on time redundancy, including

error correcting codes or message duplication are computation hungry and they may cause up to 40% of system computational power to be assigned to reliability related tasks. If the faults and their error propagation are known, it is possible to develop just the right mitigations to detect and handle errors in an efficient and accurate way. Again with CLERECO and with the employment of more fine grain reliability solutions, designers will be able to obtain significant improvements in achieved performance.

III. CLERECO FRAMEWORK

The ultimate goal of the CLERECO project is the development of a cross-layer methodology and related tools for systems reliability evaluation that enables: (a) early, fast, and accurate evaluation of computing systems reliability at every stage of the design cycle, and (b) provide designers with a valuable support for reliability related decisions that will in turn allow the design of reliable systems with improved cost-related characteristics (area, energy/power, and performance) and reduced TTM.

Figure 3 details the proposed iterative flow of the CLERECO framework and its role as a key assistant to the design team of the system. At each design stage (refinement phase), the expected system's reliability is evaluated by CLERECO provided the known information about the system's hardware and software components. Depending on the CLERECO evaluation, corresponding design decisions to add or remove reliability mechanisms are taken, eventually leading to a system with just the right level of reliability.

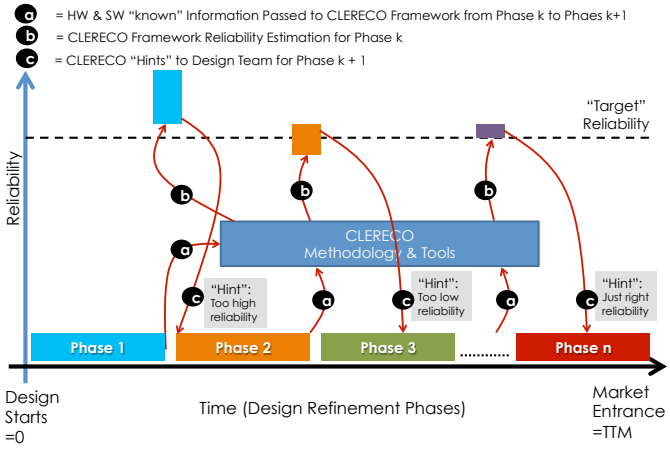

Figure 3: CLERECO reliability oriented design flow

A. CLERECO methodology

In order to develop a cross-layer methodology, CLERECO investigates several aspects of the reliability in the scenario of the computing continuum. In particular CLERECO methodology aims at:

- assessing all different layers of abstraction of the system stack. The lower layers that are related to manufacturing technology will be considered by their inherent raw error producing rates (i.e., as sources of error) while higher layers will be considered by their error propagation profile (i.e., as error blockers/filters, or error amplifiers);
- elaborating a statistical model for the evaluation of the system's reliability to eliminate the need for extremely time consuming and costly solutions such as fault injection campaigns either at the simulation level or on actual system prototypes. The statistical model calculates the expected system's reliability when fed with reliability-related information for the hardware components of the system and the software modules that are expected to run on top of them;
- providing automatic component selection, which will match the design team requirements in terms of performance, power/energy and reliability.

B. The cross-fertilization of the computing continuum

CLERECO motivations and goals come from the strong idea that Embedded Systems (ES) and (HPC) High Performance Computing, two main computing segments seating on both ends of the computing systems spectrum, are converging towards an emerging computing continuum, where the same technologies and industrial players will act across all computing segments, and where we will see ES with HPC functionalities, and HPC platforms used in real-time systems. Both worlds will merge into this computing continuum and face the common reliability challenge that sources from the inherent unreliability of the underlying technologies used in all domains. In CLERECO we observe a synergy and cross-fertilization between these two realities and believe that they may represent a way to ramp-up the research in this field joining competences and know-how.

ES has historically been involved in design of safety-critical applications (e.g., aerospace, automotive, robotics, medical, etc.) requiring high-reliability. ES therefore has established approaches for the design of reliable systems trying to take into account reliability starting from the initial phases of the design and trying to include the side effects that software layers have on the overall system reliability. One of the main contributions of ES is also a very clear and precise set of reliability metrics and standards, which are accurately evaluated at the end of the design process resorting to massive error injection campaigns.

On the other side, HPC has been the leading technology segment that pushed the IC and other technologies toward new limits to achieve higher performance. HPC has therefore put a significant effort in understanding the impact of new technologies on the system's reliability. Together with technology improvement, high performance has been guaranteed by the massive parallelization of HPC computing systems. HPC may therefore provide reliability definitions and metrics for large parallel jobs and multi/many cores systems as well as Reliability Availability Serviceability (RAS) assessment for large heterogeneous systems.

C. CLERECO approach

The approach adopted by CLERECO to achieve these very ambitious targets comprehends the analysis of components used in the computing continuum. In particular, this consists of the characterization and in-depth understanding of the intrinsic reliability properties of future scaled CMOS of emerging post-CMOS technologies, as well as the analysis of the impact of current and future hardware and software components, including their interoperability, on the overall system's reliability (see Figure 4).

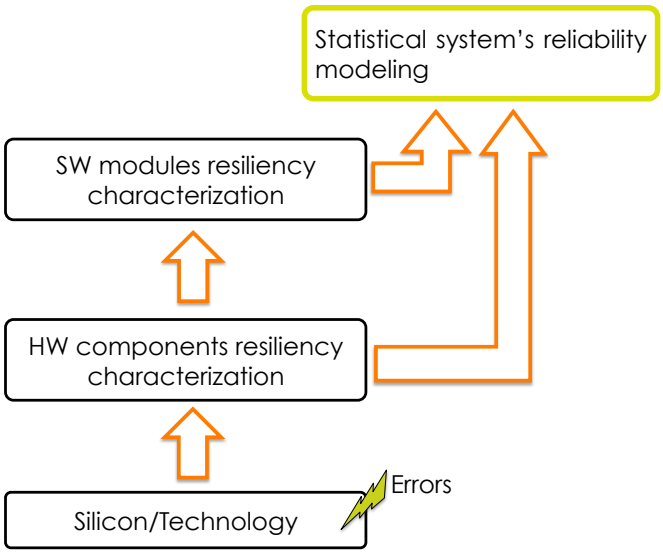

Figure 4: CLERECO early reliability estimation approach

In particular, the CLERECO methodology for early reliability evaluation is comprehensively assessed and validated in advanced designs from different applications provided by the industrial partners for the full stack of hardware and software layers. Moreover, since the target computing is not restricted to a single application domain, but rather to the new computing continuum, all the employed different software layers and the characterization of specific parameters that may affect the reliability are investigated in details.

CLERECO exploits all commonalities that the computing continuum provides, but, at the same time, it also addresses the particularities of each domain, characterizing different phases of the system operation. A complex computing system may need to employ different hardware or software mechanisms (for any of the detection, diagnosis, recovery, and repair actions) depending on its operation phase, i.e., there may be low-reliability or high-reliability phases depending on the workload of the system or the operating conditions. This tuning of the reliability mechanisms is decided based on the CLERECO framework outcome for the expected reliability of the system under a given set of conditions.

CLERECO methodology targets all different design refinement stages, from early conceptual and specification phases, to architectural design phases, up to silicon prototypes validation and also during the system operation in the field (to assist graceful degradation of the system due to aging and wear-out effects). Depending on the design phase on which CLERECO is employed, different reliability-related information for the design components (hardware and software) are used by the CLERECO framework to deliver accurate estimations for the expected system reliability.

IV. Related works

The global research community has been very active in researching technological and architectural methods to improve reliability of specific components and systems. However the CLERECO project takes a different direction, trying to understand how the application and combination of these methods impact the overall system's reliability in order to reduce the use of aggressive reliability design solutions and to improve systems performance and energy/power budget while meeting the requested reliability levels. Having this in mind this section will specifically overview how reliability is nowadays evaluated at different design stages.

A. Reliability prediction at circuit and gate-level

Circuit-level reliability estimation tries to estimate the probability of a given failure mode at the output of a logic gate hit by a particle or affected by other types of stresses [4], [5]. Today, reliability device simulators have become an integral part of the design process. These simulators successfully model the most significant physical failure mechanisms in modern electronic devices. Moreover, some analyses pointed out that, the performance improvement of a full technological generation step can be lost due to process variations [6]. In response to the need to analyze designs under process variations, researchers have developed statistical timing analysis techniques to be applied for deep sub-micron chip designs [7].

Gate-level reliability estimation moves the focus to the nodes of a netlist [5]. Estimating the error susceptibility of a node requires computing the probability of sensitizing the node with an input vector able to propagate the erroneous value to one of the outputs of the circuit [6]. This however requires the simulation of several random vectors whose number significantly increases with the size of the circuit [4], [5], [8], [9], [10]. Reliability prediction tools now model the failure probability of chips at the end of life by analyzing only the single dominant wear-out mechanisms. Modern prediction tools do not predict the random, post burn-in, failure rate that would be seen in the field.

B. Reliability prediction at processor architectural level

The research community has provided a lot of results in the area of Architectural Vulnerability Factor (AVF) calculation and AVF estimation. The AVF is the probability that a fault in a processor structure will result in a visible error in the final output of a program. Most studies estimate reliability in terms of AVF. Most attempts are offline analysis with complex simulators [11], [12], [13]. This offline estimation is a complex process, requiring many resources to track values and instructions as they travel through a processor. Normally only a

limited number of instructions can be analyzed in a reasonable amount of time. There has been some work on estimating the AVF in real time [14], [15]. Walcott [15] et al. apply statistical analysis using a detailed simulator to analyze the AVF behavior at large scale. Then they use linear regression to explore the relationship between AVF and various microarchitecture level variables. Duan et al. [16] proposes the use of boosted regression trees as a predictive model. Later, Biswas et al. [17] extend this work by calculating and estimating vulnerability over short windows of time, providing better opportunities for reconfigurations. Soundararajan et al. [14] propose a method to estimate AVF for the reorder buffer (ROB) in the processor. Fu et al. [18] explores program reliability/vulnerability phase behavior. They observe that a methodology (based on code structure) shows promise in classifying program reliability phase behavior. They also explore the use of performance counters similar to previous works [15]. However, they only explore the AVF estimation for the issue queue and the reorder buffer in an out-of-order processor.

C. Reliability prediction of caches

Memory structures reliability is hard to predict, and deserve a more specific solution [19]. Following Duan et al. [16] work on using boosted regression trees as a predictive model, Ma et al. [20] developed a model based on Bayesian additive regression trees for the cache memories. Cheng et al. [21] studied the variability in AVF for different cache configurations. Li et al. [22] proposed to use simple modifications to the processor to estimate AVF.

D. Reliability prediction at system level

When analyzing the lifetime reliability of processor-based systems, it is essential to investigate the impact at system level. Srinivasan et al. [23] described a model for lifetime analysis for microprocessors and conducted dynamic reconfigurations based on the model. In this model, authors assume identical vulnerability of devices and uniform device density over the whole chip. Later, Shin et al. [24] made a more fine-grain model where different structures of the processor (e.g., register file, functional units) had different failure mechanisms. However, the model suffered from the same inaccuracies.

Other works predict lifetime reliability based on simulations [25]. Similar to previous works [23], [24] , the failure mechanisms do not consider aging effects, which lead to inaccuracies in the simulation results. Huang and Xu [26] have proposed AgeSim, a simulation framework for evaluating the lifetime reliability of SoCs at system level.

Recently, researchers have begun to explore the system-level impact of variations on power, performance, and reliability. Humenay et al. [27] and Romanescu et al. [28] developed models of process variations on pipelined processors. They show that globally-asynchronous, locally synchronous (GALS) design techniques may offer ways to mitigate the impact of correlated within-die variations, but random variations, particularly within memory structures, cannot be easily addressed with these coarse-grained approaches.

Reliability assessment is a critical topic, especially in large and heterogeneous systems. In fact, reliability evaluation, at system level, often relies on statistical models. Thus, model selection is crucial to obtain trustworthy results, depending on the application field. The research activity in this field is active even if not specifically related to the design flow of complex digital systems. Extreme value theory is often the theoretical foundation to build statistical models to evaluate both lifetime and reliability in several fields [29].

E. Reliability impact of software

Scarce work has focused on systematically including the software into the reliability evaluation process. The work published in [30] analyzes various compiler optimization effects on the AVF of an embedded processor. Similar approaches at the compiler level have been also proposed in [31] and [32]. In [33] the authors proposed a first attempt of performing static analysis of a computer system including its software. An interesting solution that includes the software layer is provided by Sridharan and Kaeli [34]. They propose to compute a Program Vulnerability Factor (PVF) for a set of benchmarks exploited to improve AVF computation for several microprocessors. However, neither the final software workload, nor the full stack are explicitly considered. [35] proposes a solution that considers the impact of the application software running on embedded microprocessors however it does not consider the operating system (OS) and multi-processing solutions in the overall evaluation. One of the few attempts to consider the full software stack including the operating system has been proposed in [36]. This however presents very preliminary results and it is uncertain how it can be extended to modern systems employing massive parallelism.

F. Impact of errors

The impact of soft errors is an important emerging concern in the design and implementation of future microprocessors. A considerable amount of research at the microarchitecture level has conducted fault injection studies using software-based methods [37], [38], [39], [40], [41], [42]. They focus on understanding how errors in low-level circuits or hardware structures manifest at the architecture level. Inside this large body of work about fault injections and soft errors, there are studies that characterize the fault tolerant behavior of the system. The susceptibility of commodity operating systems running on IA-32 and IA-64 microprocessors to soft errors was investigated in [43]. The results indicated that with improved microprocessor support like the Machine Check Architecture (MCA), and a little application knowledge, the system can reduce the need for reboots due to the detected soft errors. Additionally, previous works [11], [37] have noticed that not all faults at the low levels affect program correctness.

Identifying the effects of faults in program behavior and which programs is more fault resilient at higher levels has also been deeply studied. Some works [44], [45], [46] have analyzed certain aspects related to program output require-

ments, algorithm features, and cognitive resilience of various applications from the application's standpoint.

Finally, another research vector has explored the possible anomalous symptoms or atypical events caused by faults, and used this information to perform error detection at different levels [47], [48], [49]. This class of research tries to characterize and catch selected events, symptoms, to diagnose the likely presence of a failure caused by soft errors.

V. Conclusions

Increasing reliability of the next generation computing systems while reducing the negative impact of reliability solutions on the TTM, power budget and performance of the system is a key achievement for supporting the ICT technology in the upcoming years. New approaches to precisely evaluate the reliability of a system at early design stages and therefore to take into account different abstraction levels and both software and hardware components are therefore mandatory.

The CLERECO project addresses this problem aiming at the development of a promising framework to adequately deal with an increasingly electronic devices unreliability. The CLERECO consortium including a significant industrial involvement both in the ES and HPC computing sectors, creates a good opportunity to achieve this goal with the potential to build a new framework for the efficient design of systems that achieve tomorrows requirements in terms of performance, survivability, reliability, cost and power dissipation.

VI. Acknowledgments

This research has been supported by the 7th Framework Program of the European Union through the CLERECO Project, under Grant Agreement 611404. Moreover, the work has been supported in part by EUs European Social Fund (ESF) and Greek national funds under the Thales/HOLISTIC project.

References

[1] D. Buchholz and J. Dunlop, "The future of enterprise computing: Prepare for compute continuum." [Available Online]: http://goo.gl/KYb0H8, May 2011.

[2] S. Guertin and M. White, "CMOS reliability challenges the future of commercial digital electronics and nasa." [Available Online]: http://goo.gl/p0gxld, 2010.

[3] CLERECO Consortium, "Cross-layer early reliability evaluation for the computing continuum official website." [Available Online]: http://www.clereco.eu, 2013.

[4] M. Omana, G. Papasso, D. Rossi, and C. Metra, "A model for transient fault propagation in combinatorial logic," in *Proceedings of the 9th IEEE On-Line Testing Symposium, 2003. IOLTS 2003.*, pp. 111–115, July 2003.

[5] A. Maheshwari, I. Koren, and N. Burleson, "Techniques for transient fault sensitivity analysis and reduction in vlsi circuits," in *Proceedings of the 18th IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems, 2003. DFTS 2003.*, pp. 597–604, Nov. 2003.

[6] K. Bowman, S. Duvall, and J. Meindl, "Impact of die-to-die and within-die parameter fluctuations on the maximum clock frequency distribution for gigascale integration," *IEEE Journal of Solid-State Circuits*, vol. 37, pp. 183–190, Feb 2002.

[7] A. Agarwal, D. Blaauw, and V. Zolotov, "Statistical timing analysis for intra-die process variations with spatial correlations," in *Proceedings of the International Conference on Computer Aided Design, 2003. ICCAD 2003.*, pp. 900–907, Nov 2003.

[8] K. Mohanram and N. Touba, "Cost-effective approach for reducing soft error failure rate in logic circuits," in *Proceedings of the International Test Conference, 2003. ITC 2003.*, vol. 1, pp. 893–901, Sept 2003.

[9] M. Sonza Reorda and M. Violante, "Accurate and efficient analysis of single event transients in vlsi circuits," in *Proceedings of the 9th IEEE On-Line Testing Symposium, 2003. IOLTS 2003.*, pp. 101–105, July 2003.

[10] P. Shivakumar, M. Kistler, S. Keckler, D. Burger, and L. Alvisi, "Modeling the effect of technology trends on the soft error rate of combinational logic," in *Proceedings of the International Conference on Dependable Systems and Networks, 2002. DSN 2002.*, pp. 389–398, 2002.

[11] S. S. Mukherjee, C. Weaver, J. S. Emer, S. K. Reinhardt, and T. M. Austin, "A systematic methodology to compute the architectural vulnerability factors for a high-performance microprocessor.," in *MICRO*, pp. 29–42, ACM/IEEE, 2003.

[12] X. Li, S. V. Adve, P. Bose, and J. A. Rivers, "Softarch: An architecture level tool for modeling and analyzing soft errors.," in *Proceedings of the International Conference on Dependable Systems and Networks, 2005. DSN 2005.*, pp. 496–505, IEEE Computer Society, June 2005.

[13] N. J. Wang, A. Mahesri, and S. J. Patel, "Examining ace analysis reliability estimates using fault-injection.," in *ISCA* (D. M. Tullsen and B. Calder, eds.), pp. 460–469, ACM, 2007.

[14] N. Soundararajan, A. Parashar, and A. Sivasubramaniam, "Mechanisms for bounding vulnerabilities of processor structures.," in *Proceedings of the 34th Annual International Symposium on Computer Architecture, 2007. ISCA 2007.*, pp. 506–515, ACM, 2007.

[15] K. R. Walcott, G. Humphreys, and S. Gurumurthi, "Dynamic prediction of architectural vulnerability from microarchitectural state.," in *Proceedings of the 34th Annual International Symposium on Computer Architecture, 2007. ISCA 2007.*, pp. 516–527, ACM, 2007.

[16] L. Duan, B. Li, and L. Peng, "Versatile prediction and fast estimation of architectural vulnerability factor from processor performance metrics," in *Proceedings of the IEEE 15th International Symposium on High Performance Computer Architecture, 2009. HPCA 2009.*, pp. 129–140, IEEE, 2009.

[17] A. Biswas, N. Soundararajan, S. S. Mukherjee, and S. Gurumurthi, "Quantized AVF: A means of capturing vulnerability variations over small windows of time." [Available Online] https://cheetah.cs.virginia.edu/~gurumurthi/papers/selse09_qavf.pdf, 2009.

[18] X. Fu, J. Poe, T. Li, and J. A. B. Fortes, "Characterizing microarchitecture soft error vulnerability phase behavior.," in *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2006. MASCOTS 2006.*, pp. 147–155, IEEE Computer Society, Sept 2006.

[19] A. Biswas, P. Racunas, R. Cheveresan, J. S. Emer, S. S. Mukherjee, and R. Rangan, "Computing architectural vulnerability factors for address-based structures.," in *Proceedings of the 32Nd Annual International Symposium on Computer Architecture 2005. ISCA 2005*, pp. 532–543, IEEE Computer Society, may 2005.

[20] A. Ma, Y. Cheng, and Z. Xing, "Accurate and simplified prediction of avf for delay and energy efficient cache design.," *J. Comput. Sci. Technol.*, vol. 26, no. 3, pp. 504–519, 2011.

[21] Y. Cheng, A. Ma, Y. Tang, and M. Zhang, "Accurate vulnerability estimation for cache hierarchy," in *Proceedings of the 7th International Conference on Networked Computing and Advanced Information Management 2011. NCM 2001.*, pp. 7–14, IEEE, 2011.

[22] X. Li, S. V. Adve, P. Bose, and J. A. Rivers, "Online estimation of architectural vulnerability factor for soft errors," in *Proceedings of the 35th International Symposium on Computer Architecture, 2008. ISCA 2008.*, pp. 341–352, IEEE, 2008.

[23] J. Srinivasan, S. V. Adve, P. Bose, and J. A. Rivers, "The case for lifetime reliability-aware microprocessors," *Proceedings of the 31st Annual International Symposium on Computer Architecture, 2004. ISCA 2004*, vol. 32, no. 2, p. 276, 2004.

[24] J. Shin, V. V. Zyuban, Z. Hu, J. A. Rivers, and P. Bose, "A framework for architecture-level lifetime reliability modeling.," in *Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2007. DSN 2007*, pp. 534–543, IEEE Computer Society, June 2007.

[25] T. S. Rosing, K. Mihic, and G. D. Micheli, "Power and reliability management of socs.," *IEEE Trans. VLSI Syst.*, vol. 15, no. 4, pp. 391–403, 2007.

[26] L. Huang and Q. Xu, "Agesim: A simulation framework for evaluating the lifetime reliability of processor-based socs.," in *Proceedings of the Design, Automation, and Test in Europe Conference 2010. DATE 2010*, pp. 51–56, IEEE, March 2010.

[27] E. Humenay, D. Tarjan, W. Huang, and K. Skadron, "Impact of parameter variations on multicore architectures," in *Workshop on Architectural Support for Gigascale Integration (ASGI-06, held in conjunction with ISCA-33)*, 2006.

[28] B. Romanescu, S. Ozev, and D. Sorin, "Quantifying the impact of process variability on uniprocessor behavior," in *Workshop on Architectural Reliability*, 2006.

[29] R. L. Smith, "Statistics of extremes, with applications in environment, insurance, and finance," *Monographs on Statistics and Applied Probability*, vol. 99, pp. 1–78, 2004.

[30] T. M. Jones, M. F. O'Boyle, and O. Ergin, "Evaluating the effects of compiler optimisations on AVF," in *Proceedings of the Workshop on interaction between compilers and computer architecture (INTERACT-12)*, 2008.

[31] V. Sridharan and D. R. Kaeli, "Eliminating microarchitectural dependency from architectural vulnerability.," in *Proceedings of the IEEE 15th International Symposium on High Performance Computer Architecture, 2009. HPCA 2009.*, pp. 117–128, IEEE Computer Society, Feb 2009.

[32] J. A. Butts and G. S. Sohi, "Dynamic dead-instruction detection and elimination.," in *Proceedings of the 10th International Conference on Architectural Support for Programming Languages and Operating Systems, 2002. ASPLOS X.* (K. Gharachorloo, ed.), pp. 199–210, ACM Press, 2002.

[33] A. Benso, S. Di Carlo, G. Di Natale, and P. Prinetto, "Static analysis of seu effects on software applications.," in *Proceedings of the International Test Conference, 2002. ITC 2002.*, pp. 500–508, IEEE Computer Society, 2002.

[34] V. Sridharan and D. R. Kaeli, "Using pvf traces to accelerate avf modeling," in *Proceedings of the IEEE Workshop on Silicon Errors in Logic-System Effects*, pp. 23–24, 2010.

[35] A. Savino, S. Di Carlo, G. Politano, A. Benso, A. Bosio, and G. Di Natale, "Statistical reliability estimation of microprocessor-based systems.," *IEEE Trans. Computers*, vol. 61, no. 11, pp. 1521–1534, 2012.

[36] V. Sridharan and D. R. Kaeli, "The effect of input data on program vulnerability," in *Workshop on System Effects of Logic Soft Errors (SELSE-5)*, 2009.

[37] N. J. Wang, J. Quek, T. M. Rafacz, and S. J. Patel, "Characterizing the effects of transient faults on a high-performance processor pipeline.," in *Proceedings of the International Conference on Dependable Systems and Networks, 2004. DSN 2004.*, pp. 61–70, IEEE Computer Society, June 2004.

[38] G. P. Saggese, N. J. Wang, Z. Kalbarczyk, S. J. Patel, and R. K. Iyer, "An experimental study of soft errors in microprocessors.," *IEEE Micro*, vol. 25, no. 6, pp. 30–39, 2005.

[39] D. Thaker, D. Franklin, J. Oliver, S. Biswas, D. Lockhart, T. Metodi, and F. Chong, "Characterization of error-tolerant applications when protecting control data," in *Proceedings of the IEEE International Symposium on Workload Characterization, 2006.*, pp. 142–149, IEEE, Oct 2006.

[40] S. Z. Shazli, M. A. Abdul-Aziz, M. B. Tahoori, and D. R. Kaeli, "A field analysis of system-level effects of soft errors occurring in microprocessors used in information systems.," in *Proceedings of the IEEE International Test Conference, 2008. ITC 2008.* (D. Young and N. A. Touba, eds.), pp. 1–10, IEEE, Oct 2008.

[41] M. Rahman, B. R. Childers, and S. Cho, "Stealth works: Emulating memory errors," in *Proceedings of the First International Conference on Runtime Verification*, RV'10, (Berlin, Heidelberg), pp. 360–367, Springer-Verlag, 2010.

[42] S. Pan, Y. Hu, and X. L. 0001, "IVF: Characterizing the vulnerability of microprocessor structures to intermittent faults.," *IEEE Transactions on Very Large Scale Integration (VLSI)*, vol. 20, pp. 777–790, May 2012.

[43] A. Messer, P. Bernadat, G. Fu, D. Chen, Z. Dimitrijevic, D. J. F. Lie, D. Mannaru, A. Riska, and D. S. Milojicic, "Susceptibility of commodity systems and software to memory soft errors.," *IEEE Trans. Computers*, vol. 53, no. 12, pp. 1557–1568, 2004.

[44] M. Breuer, "Multi-media applications and imprecise computation," in *Proceedings of the 8th Euromicro Conference on Digital System Design, 2005.*, pp. 2–7, IEEE, Aug 2005.

[45] X. Li and D. Yeung, "Application-level correctness and its impact on fault tolerance," in *Proceedings of the IEEE 13th International Symposium on High Performance Computer Architecture, 2007. HPCA 2007.*, pp. 181–192, Feb 2007.

[46] L. Leem, H. Cho, J. Bau, Q. A. Jacobson, and S. Mitra, "ERSA: Error resilient system architecture for probabilistic applications.," in *Proceedings of the Design, Automation Test in Europe Conference Exhibition 2010. DATE 2010.*, pp. 1560–1565, IEEE, March 2010.

[47] P. Racunas, K. Constantinides, S. Manne, and S. S. Mukherjee, "Perturbation-based fault screening.," in *Proceedings of the IEEE 13th International Symposium on High Performance Computer Architecture, 2007. HPCA 2007.*, pp. 169–180, IEEE Computer Society, Feb 2007.

[48] N. J. Wang and S. J. Patel, "Restore: Symptom-based soft error detection in microprocessors.," *IEEE Trans. Dependable Sec. Comput.*, vol. 3, no. 3, pp. 188–201, 2006.

[49] P. Ramchandran, S. Adve, V. Adve, and Y. Z. M.-L. Li, "Swat: An error resilient system," in *4th Workshop on Silicon Errors in Logic - System Effects (SELSE - IV)*, 2008.